MODUL PRAKTIKUM

Bahasa Pemrograman

(menggunakan JAVA)

Dengan NetBeans dan Ireport

NetBeans

Melati Suci Mayasari, M. Kom

Sekolah Tinggi Manajemen Informatika dan Komputer

ATMA LUHUR

PRAKATA

Bahasa pemrograman telah banyak disediakan untuk membuat sebuah aplikasi baik yang berskala kecil atau berskala besar dan menjawab berbagai kebutuhan pengguna. Bahasa pemrograman tersebut ada yang bersifat gratis, ada pula yang bersifat membutuhkan biaya untuk menggunakannya. Salah satu bahasa pemrograman yang bersifat gratis adalah bahasa pemrograman JAVA yang merupakan bahasa pemrograman berorientasi objek. Dalam modul praktikum ini Bahasa Java akan digunakan untuk memperkenalkan konsep pemrograman Berorientasi Objek.

Modul ini disusun untuk membantu dan memudahkan mahasiswa STMIK Atma Luhur Pangkalpinang dalam mengikuti mata kuliah Bahasa Pemrograman 2, tetapi tidak menutup kemungkinan bagi pembaca yang lain untuk menjadikan modul ini sebagai salah satu referensi dalam mempelajari pemrograman.

Modul ini lebih ditujukan kepada para programmer tingkat pemula. Modul ini berisi pengenalan dan dasar pemrograman berorientasi objek diantaranya konsep object oriented programming sampai dengan proses membuat program berorientasi objek berbasis GUI. Modul ini juga memperkenalkan proses membuat cetakan berupa hasil transaksi dan laporan. Adapun tools yang digunakan dalam modul ini adalah sebuah IDE yang populer dan banyak digunakan untuk membuat program java. IDE java tersebut NetBeans, sedangkan untuk membuat cetakan menggunakan ireport.

Tak lupa penulis menyampaikan banyak terima kasih kepada seluruh blogger di dunia maya yang telah bersedia membagikan ilmunya untuk penulis bagikan keseluruh para pembaca melalui modul ini. Tak lupa juga penulis menyampaikan terima kasih kepada para pengembang software Netbeans, JDK dan Ireport yang telah bersedia membagikan software ini secara gratis (open source).

Ditengah kemampuan modul ini yang sangat terbatas, semoga sedikit uraian ilmu yang terkandung didalam modul ini dapat bermanfaat bagi para pembacanya dalam mempelajari dan mempraktekkan bahasa pemrograman dengan java.

Pangkalpinang, Maret 2016

Penyusun

DAFTAR ISI

PENDAHULUAN

1. Sekilas Tentang Java

Java merupakan bagian dari bahasa pemrograman yang berorientasi objek. Mudahnya, berbicara mengenai bahasa pemrograman adalah sebuah bahasa yang berisi baris-baris perintah untuk berkomunikasi dengan komputer dan menghasilkan sebuah program. Ada banyak bahasa pemrograman, mulai dari bahasa pemrograman tingkat rendah sampai dengan bahasa pemrograman tingkat tinggi. Java adalah bahasa pemrograman yang tergolong ke dalam bahasa pemrograman tingkat tinggi karena lebih mudah di operasikan.

Sama seperti bahasa yang lain, java juga memiliki sintaks, tata bahasa dan aturan penulisan sendiri.

Java lahir pada tahun 1991, diciptakan oleh sebuah tim yang bernama green dan berjalan selama 18 bulan. Green dimotori oleh James Gosling, Mike Sheridan, Bill Joy dan Patrick Naughton beserta 9 programmer lainnya. Mereka adalah insinyur dari Sun Microsystem. Awalnya bahasa Java di beri dengan nama "oak", yaitu nama dari sebuah pohon yang tumbuh di depan jendela ruang kerja James Gosling. Tetapi, nama tersebut tidak digunakan karena sebelumnya sudah ada perangkat lunak yang di rilis dengan nama "oak". Akhirnya nama "oak" diganti dengan nama "java", nama ini diambil dari kopi murni yang digiling langsung dari biji (kopi tubruk) kesukaan Gosling. Pada tahun 1996, java versi pertama dari java di keluarkan dengan nama rilis java 1.02.

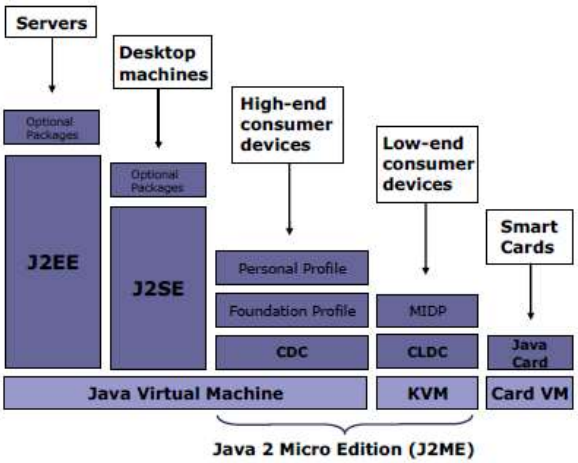
Beberapa fitur yang dimiliki java diantaranya adalah :

a. Java Virtual Machine / Cross Platform
b. Garbage Collection
c. Code Security

Dengan keluarnya versi 1.2, platform Java telah dipilah-pilah menjadi beberapa edisi yaitu The Standard Edition(J2SE), Enterprise Edition(J2EE), Mobile Edition(J2ME), dan JavaCard API. Masing-masing edisi berisi Java 2 Software Development Kit (J2SDK) untuk mengembangkan aplikasi dan Java 2 Runtime Environment (J2RE) untuk menjalankan aplikasi. Adapun kegunaan dari masing-masing edisi ini ditunjukkan pada tabel berikut.

J2SE – Java 2 Platform, Standard Edition	Aplikasi Desktop
J2EE – Java 2 Platform, Enterprise Edition	Aplikasi enterprise dengan fokus pada pengembangan sisi webserver, termasuk servlet,JSP,EJB, dan XML
J2ME – Java 2 Platform, Micro Edition	Perangkat Mobile
JavaCard	Smart Cards

Adapun platformnya yang digambarkan pada JENI (Java Education Network Indonesia) adalah sebagai berikut :



2. Karakteristik Bahasa Pemrograman Berorientasi Objek (OOP)

Object Oriented Programming (OOP) adalah sebuah bahasa pemrograman yang memandang segala sesuatu menjadi sebuah objek. Paradigma dari OOP adalah menyelesaikan masalah dengan merepresentasikan masalah ke model objek.
Karakteristik dari Object Oriented Programming adalah :

- Encapsulation / Enkapsulasi

  Disebut juga dengan pembungkusan, yaitu melindungi program dan data yang sedang diolah agar tidak diakses secara sembarangan oleh program lain. Dalam java, dasar enkapsulasi adalah class. Variabel atau methode pada sebuah class tidak dapat diakses oleh class lain dengan menjadikan class tersebut bersifat **private**, atau menjadikan class tersebut bersifat **protected**, yang hanya bisa diakses oleh turunannya (inheritance) atau menjadikan class tersebut bersifat **public**, sehingga bisa diakses oleh sembarang class.

- Inheritance / Inheritansi

  Disebut juga dengan turunan. Prinsipnya adalah sebuah class dapat diturunkan dari class yang lain. Class yang menurunkan ke class lain disebut dengan superclass, parent class atau base class atau kelas induk, sedangkan class yang merupakan turunan disebut sebagai subclass, child class atau derived class atau class turunan. Class turunan secara otomatis memiliki sifat (variable) dan kelakuan (behavior, method) yang dimiliki oleh super class-nya. Class turunan bisa menambahkan fitur atau behavior dengan mendefinisikan suatu methode di dalam class turunan tersebut.

- Polimorphisme / Polimorfisme

  Polimorfisme secara bahasa dapat diartikan dengan memiliki banyak bentuk. Kegunaan dari polimorfisme adalah agar dapat mendefinisikan beberapa konstruktor atau metode dengan karakteristik yang berbeda-beda agar nantinya

dapat digunakan untuk kasus-kasus yang berbeda. Method atau perilaku yang sama tapi implementasinya/caranya yang berbeda-beda.

3. Aplikasi Bahasa Pemrograman Java

Ada beberapa aplikasi (IDE – Integrated Development Environment) yang dapat digunakan untuk mengimplementasikan Bahasa Java, diantaranya adalah sebagai berikut :

a. Notepad
b. Textplus
c. Editplus
d. JCreator
e. Netbeans.
f. Crimson Editor
g. Eclipse
h. BlueJ
i. Dr.Java
j. JDeveloper

Dalam modul ini, untuk membuat program Java, aplikasi yang akan digunakan untuk mengimplementasikannya adalah Jcreator 4.50 dan NetBeans 6.9.1 yang akan diuraikan lebih lanjut pada pembahasan berikutnya.

Selain aplikasi diatas, dalam membuat program java, juga diperlukan aplikasi pendukung yaitu JDK dan JRE.

JDK (Java Development Kit) digunakan untuk mengakses pustaka/library atau data yang dimiliki oleh java, selain itu JDK juga diperlukan untuk mengubah kode yang ditulis dengan bahasa Java (bahasa yang dimengerti manusia) menjadi byte code (bahasa yang dimengerti komputer), proses ini dikenal dengan istilah compile atau kompilasi. Setelah kode di compile, program java siap untuk dijalankan di komputer (execute/eksekusi).

JRE (Java Runtime Environment) adalah aplikasi yang digunakan untuk menjalankan program yang dibuat dengan java, dalam hal ini program yang sudah di kompilasi.

Perbedaan dari kedua aplikasi tambahan ini adalah JDK digunakan untuk membuat program, sedangkan JRE digunakan untuk menjalankan program. Jadi kalau hanya ingin menjalankan program, maka hanya perlu menggunakan aplikasi JRE. Tetapi, pada saat penginstalan JDK di komputer, secara otomatis JRE pun akan ikut terinstall di komputer.

PRAKTIKUM DESAIN
PROGRAM SEDERHANA

1. **Pengenalan IDE NeatBeans dan JCreator**

   NetBeans adalah IDE (Integrated Development Environment) pada bahasa pemrograman JAVA. Selain Eclipse dan Sun Java Creator, Netbeans adalah salah satu IDE Java yang populer dan banyak digunakan. IDE merupakan lingkup pemrograman yang diintegrasikan kedalam suatu aplikasi perangkat lunak yang menyediakan pembangun Graphic User Interface (GUI), suatu text atau kode editor, suatu compiler atau interpreter dan suatu debugger.

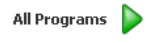
   NetBeans merupakan software development yang bersifat open source, artinya netbeans dapat dikatakan sebagai free software. Netbeans berbasis visual dan event driven. Sama seperti IDE yang lainnya, misalnya Borland Delphi dan Microsoft Visual Studio.

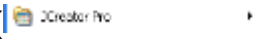
   Aplikasi JCreator adalah aplikasi untuk membuat program Java yang dibuat oleh Xinox Software. JCreator ditulis dalam bahasa C/C++ sehingga lebih cepat (dan menggunakan memori lebih sedikit) dari kebanyakan IDE yang lain. Beberapa versi dari JCreator adalah JCreator 2.50, JCreator 3.0, JCreator 3.50, JCreator 4.00 dan JCreator 4.50. Modul ini digunakan JCreator versi 4.50, lebih tepatnya JCreator 4.50.010 yang dirilis pada 10 Januari 2008.

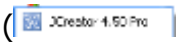
2. **Menjalankan Aplikasi JCreator**

   Untuk menjalankan aplikasi JCreator, dapat dilakukan paling tidak dengan dua cara yaitu :

   a. Klik start (

 ), kemudian pilih All Programs (

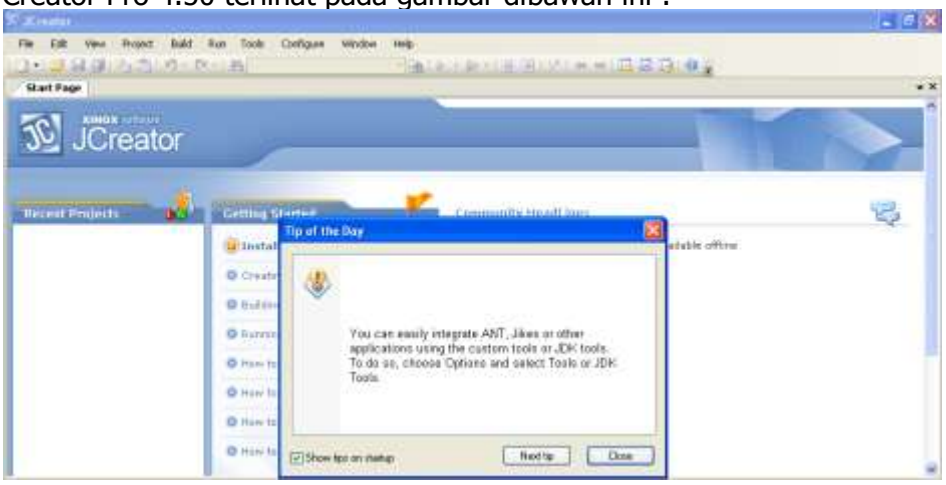
 ), dilanjutkan dengan memilih JCreator Pro (

 ), langkah terakhir mengklik JCreator Pro 4.50 (

).

   b. Klik ganda shortcut JCreator Pro 4.50 yang berada di desktop. 

   Setelah salah satu dari dua langkah diatas dilaksanakan, maka tampilan awal dari JCreator Pro 4.50 terlihat pada gambar dibawah ini :



   Klik Tombol **Close** kotak dialog **Tip of the Day.**

3. **Struktur Utama dan Sifat Bahasa Java**

Setiap bahasa pemrograman memiliki struktur bahasa sendiri yang menjadi ciri khasnya. Setiap bahasa pemrograman juga memiliki aturan sendiri yang bisa saja berbeda ataupun sama dengan aturan bahasa pemrograman yang lain.

Salah satu ciri dan konsep pokok dari pemrograman berorientasi objek yang juga dipunyai oleh bahasa pemrograman java adalah kelas (class). Dalam pembuatan program dengan java, secara tidak langsung juga pendefinisian atau pembentukan kelas. Kelas merupakan unit pembentuk program. Seluruh pembentuk program diwadahi dan ditampung di suatu kelas. Java dapat terdiri dari banyak kelas, dan sebuah kelas dapat terdiri dari banyak methode. Namun biasanya sebuah file hanya terdiri dari satu kelas yang disimpan dengan nama kelas itu sendiri. Aturan untuk nama kelas di java sangat umum yaitu nama harus dimulai huruf, setelah itu boleh kombinasi huruf dan angka.

Struktur utama Bahasa Java adalah sebagai berikut :

```
public class namakelas
{
   public static void main ( String [ ] args )
   {
       statement ;
       statement ;
       statement ;
   }
}
```

Fungsi main() harus ditetapkan sebagai berikut :

- public berarti metode dapat dipanggil dari manapun di dalam dan di luar kelas
- static berarti adalah sama untuk seluruh instant dari kelas
- void berarti metode tidak mengirim apapun setelah selesainya.
- main berarti metode awal yang dijalankan

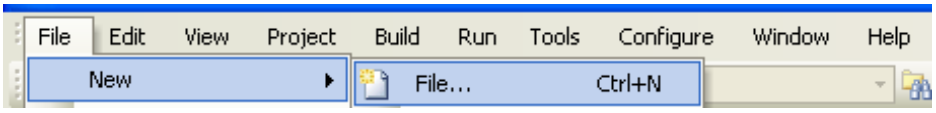
Java juga merupakan bahasa pemrograman yang bersifat case sensitive, yang berarti penulisan menggunakan huruf kapital ataupun huruf kecil pada kode program dapat berarti lain. Pasangan kurung kurawal menandakan awal dan akhir kumpulan dari instruksi-instruksi yang ada di dalam metode. Setiap instruksi atau baris-baris perintah harus diakhiri dengan titik koma untuk menandakan akhir dari satu baris perintah. Walaupun baris perintah ditulis lebih dari satu baris, jika tidak diakhiri dengan titik koma, maka Java akan menganggap baris-baris perintah tersebut merupakan satu baris perintah saja.

4. **Membuat Program Sederhana**

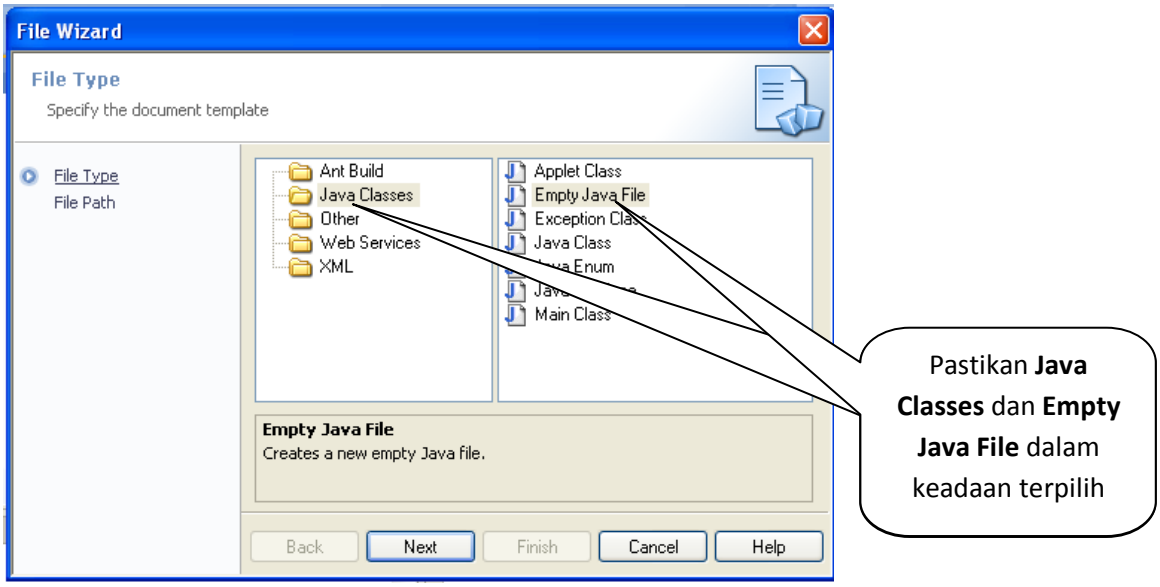
Dari banyak buku yang membahas mengenai bahasa pemrograman, selalu saja dimulai dengan membuat sebuah program yang sederhana sebagai pengantar untuk membahas topik yang lebih spesifik. Program sederhana yang dibuat diantaranya menampilkan output berupa teks "**Hello World**" atau "**Hello Dunia**".

Untuk memulai membuat sebuah program langkah-langkahnya sebagai berikut :

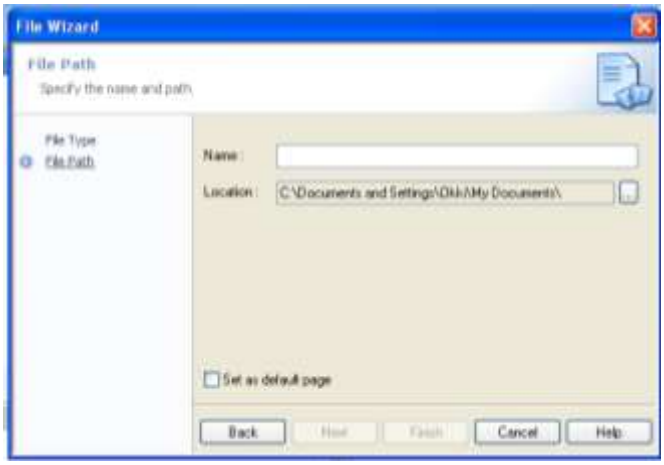
- Klik Menu File -> New -> File…



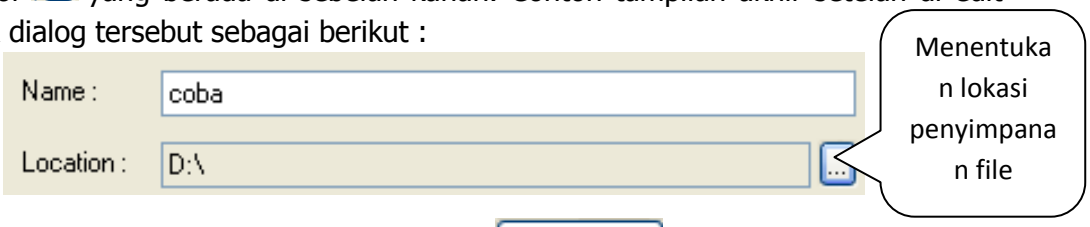
- Setelah diklik muncul kotak dialog sebagai berikut :



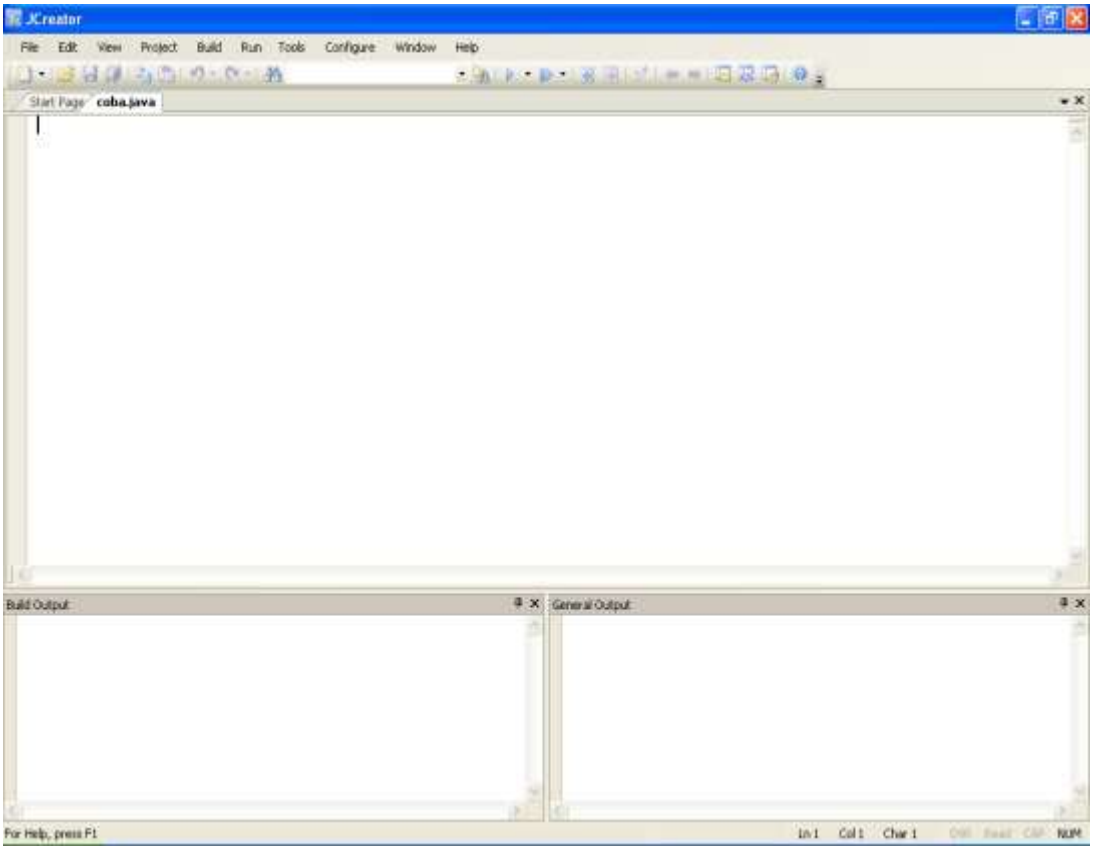
- Pastikan **Java Classes** dan **Empty Java File** dalam keadaan terpilih, dan klik tombol **Next**.
- Setelah tombol **Next** diklik, tampil kotak dialog untuk menentukan nama file dan lokasi tempat penyimpanan dengan tampilan sebagai berikut :



Isi Text Box Name dengan **coba** dan location ditentukan dengan mengklik tombol [...] yang berada di sebelah kanan. Contoh tampilan akhir setelah di edit kotak dialog tersebut sebagai berikut :



- Setelah yakin benar, klik tombol finish (Finish), dan tampilan JCreator sebagai berikut :



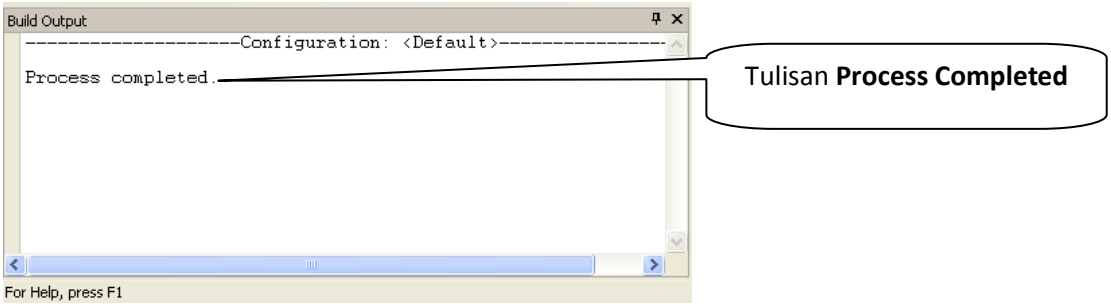
- Ketik baris perintah sebagai berikut :

```
public class coba
{
    public static void main(String args[])
    {
        System.out.println("Hallo Dunia");
    }
}
```

Sekali lagi diingatkan, bahwa bahasa Java adalah bahasa pemrograman yang bersifat case sensitive.

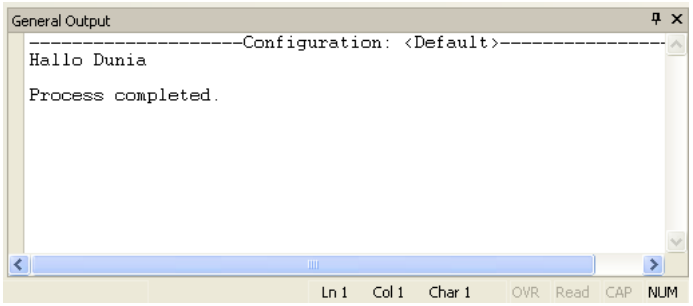
- Setelah selesai klik tombol **Run Project** pada toolbar untuk mengcompile/kompilasi baris perintah (compile apa yaaaa ???).

- Setelah diklik, sekarang perhatikan kotak dialog **Build Output** yang ada di bawah kiri dari aplikasi JCreator. Jika tampilannya bertuliskan sebagai berikut :



Maka proses compile sukses. Tidak ada kesalahan penulisan sintaks program untuk diterjemahkan. Jika bertuliskan bukan seperti diatas, perbaiki terlebih dahulu sintaks yang ditulis dan di lakukan lagi proses compile sampai berhasil.

- Setelah proses compile selesai, perhatikan kotak dialog **General Output** yang ada di bawah kanan dari aplikasi JCreator. Kotak dialog ini berfungsi untuk menampilkan output/hasil dari sintaks program yang telah dikompilasi. Tampilannya sebagai berikut :



- Secara keseluruhan pembuatan program telah selesai.

Dalam pembuatan program-program berikutnya, langkah-langkahnya juga sama dengan langkah-langkah yang telah diuraikan dalam pembuatan program sederhana diatas. Langkah dasar dari pembuatan program menggunakan JCreator.

Latihan :
Buat sebuah program sederhana, program diberi nama dengan **biodata.java**. Program tersebut berisi data pribadi (NIM, Nama, Tempat/Tanggal Lahir, Alamat dan Nomor Telepon) masing-masing !!!.

5. **Input Data**

Dalam pemrograman dengan menggunakan bahasa Java, jika ingin menginput data melalui keyboard, maka harus dideklarasikan dahulu variabel dengan menggunakan fungsi atau *class* yang ada dalam bahasa Java untuk menginput data. Proses ini disebut dengan **instansiasi**. Proses ini juga termasuk kedalam metode *object oriented programming*. Variabel yang dideklarasikan tersebut disebut sebagai **object**, unit terkecil dalam metode *object oriented programming*. Ada tiga macam kelas (*class*) yang dapat digunakan untuk menginstansiasi object untuk menginput data didalam bahasa Java, yaitu :

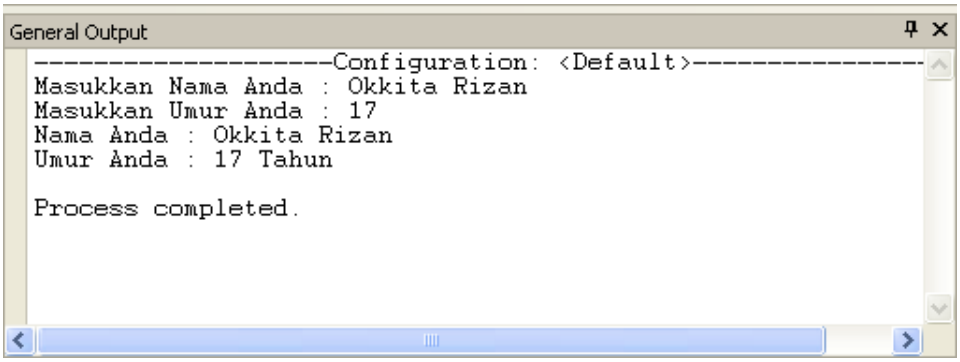
a. Buffered Reader
b. DataInputStream
c. JOptionPane

Class Buffered Reader dan Class DataInputStream, merupakan class yang digunakan untuk menginstansiasi object yang berbasis DOS, sedangkan class JOptionPane digunakan untuk menginstansiasi object dengan basis GUI (Graphical User Interface) / Windows. Class Buffered Reader dan Class DataInputSream merupakan bagian dari *library functions* dalam java, sehingga *library functions* yang membawahi kedua class tersebut harus dicantum atau diikut sertakan (import) kedalam penulisan program menginput. *Library functions* tersebut adalah **java.io.***.

Contoh Program :
Nama Program : input.java
Sintaks Program :

```
import java.io.*;

public class input
{
    public static void main (String args[]) throws IOException
    {
        try
        {

            String nama ;
            int umur ;
            //proses instansiasi
            BufferedReader tulis = new BufferedReader (new InputStreamReader(System.in));

            System.out.print("Masukkan Nama Anda : ");
            nama = tulis.readLine();
            System.out.print("Masukkan Umur Anda : ");
            umur = Integer.parseInt(tulis.readLine());

            System.out.println("Nama Anda : "+nama);
            System.out.println("Umur Anda : "+umur+" Tahun");
        }
        catch (IOException ie)
        {
            System.out.println(ie.getMessage());
        }

    }
}
```

Output Program :



Latihan :

a. Buat program untuk menghitung luas sebuah persegi panjang. Tentukan variabel yang diperlukan, Gunakan class BufferedReader untuk menginput nilai !!!
b. Buat program untuk mengkonversi satuan derajat (Celcius, Fahrenheit dan Kelvin).

PRAKTIKUM DESAIN PROGRAM
DENGAN KONSEP OBJECT ORIENTED

1. Class dan Object

Class ciri dari Object Oriented Programming. Class merupakan kumpulan dari objek yang sejenis. Sedangkan objek merupakan benda, baik secara fisik atau konseptual. Ciri dari objek adalah memiliki atribut/property/data (data member) dan method/behavior/function (member function).

Atribut adalah variabel-variabel yang menyatakan karakteristik suatu objek (what they have). Methode adalah fungsi-fungsi yang bertugas memanipulasi nilai pada data member. Fungsi yang paling sering ada pada sebuah objek adalah fungsi untuk mengubah dan menginformasikan nilai dari data member objek. Methode juga digunakan untuk mengkomunikasikan data dalam class dengan lingkungan luar class. Pengaksesan data objek secara langsung tidak diperbolehkan.

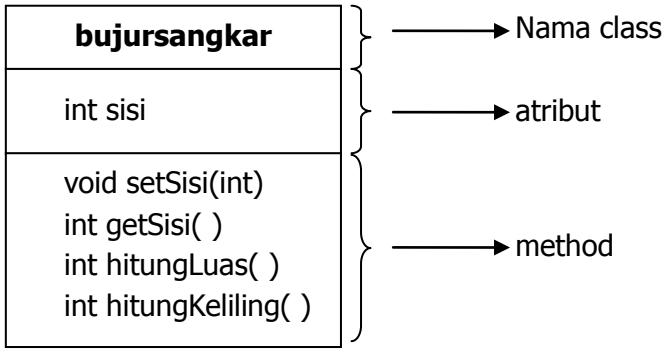
Perbedaan antara class dengan objek adalah Class merupakan desain dan objek merupakan perwujudan suatu Class. Class bersifat abstrak dan objek bersifat kongkrit.

Contoh :

Buatlah sebuah object Bujursangkar yang memiliki atribut yaitu sisi. Object tersebut memiliki fungsi diantaranya menghitung luas dan menghitung keliling. Selain itu object tersebut menyediakan fungsi untuk menentukan nilai dari atribut sisi. Object tersebut juga menyediakan fungsi untuk dapat melihat nilai dari sisi !!!.

Jawab :

Object tersebut dituangkan dalam **class bujursangkar** digambarkan sebagai berikut :



Jika dituangkan ke dalam bentuk sintaks program adalah sebagai berikut :

```
class bujursangkar
{
    int sisi ;
    void setSisi (int s)
     {
         sisi = s;
     }
    int getSisi()
     {
          return(sisi);
     }
    int hitungLuas()
     {
         return(sisi*sisi);
     }
    int hitungKeliling()
     {
        Return(4*sisi);
     }
}
```

2. Enkapsulasi

Enkapsulasi / pembungkusan, seperti yang telah disebutkan diawal berfungsi melindungi program dan data yang sedang diolah agar tidak diakses secara sembarangan oleh program lain. Yang dilindungi adalah variabel dan method dari sebuah class. Sifat dari enkapsulasi yaitu : **private**, **protected** dan **public** dengan karakteristiknya masing-masing.

- Private berkarakteristik agar variabel atau method pada sebuah object/class tidak dapat diakses oleh object/class yang lain.
- Protected berkarakteristik agar variabel atau method pada sebuah object/class dapat diakses oleh object/class turunannya, tetapi tidak dapat diakses oleh object/class yang lain.
- Public berkarakteristik agar variabel atau method pada sebuah object/class dapat diakses oleh object/class yang lain.

Berikut adalah contoh program membuat class bujursangkar yang menggunakan prinsip enkapsulasi. Pada contoh ini, ditunjukkan juga bagaimana membuat sebuah object dari class bujur sangkar. Proses ini disebut sebagai **instansiasi**. Object dari class bujursangkar bernama bjsangkar1.

Nama Program : bjrsangkar_utama.java
Sintaks Program :

```
class bujursangkar
{
      private int sisi ;
      public void setSisi (int s)
        {
             sisi = s;
        }
      public int getSisi()
         {
              return(sisi);
         }
      public int hitungLuas()
         {
             return(sisi*sisi);
         }
       public int hitungKeliling()
         {
            return(4*sisi);
         }

}

public class bjsangkar_utama
{
     public static void main(String args[])
     {
         //instansiasi object
         bujursangkar bjsangkar1 = new bujursangkar();

         bjsangkar1.setSisi(10);

         System.out.println("        BUJUR SANGKAR        ");
         System.out.println("=============================");
         System.out.println("Sisi        = "+bjsangkar1.getSisi());
         System.out.println("Luas        = "+bjsangkar1.hitungLuas());
         System.out.println("Keliling    = "+bjsangkar1.hitungKeliling());
     }
}
```

Output Program :

```
General Output
--------------------Configura
       BUJUR SANGKAR
=========================
Sisi        = 10
Luas        = 100
Keliling    = 40

Process completed.
```

3. Konstruktor

Pada saat pembentukan sebuah class, kadang kala perlu dilakukan pemberian nilai awal pada sebuah objek. Hal ini bertujuan untuk memastikan bahwa pada sebuah objek tidak memiliki nilai bukan untuk objek tersebut. Sebagai contoh, sisi sebuah bujur sangkar harusnya memiliki nilai lebih dari 0 atau tidak memiliki nilai yang bersifat karakter (contoh : sisi = 'sepuluh'). Untuk mengatasi ini dapat menggunakan konstruktor.

Konstruktor adalah metode object oriented programming yang dipanggil secara otomatis pada saat sebuah objek diciptakan. Ciri dari konstruktor adalah mempunyai nama yang sama dengan class dan tidak mempunyai nilai kembalian (return value). Konstruktor terdiri dari 3 bentuk yaitu :

a. Konstruktor tunggal (single constructor)

Sebuah class hanya memiliki satu buah konstruktor. Contoh program sebagai berikut :

Nama Program : bjrsangkar_konst

Sintaks Program :

```
class bjrsangkar
{
      private int sisi ;
      bjrsangkar()
      {
         setSisi(5);
      }
      public void setSisi (int s)
        {
             if (s>5)
             {
                 sisi = s;
             }
        }
      public int getSisi()
         {
              return(sisi);
         }
      public int hitungLuas()
         {
             return(sisi*sisi);
         }
       public int hitungKeliling()
         {
            return(4*sisi);
         }
}
```

```
public class bjrsangkar_konst
{
    public static void main(String args[])
    {
        //instansiasi object
        bjrsangkar bjsangkar1 = new bjrsangkar();
        bjrsangkar bjsangkar2 = new bjrsangkar();
        bjrsangkar bjsangkar3 = new bjrsangkar();

        bjsangkar1.setSisi(6);
        bjsangkar2.setSisi(15);
        bjsangkar3.setSisi(25);

        System.out.println("       BUJUR SANGKAR    ");
        System.out.println("============================");
        System.out.println("Sisi 1     = "+bjsangkar1.getSisi());
        System.out.println("Luas 1     = "+bjsangkar1.hitungLuas());
        System.out.println("Keliling 1 = "+bjsangkar1.hitungKeliling());
        System.out.println("Sisi 2     = "+bjsangkar2.getSisi());
        System.out.println("Luas 2     = "+bjsangkar2.hitungLuas());
        System.out.println("Keliling 2 = "+bjsangkar2.hitungKeliling());
        System.out.println("Sisi 3     = "+bjsangkar3.getSisi());
        System.out.println("Luas 3     = "+bjsangkar3.hitungLuas());
        System.out.println("Keliling 3 = "+bjsangkar3.hitungKeliling());

    }
}
```

b. Konstruktur dengan parameter

Menggunakan konstruktor dengan parameter, dapat menentukan secara langsung nilai pada objek. Dengan langkah ini bermanfaat untuk meringkas beberapa perintah. Contoh program sebagai berikut :

Nama Program : bjrsangkar_konstpar

Sintaks Program :

```
class bujurskr
{
     private int sisi ;
     bujurskr(int ss)
     {
        setSisi(ss);
     }
     public void setSisi (int s)
       {
            if (s>5)
            {
                sisi = s;
            }
            else
            {
                sisi = 1;
            }
       }
     public int getSisi()
        {
             return(sisi);
        }
     public int hitungLuas()
        {
            return(sisi*sisi);
        }
      public int hitungKeliling()
        {
           return(4*sisi);
        }
}
```

```
public class bjrsangkar_konstpar
{
    public static void main(String args[])
    {
        //instansiasi object
        bujurskr bjsangkar1 = new bujurskr(2);
        bujurskr bjsangkar2 = new bujurskr(3);
        bujurskr bjsangkar3 = new bujurskr(6);


        System.out.println("       BUJUR SANGKAR    ");
        System.out.println("============================");
        System.out.println("Sisi 1     = "+bjsangkar1.getSisi());
        System.out.println("Luas 1     = "+bjsangkar1.hitungLuas());
        System.out.println("Keliling 1 = "+bjsangkar1.hitungKeliling());
        System.out.println("Sisi 2     = "+bjsangkar2.getSisi());
        System.out.println("Luas 2     = "+bjsangkar2.hitungLuas());
        System.out.println("Keliling 2 = "+bjsangkar2.hitungKeliling());
        System.out.println("Sisi 3     = "+bjsangkar3.getSisi());
        System.out.println("Luas 3     = "+bjsangkar3.hitungLuas());
        System.out.println("Keliling 3 = "+bjsangkar3.hitungKeliling());

    }
}
```

c. Multiple constructor

Multiple constructor memiliki lebih dari satu konstruktor pada sebuah class. Bentuk seperti ini disebut juga dengan overloading constructor. Syarat pada overloading constructor adalah signature pada setiap konstruktor tidaklah sama. Signature adalah informasi untuk membedakan method seperti nama method, jumlah parameter, tipe data dan nilai kembalian (return value). Contoh program sebagai berikut :

Nama program : bjrsangkar_konstmul

Sintaks program :

```
class bjrskr_multiple
{
     private int sisi ;
     bjrskr_multiple()
     {
        setSisi(10);
     }
     bjrskr_multiple(int ss)
     {
        setSisi(ss);
     }
     public void setSisi (int s)
       {
            if (s>5)
            {
                sisi = s;
            }
            else
            {
                sisi = 1;
            }
       }
```

```
        public int getSisi()
            {
                 return(sisi);
            }
        public int hitungLuas()
            {
                return(sisi*sisi);
            }
         public int hitungKeliling()
            {
               return(4*sisi);
            }
    }

public class bjrsangkar_konstmul
{
    public static void main(String args[])
    {
        //instansiasi object
        bjrskr_multiple bjsangkar1 = new bjrskr_multiple();
        bjrskr_multiple bjsangkar2 = new bjrskr_multiple(6);
        bjrskr_multiple bjsangkar3 = new bjrskr_multiple();

        bjsangkar3.setSisi(8);

        System.out.println("       BUJUR SANGKAR    ");
        System.out.println("============================");
        System.out.println("Sisi 1     = "+bjsangkar1.getSisi());
        System.out.println("Luas 1     = "+bjsangkar1.hitungLuas());
        System.out.println("Keliling 1 = "+bjsangkar1.hitungKeliling());
        System.out.println("Sisi 2     = "+bjsangkar2.getSisi());
        System.out.println("Luas 2     = "+bjsangkar2.hitungLuas());
        System.out.println("Keliling 2 = "+bjsangkar2.hitungKeliling());
        System.out.println("Sisi 3     = "+bjsangkar3.getSisi());
        System.out.println("Luas 3     = "+bjsangkar3.hitungLuas());
        System.out.println("Keliling 3 = "+bjsangkar3.hitungKeliling());

    }
}
```

4. Turunan

Turunan (Inheritance), secara bebas diterjemakan sebagai pewarisan, yaitu sebuah konsep dimana kita membuat sebuah class baru dengan mengembangkan class yang sudah pernah dibuat sebelumnya.

Seperti yang telah diuraikan diatas, Class yang menurunkan ke class lain disebut dengan superclass, parent class atau base class atau kelas induk, sedangkan class yang merupakan turunan disebut sebagai subclass, child class atau derived class atau kelas turunan. Class turunan secara otomatis memiliki sifat (variable) dan kelakuan (behavior, method) yang dimiliki oleh super class-nya. Class turunan bisa menambahkan fitur atau behavior dengan mendefinisikan suatu methode di dalam class turunan tersebut. Salah satu keuntungan dari sifat turunan ini adalah cukup mendefinisikan satu kali saja atribut atau method yang sama pada class supernya dan dapat digunakan di seluruh class turunannya.

Sebagai contoh, buatlah sebuah object kubus. Object kubus tersebut merupakan turunan dari object bujursangkar. Object kubus memiliki atribut sisi dan memiliki fungsi diantaranya adalah menghitung luas dan menghitung volume. Object tersebut juga memiliki fungsi untuk melihat nilai dari sisi !!

Jawab :

Object kubus tersebut akan digambarkan kedalam class kubus dan dibandingkan dengan class bujursangkar yang telah digambarkan sebagai berikut :

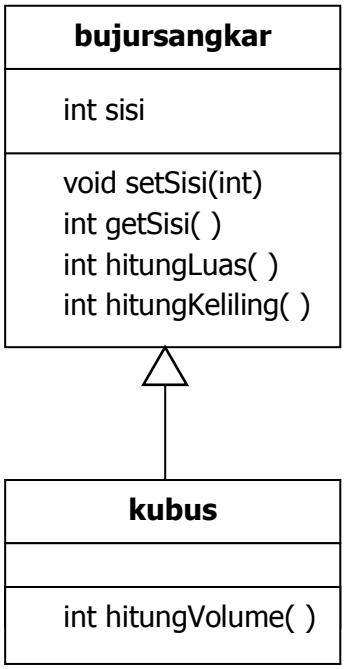
bujursangkar
int sisi
void setSisi(int) int getSisi( ) int hitungLuas( ) int hitungKeliling( )

kubus
int sisi
void setSisi(int) int getSisi( ) int hitungLuas( ) int hitungVolume( )

Dari perbandingan diatas, terlihat bahwa terdapat atribut dan fungsi yang sama antara class bujursangkar dan class kubus. Sehingga, class kubus merupakan **superset** dari class bujursangkar. Metode dari OOP mengizinkan untuk membuat atribut dan methode dari class **superset** yang ada pada kelas **subset**nya. Sehingga class kubus tersebut menjadi sebagai berikut :

kubus
int hitungVolume( )

Sehingga, secara hierarki kedua class tersebut digambarkan sebagai berikut :



Dalam baris perintah program, ciri dari sebuah class yang diturunkan dari class lain adalah adanya perintah **extends** pada nama classnya. Bisa dilihat pada program berikut ini :

Nama program : turunankubus.java
Sintaks program :

```
class bujursangkar
{
      protected int sisi ;
      public void setSisi (int s)
        {
             sisi = s;
        }
      public int getSisi()
         {
              return(sisi);
         }
      public int hitungLuas()
         {
             return(sisi*sisi);
         }
       public int hitungKeliling()
         {
            return(4*sisi);
         }

}
```

```
class kubus extends bujursangkar
{
    public int hitungLuas()
    {
        return(super.hitungLuas()*6);
    }
    public int hitungVolume()
    {
        return (super.hitungLuas()*super.getSisi());
    }
}


public class turunankubus
{
    public static void main (String args[])
    {
        bujursangkar objsangkar = new bujursangkar();
        kubus objkubus = new kubus();

        objsangkar.setSisi(10);
        objkubus.setSisi(20);

        System.out.println("       B U J U R S A N G K A R    ");
        System.out.println("----------------------------------");
        System.out.println(" Sisi           : "+objsangkar.getSisi());
        System.out.println(" Keliling       : "+objsangkar.hitungKeliling());
        System.out.println(" Luas           : "+objsangkar.hitungLuas());
        System.out.println("==================================");
        System.out.println();
        System.out.println("          K  U  B  U  S          ");
        System.out.println("----------------------------------");
        System.out.println(" Sisi           : "+objkubus.getSisi());
        System.out.println(" Luas           : "+objkubus.hitungLuas());
        System.out.println(" Volume         : "+objkubus.hitungVolume());
        System.out.println("==================================");

    }
}
```

Output program :

```
--------------------Configuration: <Def
       B U J U R S A N G K A R
----------------------------------
 Sisi           : 10
 Keliling       : 40
 Luas           : 100
==================================

          K  U  B  U  S
----------------------------------
 Sisi           : 20
 Luas           : 2400
 Volume         : 8000
==================================

Process completed.
```

5. Polimorphisme

Secara mudah polymorphism bisa disamakan dengan method overloading, dimana didalam class bisa terdapat beberapa method dengan nama sama. Seperti yang telah diuraikan diatas, kegunaan dari polimorfisme adalah agar dapat mendefinisikan beberapa konstruktor atau metode dengan karakteristik yang berbeda-beda agar nantinya dapat digunakan untuk kasus-kasus yang berbeda. Method atau perilaku yang sama tapi implementasinya/caranya yang berbeda-beda.

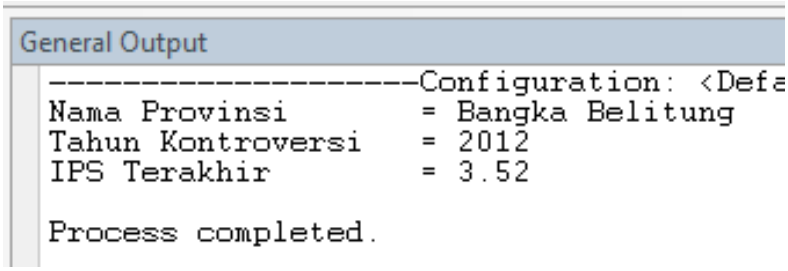
Sebagai contoh, buatlah sebuah class yang bernama tampildata. Class tersebut memiliki sebuah method bernama cetak data. Method tersebut menerapkan konsep polimorpisme, karena berfungsi untuk mencetak data dengan tipe data string, integer, double secara sekaligus. Class tersebut memiliki object yang bernama dataku. Yang dicetak dari object ini adalah kalimat **"Bangka Belitung"**, angka **2010** dan angka **3,52** !!!

Nama program : polimorphisme.java

Sintaks program :

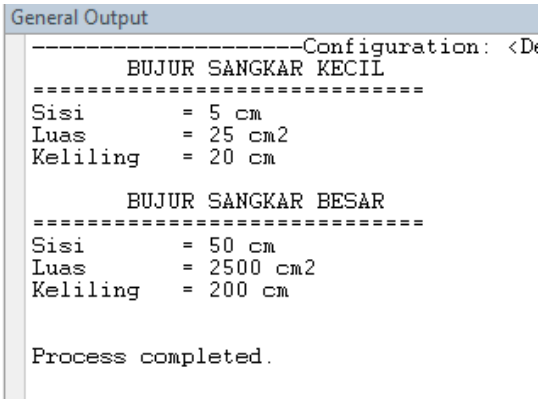
```
class tampildata
{
    public void cetakdata(String x)
    {
        System.out.println(x);
    }
    public void cetakdata(int x)
    {
        System.out.println(x);
    }
    public void cetakdata(double x)
    {
        System.out.println(x);
    }
}

public class polimorphisme
{
    public static void main (String args[])
    {
        tampildata dataku = new tampildata();

        System.out.print("Nama Provinsi       = "); dataku.cetakdata("Bangka Belitung");
        System.out.print("Tahun Kontroversi   = "); dataku.cetakdata(2012);
        System.out.print("IPS Terakhir        = "); dataku.cetakdata(3.52);
    }
}
```

Output program :

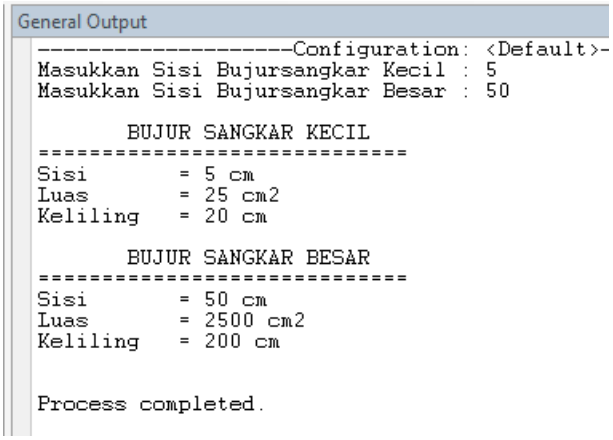


Latihan :

1) Buatlah sebuah **class latbujursangkar** yang memiliki dua buah objek yaitu **bjsangkarkecil** dan **bjsangkarbesar**. Class tersebut memiliki atribut sisi dan memiliki fungsi menghitung luas dan menghitung keliling beserta fungsi untuk menentukan nilai dari atribut sisi. Masing-masing object memiliki sisi yang berbeda. Untuk object bjsangkarkecil memiliki sisi 5 cm sedangkan object bjsangkarbesar memiliki sisi 50 cm. Terapkan konsep enkapsulasi, Sehingga output programnya adalah sebagai berikut :



2) Buat **class latbujursangkar2** dengan ketentuan sama seperti soal no. 1, tetapi baik object bjsangkarkecil dan object bjsangkarbesar, untuk sisinya diinput melalui keyboard. Dengan outputprogram sebagai berikut :



3) Terapkan konsep **inheritance** untuk membuat **class persegipanjang** sebagai **superclassnya** dan **class balok** sebagai **class turunannya**. Class persegipanjang mempunyai atribut panjang dan lebar, sedangkan class balok mempunyai atribut panjang, lebar dan tinggi. Class persegipanjang mempunyai method untuk mengambil lebar dan panjang, menghitung luas dan menghitung keliling. Class balok mempunyai method untuk mengambil lebar, panjang dan tinggi, menghitung luas dan menghitung volume !!!
4) Terapkan konsep **polimorphisme** untuk menampilkan data karyawan yang terdiri dari NIP, NAMA, Alamat, Gol.Darah, Gaji Pokok dan Tunjangan. Data tersebut diinput melalui keyboard !!!

PRAKTIKUM DESAIN PROGRAM
GRAPHICAL USER INTERFACE (GUI)

Membuat program dengan konsep Graphical User Interface (GUI) menjadikan interaksi dengan user lebih menarik dibandingkan dengan menggunakan konsep ini, dalam hal keindahan tampilan dan kemudahan penggunaan program. Tidak sebatas interaksi dengan keyboard saja, user bisa berinteraksi dengan program menggunakan mouse. Beberapa komponen dari GUI adalah windows, menu, textbox, combobox dll.

Bahasa Java menyediakan dua buah package yang menyimpan berbagai class GUI yaitu AWT (Abstract Windows Toolkit) dan SWING. Perbedaan kedua package ini adalah Package AWT terdapat pada **java.awt**, sedangkan SWING terdapat pada **javax.swing** dan package SWING memiliki komponen yang lebih banyak dari AWT. Beberapa komponan dari SWING adalah button, combo box, check box, label, list, scrollbar, textfield, radio button, option pane, progress bar, tabel, menu dan text area. Setiap komponen memiliki fungsi dan cara pembuatan yang berbeda. Semua komponan pada swing diawali dengan huruf "J", misalkannya JButton, JTextArea. Untuk mempraktekkan program GUI ini, akan menggunakan tools NetBeans.

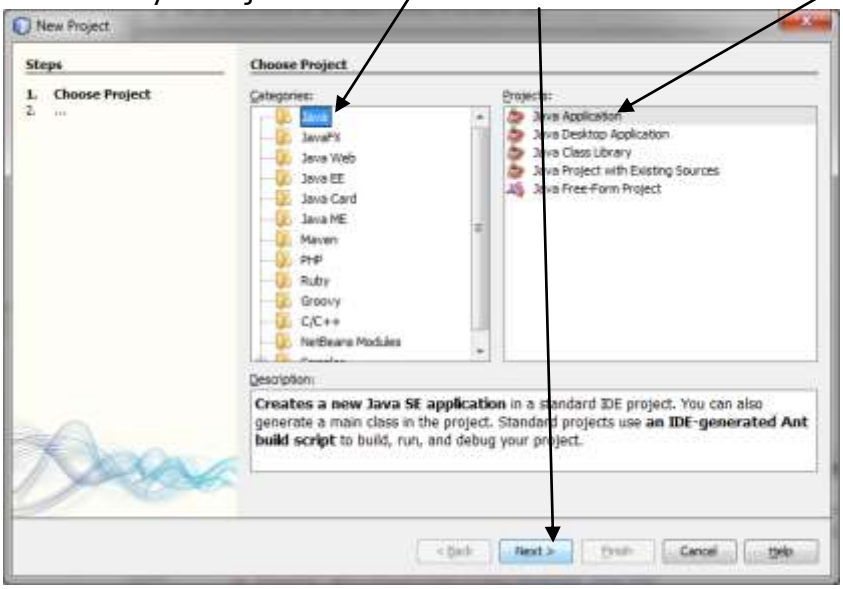
1. Buat Project

Pada beberapa contoh program/form dibawah ini disimpan dalam satu project yang sama. Sebelum membuat form, terlebih dahulu membuat projectnya, langkah-langkahnya sebagai berikut :

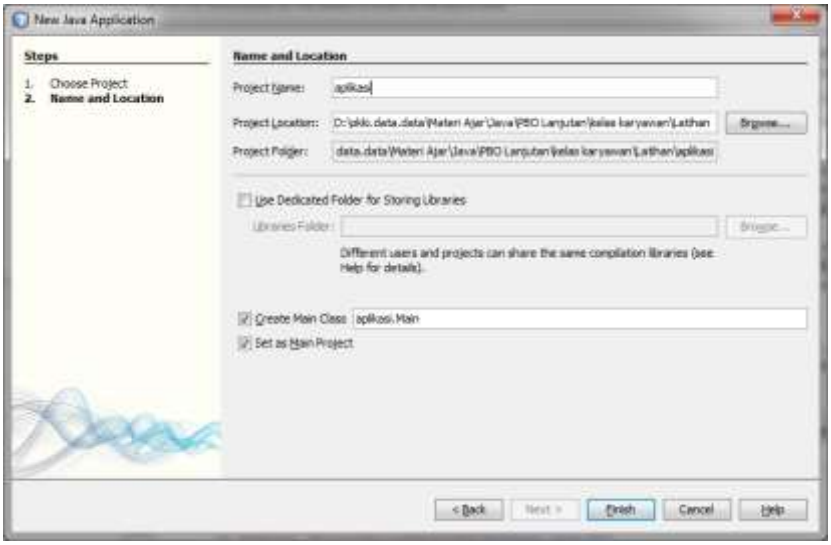
a) Jalankan/Buka Aplikasi NetBeans.

b) Pilih Menu **File -> new project**

c) Pada Bagian **Categories** pilih **Java** dan Bagian **Projects** pilih **Java Application** di sebelahnya. Lanjutkan klik tombol **Next**.



d) Tentukan/buat nama project (bebas tetapi tidak menggunakan kata yang sudah menjadi haknya netbeans). Pada langkah ini juga silahkan juga ditentukan tempat untuk menyimpan projectnya.

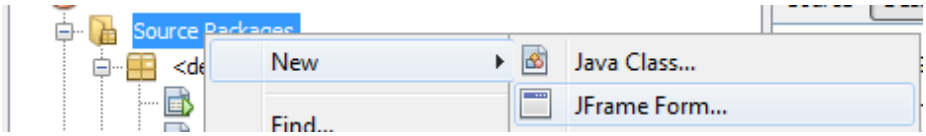


e) Silahkan klik tombol **Finish** dan langkah-langkah untuk membuat project telah selesai.

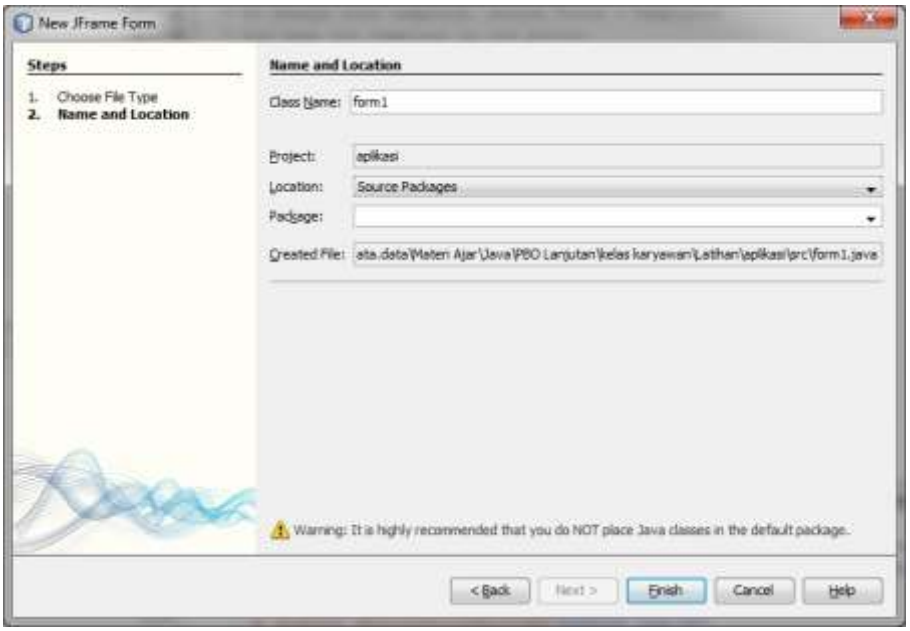
2. Form dengan Jlabel

Langkah-langkahnya sebagai berikut :

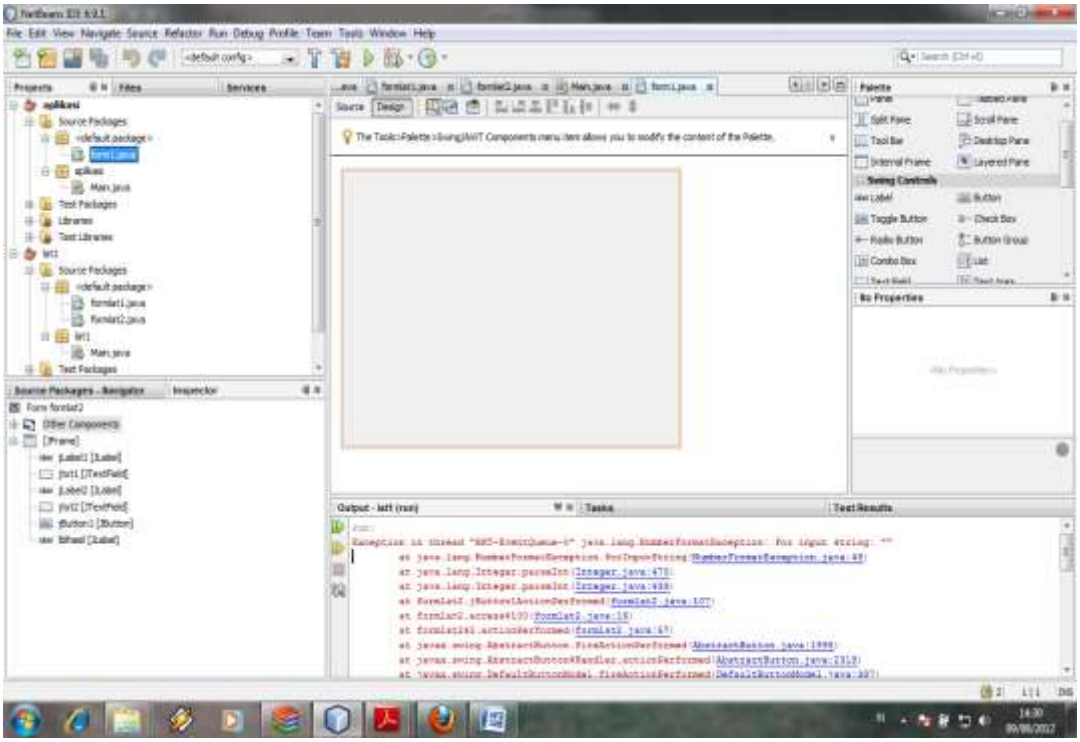
a) Klik kanan pada **source package**, pilih **New** dan pilih **JframeForm**



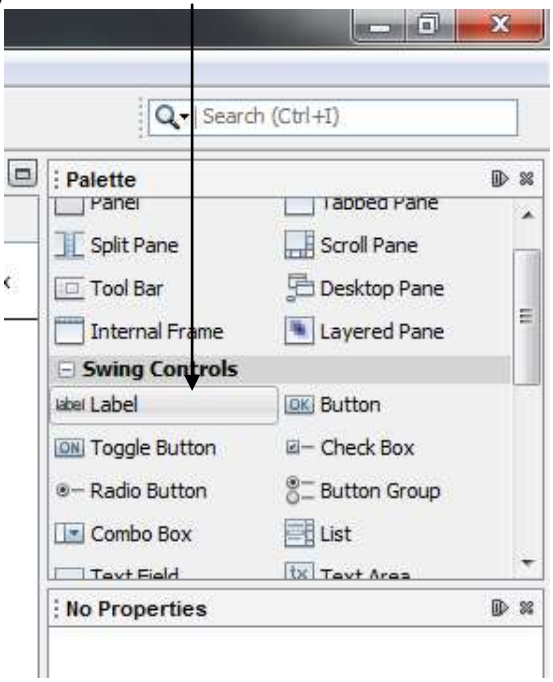
b) Buat nama file untuk form tersebut.



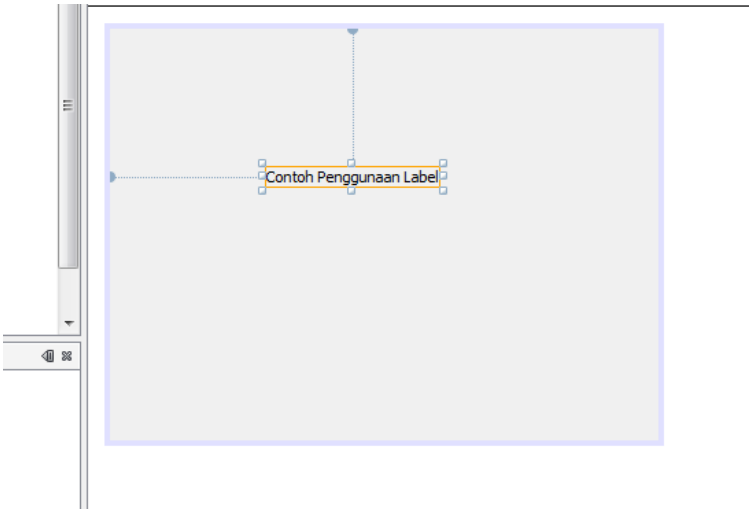
c) Jika nama file sudah dibuat berikut tampilan project.



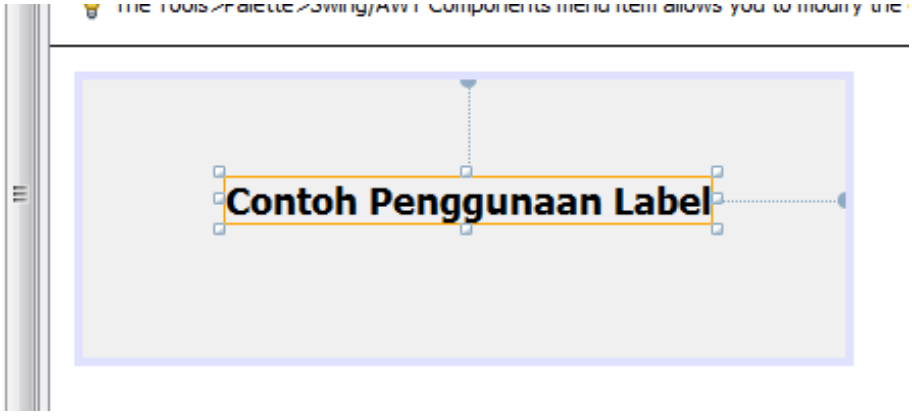
d) Pada **Swing Pallet**, klik item **Label** dan kemudian klik diatas **JFrame**.



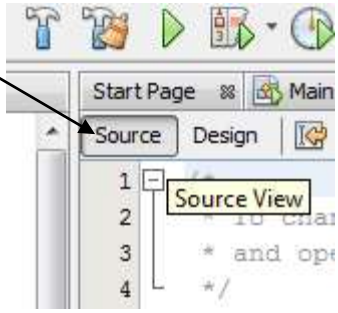
e) Ganti tulisan pada teks dengan contoh tampilan sebagai berikut :



f) Desain ulang **JFrame** dengan tampilan sebagai berikut :



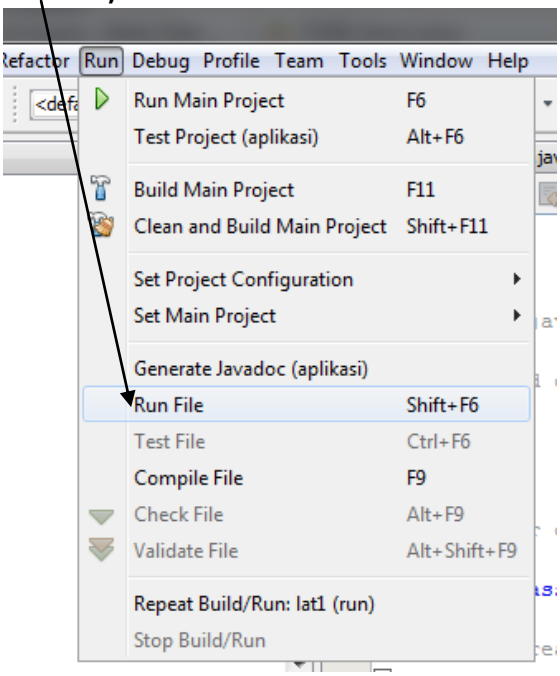
g) Klik tabulasi **Source** dengan tampilan sebagai berikut :



h) Tambahkan dua listing program sebagai berikut:

```
15   */
16  public class form1 extends javax.swing.JFrame {
17
18      /** Creates new form form1 */
19      public form1() {
20          super("Latihan Form 1");
21          setLocation(100, 200);
22          initComponents();
23      }
24
```

i) Form sudah bisa dirunning untuk melihat hasilnya, dengan cara klik menu **Run** dan klik sub menu **Run File,** atau melalui tombol **shift+f6**.

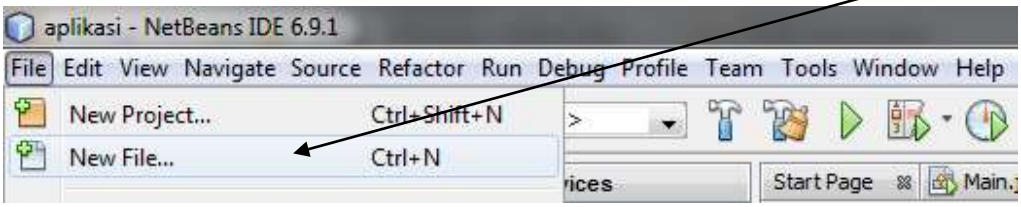


j) Tampilan akhir sebagai berikut :

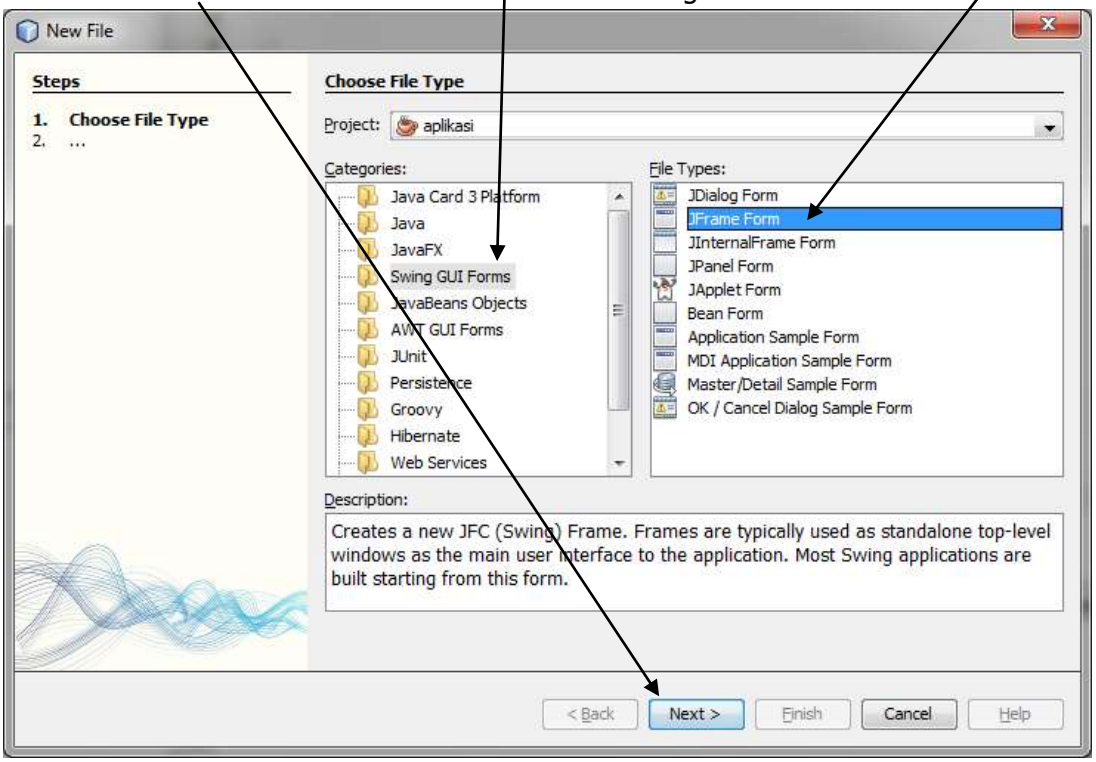
3. **Form dengan Event**

Event adalah respon yang diberikan oleh suatu objek. Berikut adalah contoh respon yang diberikan oleh objek button.

a) Buat form baru dengan cara klik menu **File**, kemudian klik sub menu **New File**.

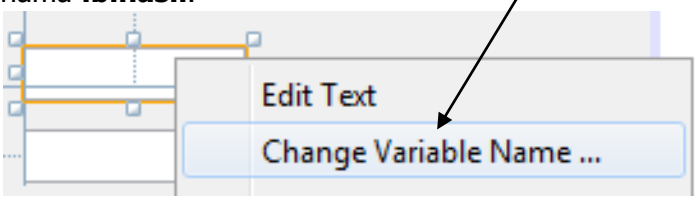


b) Pilih bagian **Categories** pilih **Swing GUI Forms** dan **File Types**nya **JFrame Form**. Klik **Next** dan tentukan nama file sesuai keinginan.



c) Rancang form dengan tampilan sebagai berikut :

d) Komponen yang digunakan berupa **Label**, **Textfield** dan **Button.** Untuk Textfield, ubah masing-masing namanya (**Change Variable Name**) yaitu **txtangka1** dan **txtangka2**. Label yang digunakan untuk menampilkan hasil ganti dengan nama **lblhasil**.



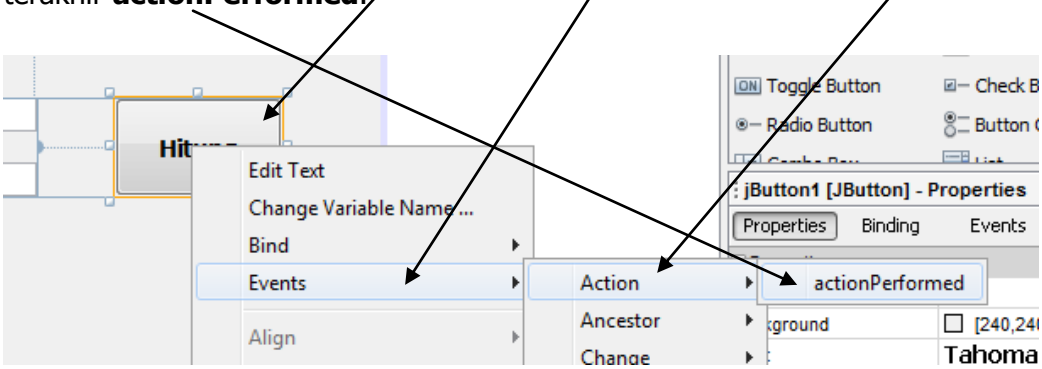
e) Klik tabulasi **source** dan tambahkan listing program berikut untuk membuat mendeklarasikan variabel.

```
public class form2 extends javax.swing.JFrame {
    private String str = "";
    private int a, b;
```

f) Tambahkan listing program pada bagian konstruktor class untuk membuat judul dan menentukan posisi frame.

```
    private int a, b;
    /** Creates new form form2 */
    public form2() {
        super("Penjumlahan");
        setLocation(100,200);
        initComponents();
    }
```

g) Klik kanan pada tombol **Hitung** dan pilih **Events** , dilanjutkan pilih **Action** dan terakhir **actionPerformed**.



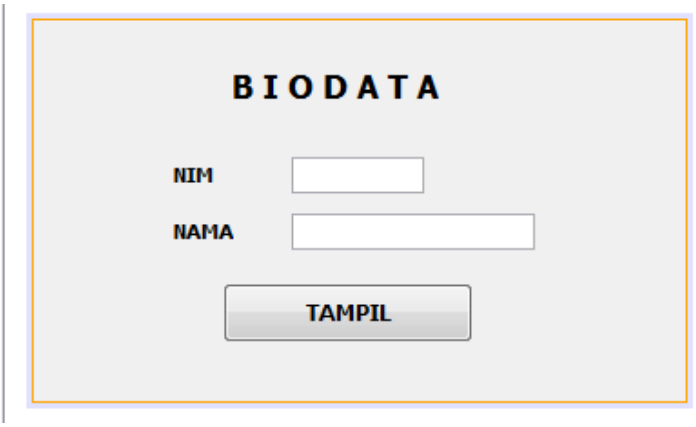
h) Tambahkan listing program sebagai berikut :

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent
        // TODO add your handling code here:
        a = Integer.parseInt(txtangka1.getText());
        b = Integer.parseInt(txtangka2.getText());
        int hasil = a + b;
        lblhasil.setText(str + hasil);
    }
```

i) Program silahkan di running.

4. Form dengan JOptionPane

a) Buat sebuah form baru (masih didalam project yang sama) dan rancang form dengan tampilan seperti berikut. Ubah nama objek sesuai keinginan.



b) Tambahkan listing program pada bagian konstruktor dengan perintah sebagai berikut.

```
public form3() {
    super("Data");
    setLocation(100,200);
    initComponents();
```

c) Klik kanan pada tombol **TAMPIL**, pilih **Events -> Action -> actionPerformed**. Tambahkan listing program sebagai berikut.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String tampil="";
    tampil +="Hai, Perkenalkan Saya mahasiswa STMIK Atma Luhur \n";
    tampil +="dgn NIM  : "+ txtNIM.getText()+"\n";
    tampil +="dan NAMA : "+ txtNAMA.getText() +"\n\n";
    tampil +="Salam....";

    JOptionPane.showMessageDialog(null, tampil,"Biodata",
    JOptionPane.INFORMATION_MESSAGE);
}
```

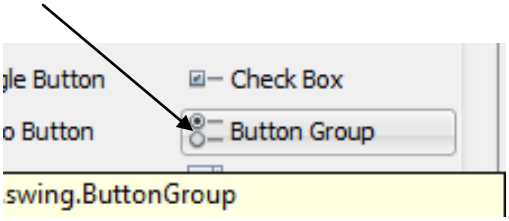
d) Program silahkan di running.

5. Membuat form dengan komponen Radio Button dan Check Box.

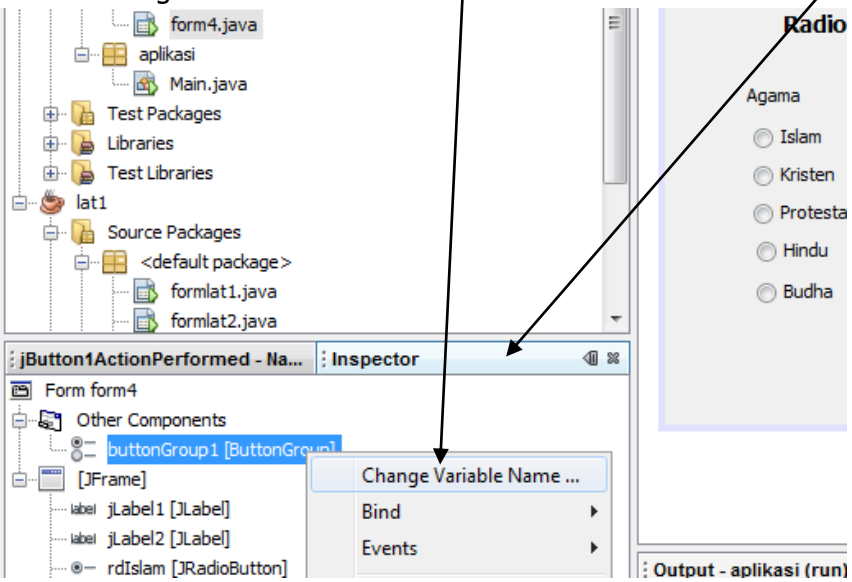
Perbedaan Radio Button dengan Check Box adalah Radio Button bersifat pilihan tunggal pilihan hanya satu, sedangkan untuk check box bersifat pilihan ganda atau boleh memilih lebih dari satu. Untuk Radio Button mesti dimasukkan kedalam komponen Button Group.

a) Buat form baru dengan masih disimpan pada project yang sama. Rancang form dengan tampilan sebagai berikut

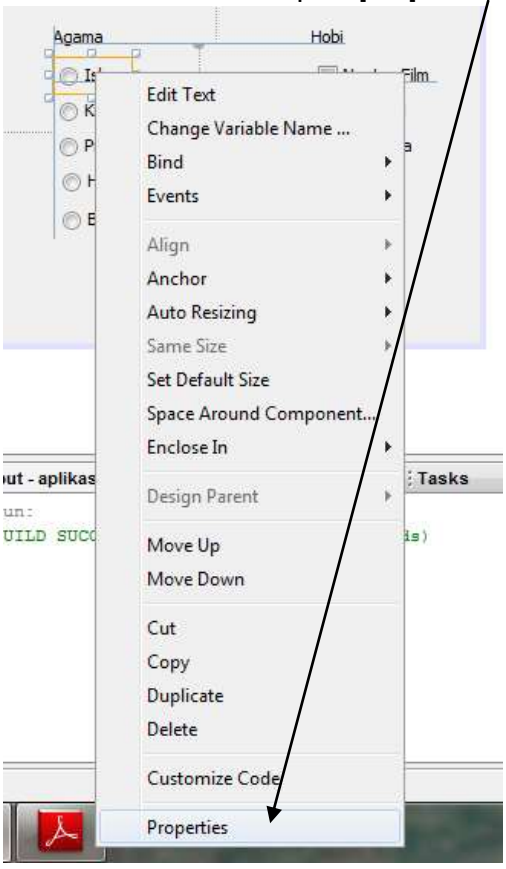
b) Tambahkan objek **Button Group** dan klik sembarang tempat di frame.



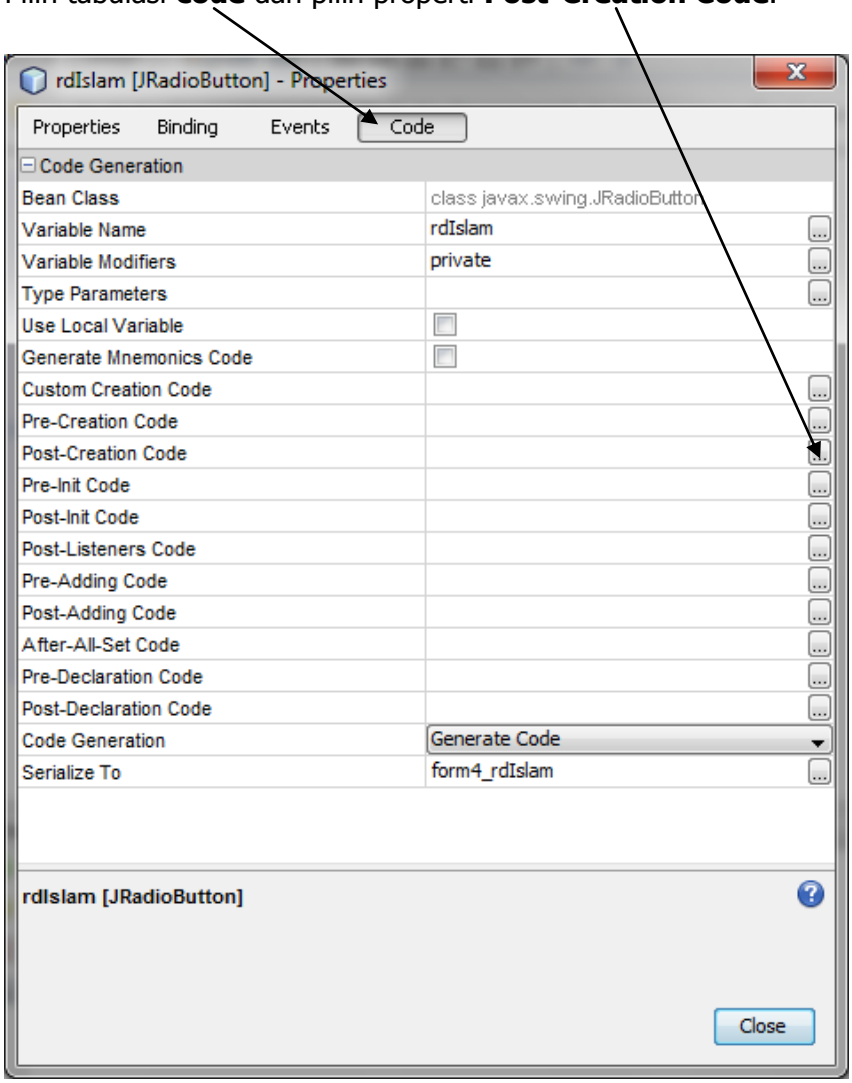
c) Untuk mengubah nama objek dari **Button Group**, lihat di **Jendela Inspector**. Klik Kanan pada **Button Group** , pilih **change variable name** dan ubah nama objeknya sesuai keinginan.



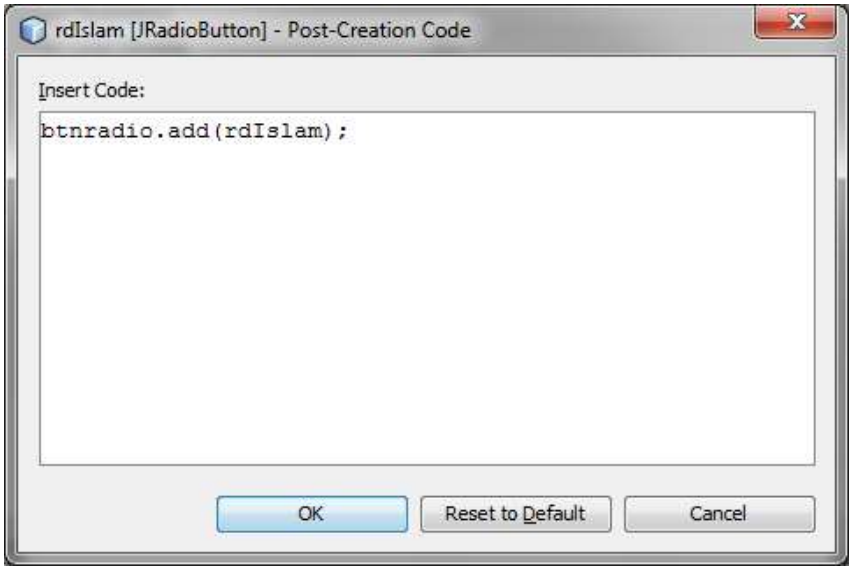
d) Masukkan objek radio masing-masing ke dalam button group dengan cara :
   1) Klik kanan pada **radio button Islam** dan pilih **properties**.



2) Pilih tabulasi **code** dan pilih properti **Post-Creation Code**.



3) Masukkan perintah sebagai berikut, jika sudah klik **OK** :



4) Ulangi langkah yang sama untuk keempat radio button yang lain.

e) Tambahkan listing program pada button **Tampil** dengan perintah sebagai berikut:

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String Tampil="Contoh penggunaan Radio Button dan Check Box \n\n";
        Tampil+="Agama : ";
        if(rdIslam.isSelected())
        {
            Tampil+=""+ rdIslam.getText() +"\n\n";
        }
        else if (rdKristen.isSelected())
        {
            Tampil+=""+ rdKristen.getText() +"\n\n";
        }
        else if (rdProtestan.isSelected())
        {
            Tampil+=""+ rdProtestan.getText() +"\n\n";
        }
        else if (rdHindu.isSelected())
        {
            Tampil+=""+ rdHindu.getText() +"\n\n";
        }
        else if (rdBudha.isSelected())
        {
            Tampil+=""+ rdBudha.getText() +"\n\n";
        }
        else
        {
            Tampil+="Belum Dipilih\n\n";
        }
        Tampil+="Hobi : ";
        if(chBaca.isSelected() && chNonton.isSelected() && chRekreasi.isSelected())
        {
            Tampil+=""+ chNonton.getText() +", "+ chBaca.getText() +" dan ";
            Tampil+=chRekreasi.getText() +"\n";
        }
        else if(chBaca.isSelected() && chNonton.isSelected() )
        {
            Tampil+=""+ chNonton.getText() +" dan "+ chBaca.getText() +"\n";
        }
        else if(chRekreasi.isSelected() && chNonton.isSelected() )
        {
            Tampil+=""+ chRekreasi.getText() +" dan "+ chNonton.getText() +"\n";
        }
        else if(chRekreasi.isSelected() && chBaca.isSelected() )
        {
            Tampil+=""+ chRekreasi.getText() +" dan "+ chBaca.getText() +"\n";
        }
        else if(chNonton.isSelected())
        {
            Tampil+=""+ chNonton.getText() +"\n";
        }
        else if(chRekreasi.isSelected())
        {
            Tampil+=""+ chRekreasi.getText() +"\n";
        }
        else if(chBaca.isSelected())
        {
            Tampil+=""+ chBaca.getText() +"\n";
        }
```

```
        else
        {
            Tampil+="Belum dipilih\n";
        }
        Tampil+="-----------------------------------\n";
        JOptionPane.showMessageDialog(null, Tampil, "Contoh",
        JOptionPane.INFORMATION_MESSAGE);
    }
```

f) Form silahkan di running

PRAKTIKUM DESAIN PROGRAM
FORM DENGAN DATABASE DAN ODBC

Database atau basis data adalah himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa sehingga dapat dimanfaatkan kembali dengan cepat dan mudah. Database sendiri merupakan sekumpulan dari tabel-tabel yang membentuk relasi. Sedangkan tabel berisi record-record yang sejenis. Anggap saja database adalah sebuah lemari arsip. Didalam lemari arsip terdapat banyak dokumen. Dokumen-dokumen tersebut dikelompokkan kedalam map-map tertentu. Map diibaratkan sebuah tabel, sedangkan dokumen adalah recordnya.

Open Database Connectivity (ODBC) merupakan sebuah Aplication Programming Interface (API) untuk konektivitas database. ODBC memungkinkan database dapat diakses dengan aplikasi selain dari program database. Melalui ODBC yang ditunakan untuk mengakses database, termasuk database lokal maupun database server. ODBC didasarkan pada Structure Query Language (SQL) sebagai standar untuk mengakses data. ODBC ini memberikan kemudahan dan kebebasan kepada pengguna untuk memilih jenis database yang ingin digunakan sebagai server, dan pada saat yang sama tetap memiliki kebebasan untuk mengakses jenis database yang lainnya.

Contoh aplikasi dibawah ini memanfaatkan ODBC sebagai jembatan untuk menghubungkan database dengan form. Database yang digunakan adalah Microsoft Access dengan toolsnya Microsoft Office Access 2007.

1. Membuat Project

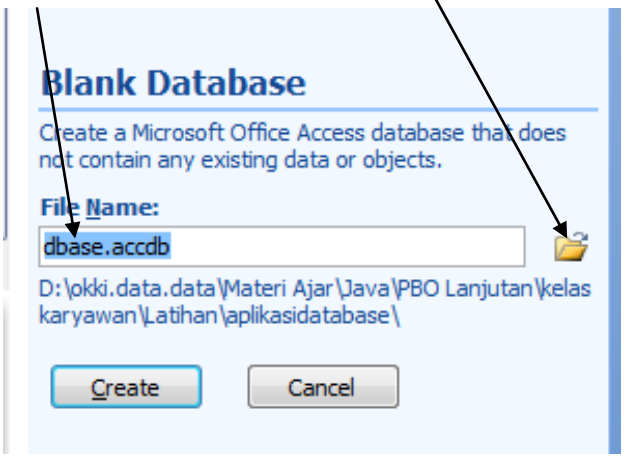
Sebelum membuat database dan konektivitasnya, ada baiknya membuat project baru terlebih dahulu. Hal ini dilakukan untuk mempermudah menentukan lokasi database yang nantinya akan dibuat. Buka aplikasi **Netbeans** -> Buat project baru, tentukan nama project sesuai dengan keinginan.

2. Membuat Database

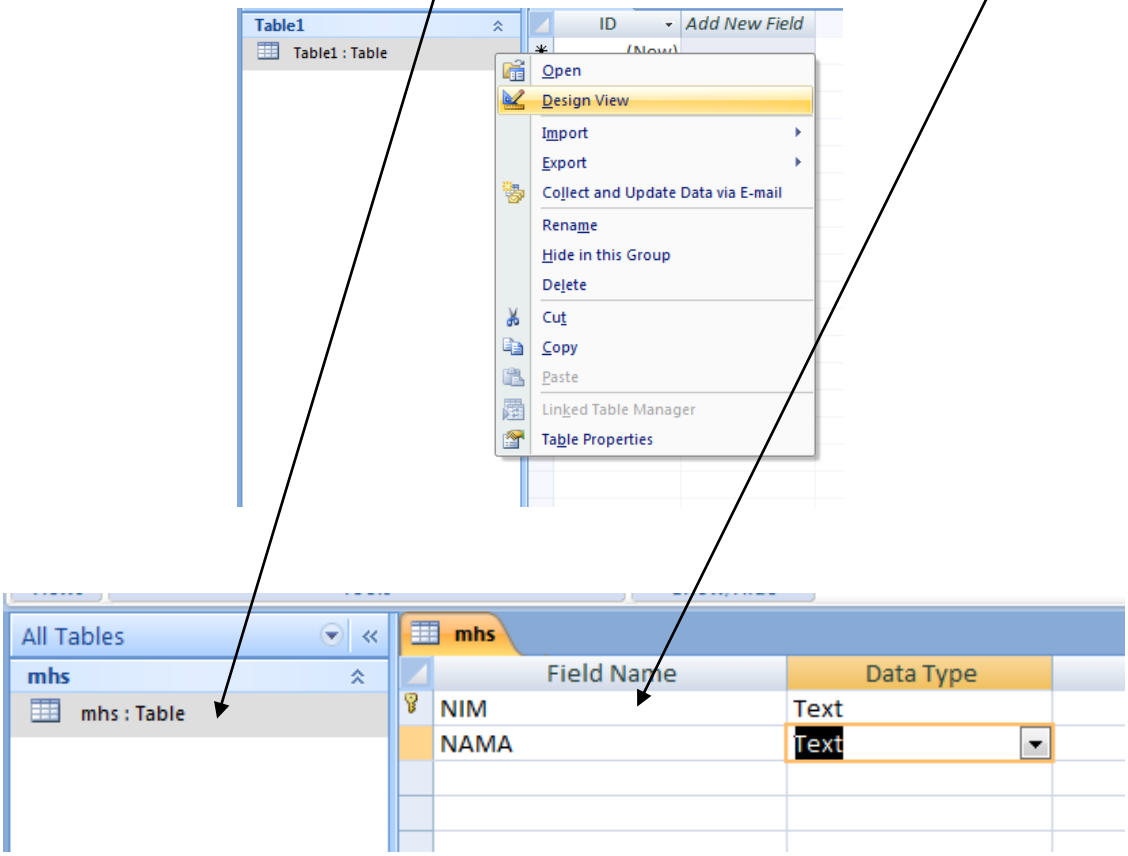
Langkah-langkah sebagai berikut :

a) Buka/jalankan **Microsoft Office Access 2007**.

b) Buat database baru dan tentukan **lokasi penyimpanannya**. Database disimpan didalam project baru yang telah dibuat. Sebagai contoh database disimpan dengan nama **dbase.accdb**.



c) Buat tabel baru dengan nama **mhs**. Buat dua buah field, masing-masing **NIM (text)** dan **NAMA (text)**.

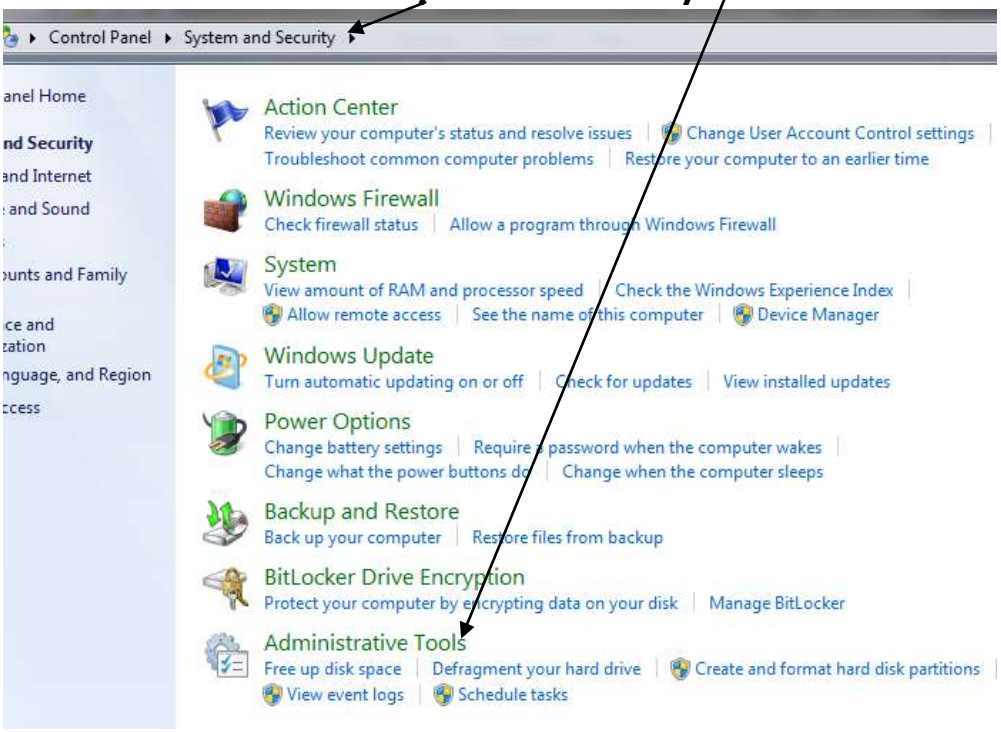


d) Simpan ulang dan Microsoft Office Access 2007 dapat ditutup.

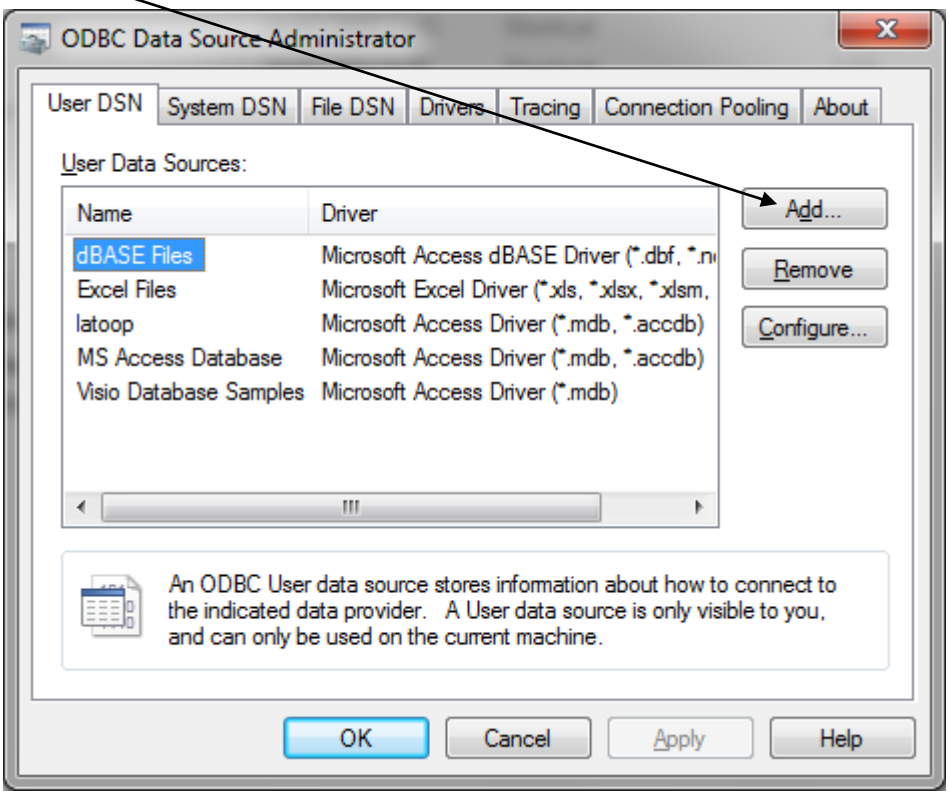
3. Membuat ODBC

Langkah-langkahnya sebagai berikut :

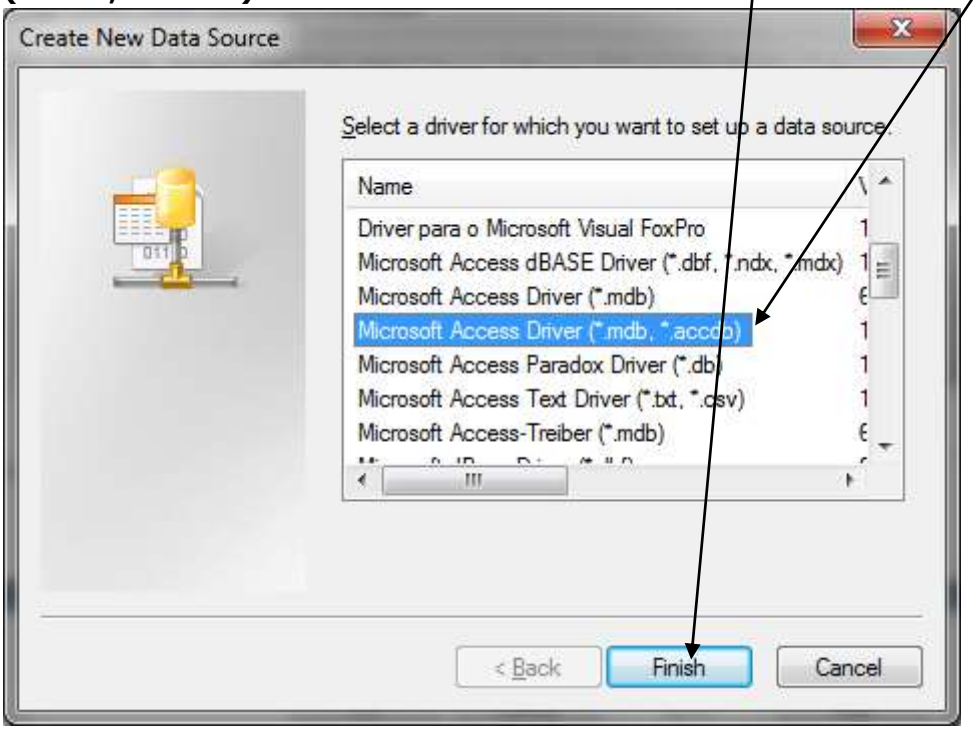
a) Buka **Control Panel**, cari/pilih **Administrative Tools**. Pada windows 7 administrative tools berada di **System and Security**.



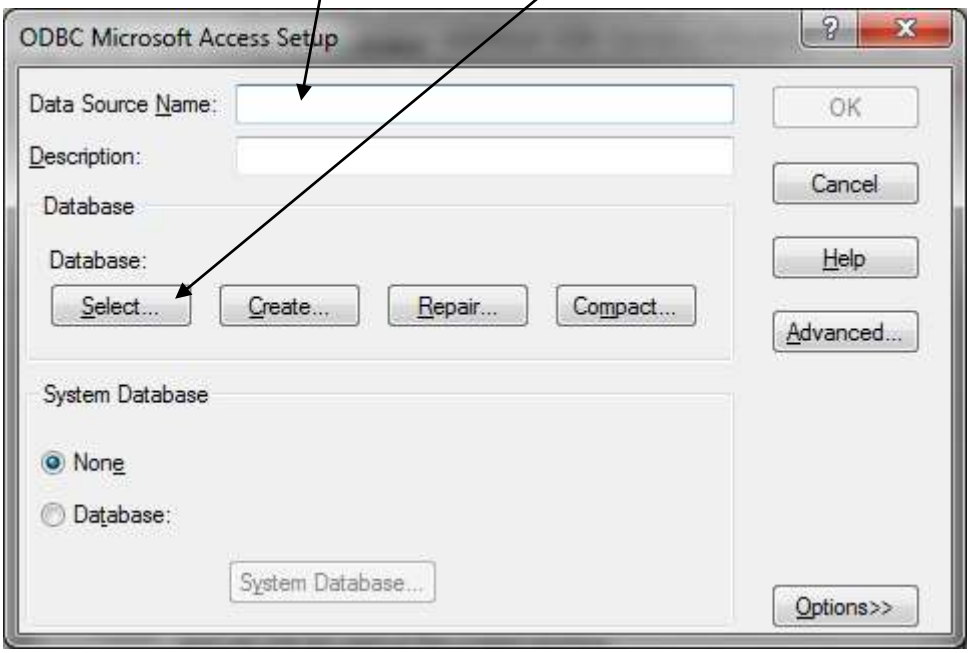
b) Pada **Administrative Tools**, buka **Data Source (ODBC)**. Lanjutkan klik tombol **Add**



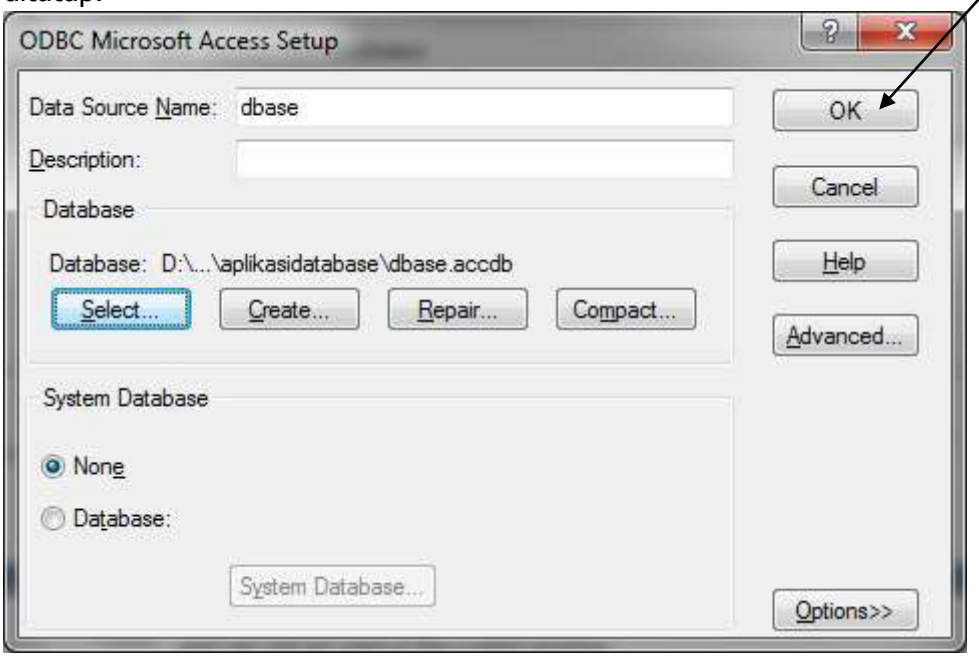
c) Pada kotak dialog **Create New Data Source**, pilih **Microsoft Access Driver (*.mdb, *.accdb)**. Kemudian silahkan klik tombol **Finish**.



d) Pada kotak dialog **ODBC Microsoft Access Setup**, tentukan nama Data Source Namenya (contoh : **dbase**). Klik tombol **select** untuk mencari database yang telah dibuat.



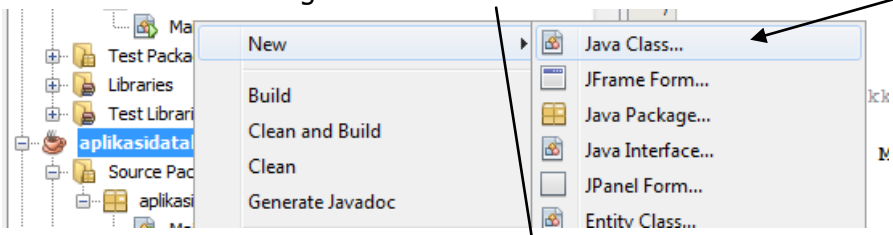
e) Tampilan akhir ODBC Microsoft Access Setup seperti berikut ini. Klik tombol **OK** dan klik **OK** pada **ODBC Data Source Administrator**, **control panel** dapat ditutup.



4. Membuat form dengan database

Langkah-langkahnya sebagai berikut :

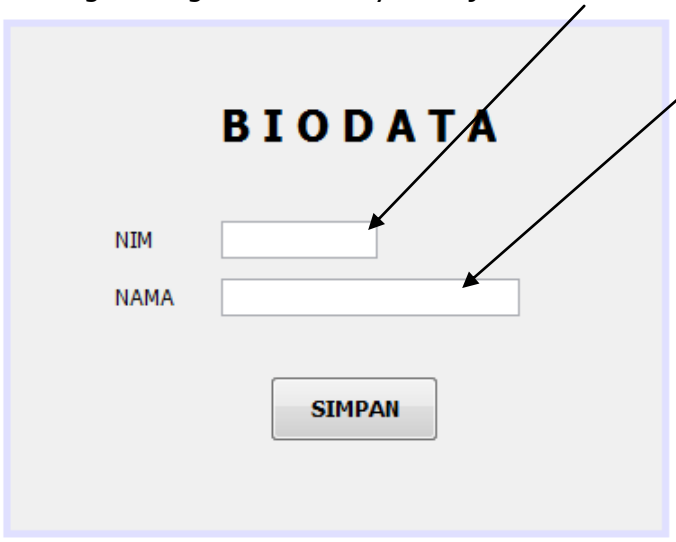
a) Pada project baru yang telah dibuat, tambahkan sebuah class baru (**Java Class**). Class tersebut beri dengan nama **Koneksi**.



b) Pada class **Koneksi**, tambahkan listing program sebagai berikut (**sintaks program yang ditambahkan terdapat dalam kotak segi empat**):

```
import java.sql.*;

public class Koneksi {
    public Koneksi () {}
    public Connection openKoneksi() throws SQLException
    {
        Connection con=null;
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:dbase","","");
            return con;
        }
        catch (SQLException se)
        {
            System.out.println("No Connection Open");
            return null;
        }
        catch (Exception e)
        {
            System.out.println("Cannot Open Connection");
            return null;
        }
    }
}
```

c) Buat sebuah form baru dan rancang tampilan form seperti berikut ini. Khusus untuk text field masing-masing ubah namanya menjadi **txtNIM** dan **txtNAMA**.



d) Klik tabulasi **Source**. Tambahkan perintah berikut :

```
import java.sql.*;
public class formdata extends javax.swing.JFrame {
```

e) Tambahkan listing program pada tombol simpan sebagai berikut :

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try
    {
        Koneksi k=new Koneksi();
        Connection cn=k.openKoneks();
        Statement st=cn.createStatement();
        String sql="insert into mhs ";
               sql+="values('"+txtNIM.getText()+"','"+txtNAMA.getText()+"');";
        st.executeUpdate(sql);
        st.close();
        cn.close();
        JOptionPane.showMessageDialog(null, "Data Berhasil Disimpan","SIMPAN",
                JOptionPane.INFORMATION_MESSAGE);
    }
    catch(SQLException sqe)
    {
        JOptionPane.showMessageDialog(null, "Gagal Simpan","Gagal Simpan",
                JOptionPane.WARNING_MESSAGE);
    }
}
```

f) Form silahkan di running.

RANCANGAN APLIKASI SISTEM INFORMASI BERORIENTASI OBJEK

Berikut adalah salah satu contoh rancangan aplikasi sistem informasi yang dapat menjadi panduan dalam merancang aplikasi sistem informasi yang lain. Aplikasi yang dibuat menggunakan pendekatan berorientasi objek. Tema sistem informasi yang digunakan dalam modul ini adalah mengenai Sistem Informasi Penjualan Tunai. Aplikasi ini menggunakan menu utama sebagai fasilitas integrasi antar modulnya, modul/form master, modul/form transaksi dan modul laporan.

Untuk rancangan keluaran dari sistem berupa laporan ataupun hasil transaksi menggunakan aplikasi ireport yang memang disediakan untuk membuat laporan dan sudah terkoneksi dengan netbeans. Dalam Modul ini ireport yang digunakan versi 5.01 (**ireport 5.01**). Sama dengan netbeans, aplikasi ini dapat didownload secara gratis (open source) di http://sourceforge.net/projects. Database yang digunakan adalah Microsoft Office Access 2007. Koneksi database dengan aplikasi menggunakan ODBC (Open akan Database Conectivity).

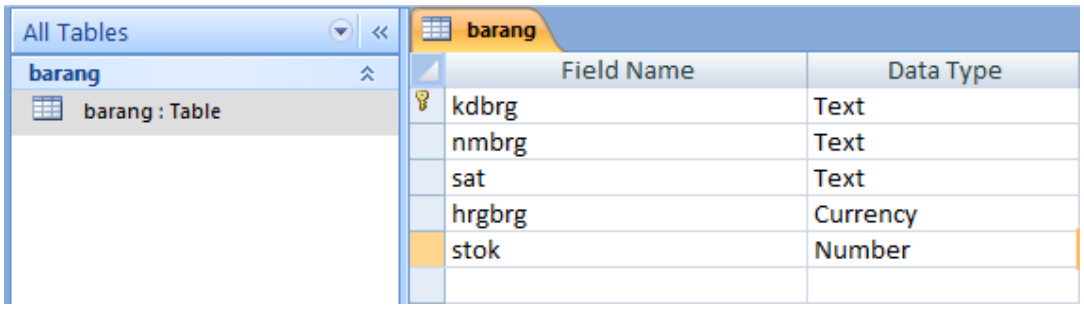
1. Buat Database

Buka Aplikasi Microsoft Office Acces 2007. Buat database dengan nama **dbjual.accdb**. Pada database tersebut buat beberapa tabel yang diperlukan yaitu **barang**, **pelanggan**, **pesanan**, **detailpesan** dan **nota** dengan spesifikasi sebagai berikut :

a. barang

No	Nama Field	Jenis	Lebar	Desimal	Keterangan
1	kdbrg	text	5	-	Primary Key, Kode Barang
2	nmbrg	text	20	-	Nama Barang
3	sat	text	10	-	Satuan Barang
4	hrgbrg	currency	6	-	Harga Barang
5	stok	number	3	-	Stok Barang

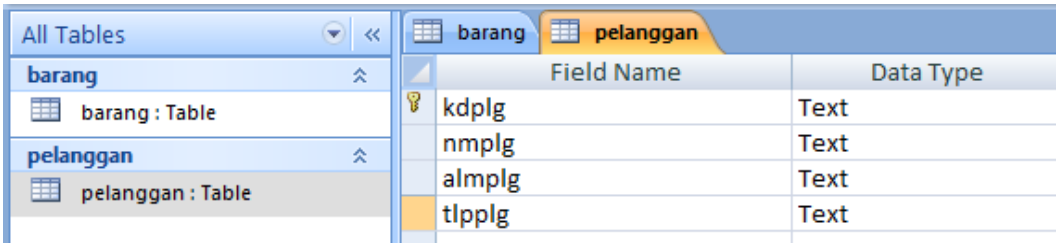
Contoh tampilan :



b. pelanggan

No	Nama Field	Jenis	Lebar	Desimal	Keterangan
1	kdplg	text	5	-	Primary Key, Kode Pelanggan
2	nmplg	text	20	-	Nama Pelanggan
3	almplg	text	100	-	Alamat Pelanggan
4	tlpplg	text	12	-	Telepon Pelanggan

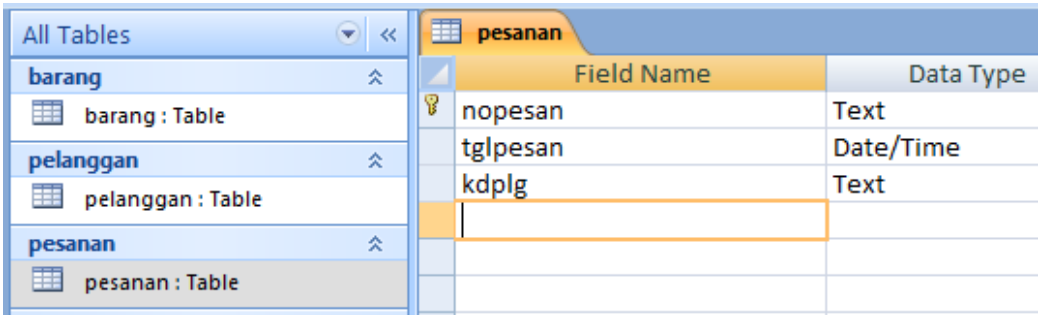
Contoh tampilan :



c. pesanan

No	Nama Field	Jenis	Lebar	Desimal	Keterangan
1	nopesan	text	5	-	Primary Key, Nomor Pesanan
2	tglpesan	date/time	8	-	Tanggal Pesanan
3	kdplg	text	5	-	Kode Pelanggan

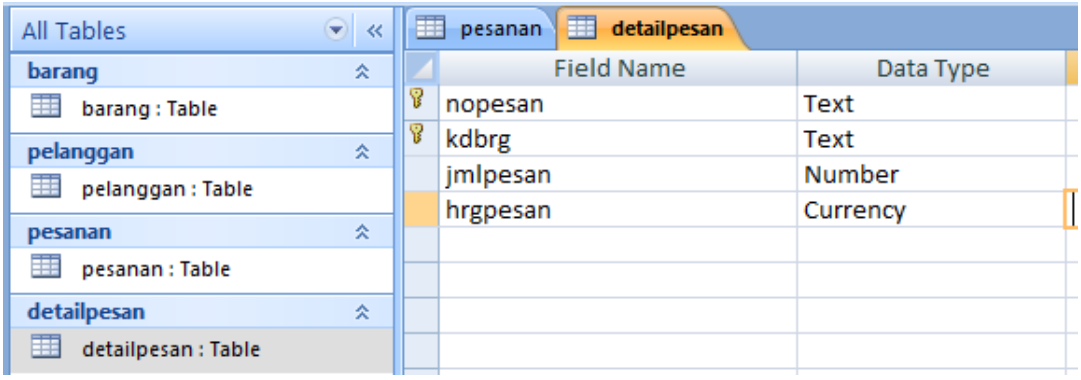
Contoh tampilan :



d. detailpesan

No	Nama Field	Jenis	Lebar	Desimal	Keterangan
1	nopesan	text	5	-	Primary Key, Nomor Pesanan
2	kdbrg	text	5	-	Primary Key, Kode Barang
3	jmlpesan	number	3	-	Jumlah Pesanan
4	hrgpesan	currency	6	-	Harga Pesanan

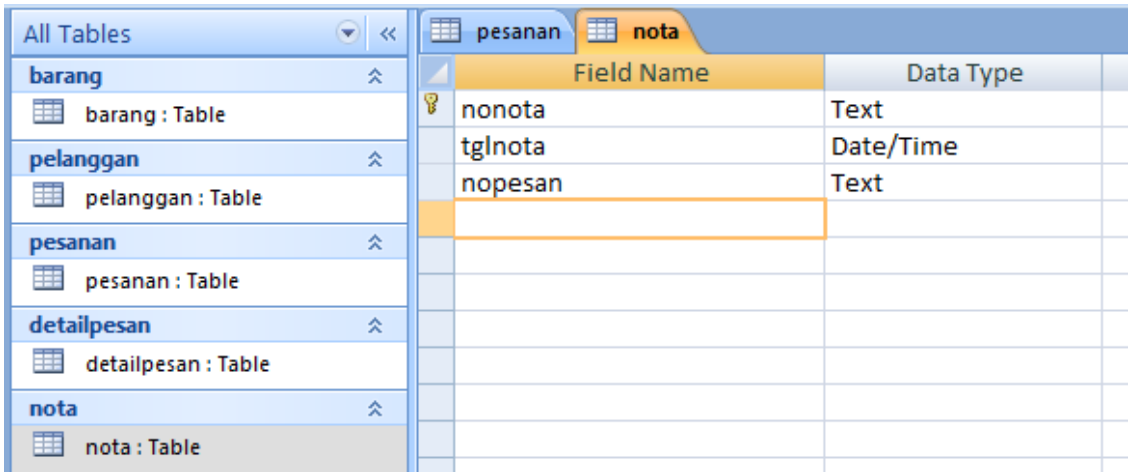
Contoh tampilan :



e. nota

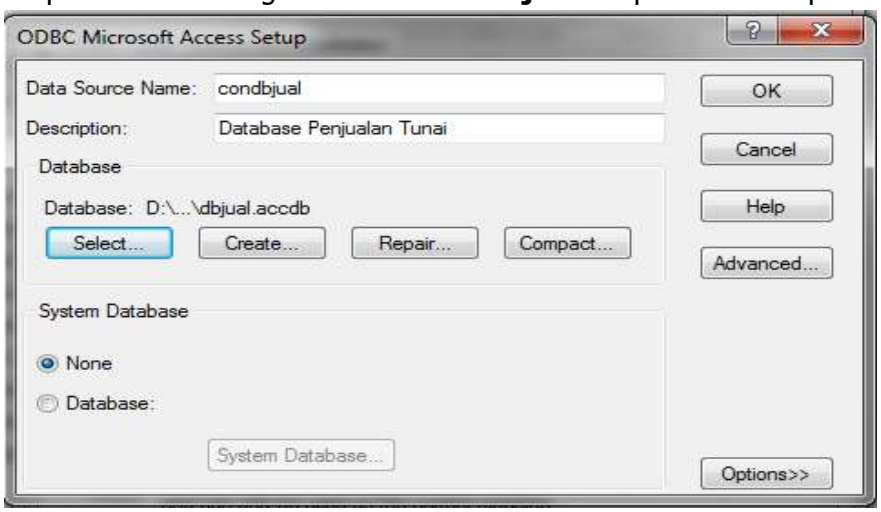
No	Nama Field	Jenis	Lebar	Desimal	Keterangan
1	nonota	text	5	-	Primary Key, Nomor Nota
2	tglnota	date/time	8	-	Tanggal Nota
3	nopesan	texte	5		Nomor Pesanan

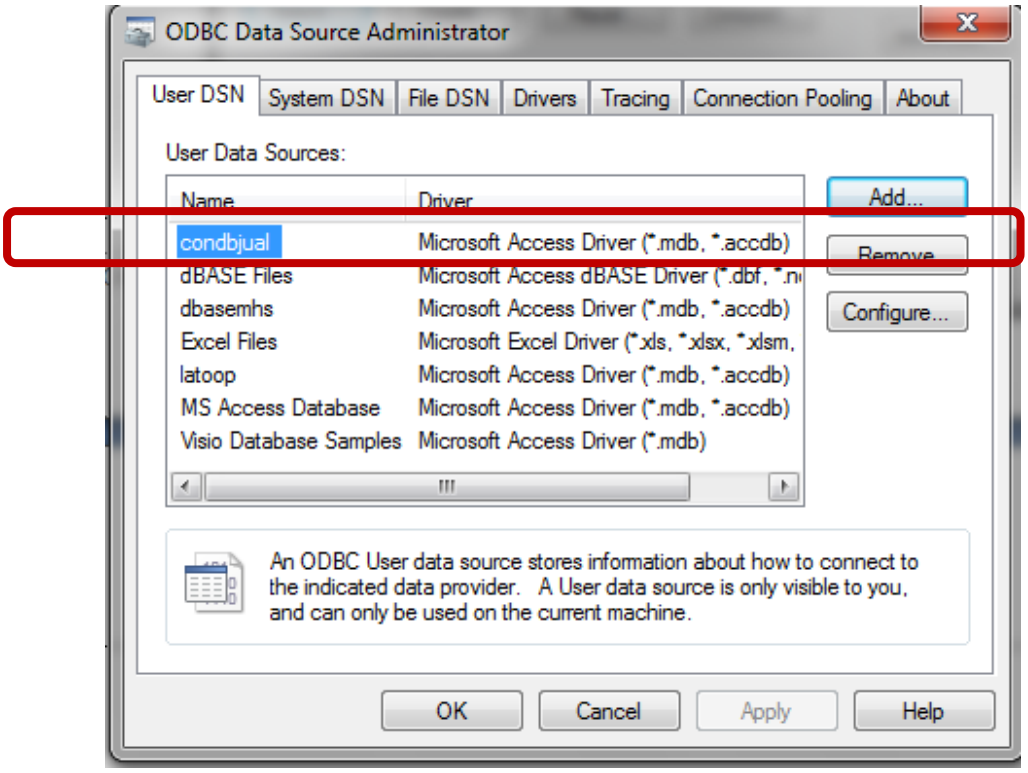
Contoh tampilan :



2. Koneksi pada ODBC

Buat koneksi pada ODBC dengan nama : **condbjual**. Seperti contoh pada gambar berikut :

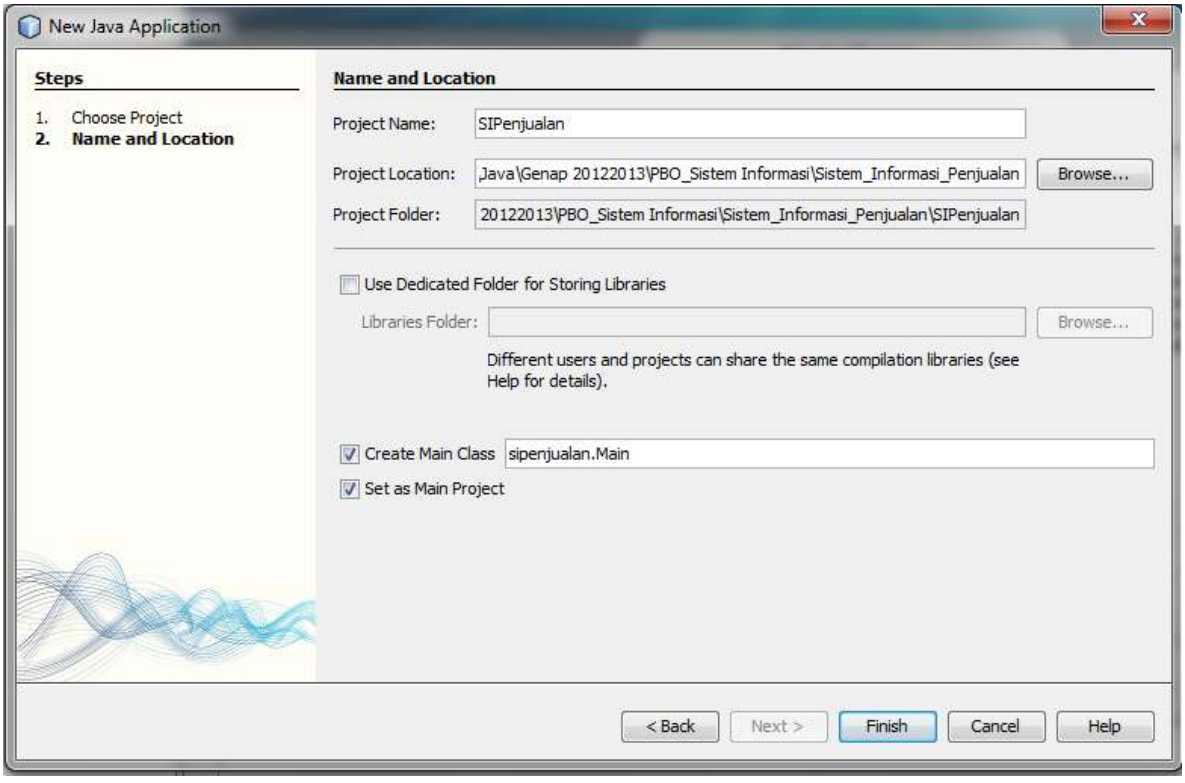


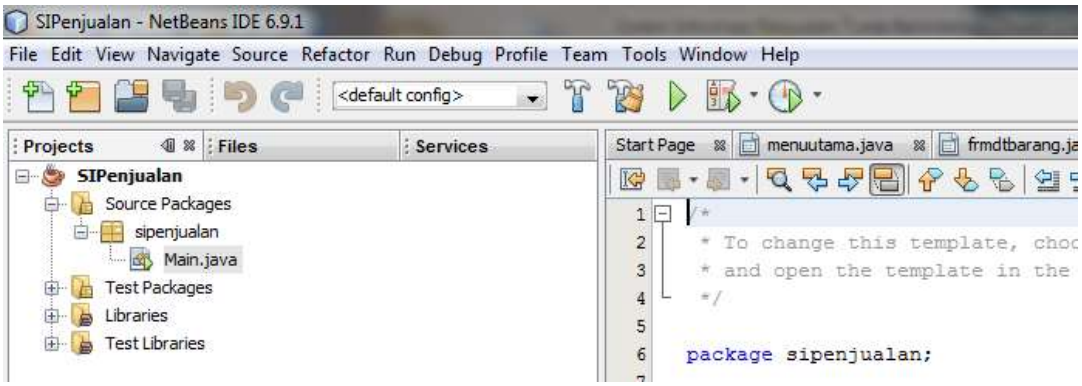


3. Buat Aplikasi pada Netbeans

a. Buat Project Baru

Buka aplikasi NetBeans dan buat sebuah project baru. Project baru tersebut beri dengan nama : **SIPenjualan.** Lokasi penyimpanan project silahkan ditentukan sesuai dengan keinginan.



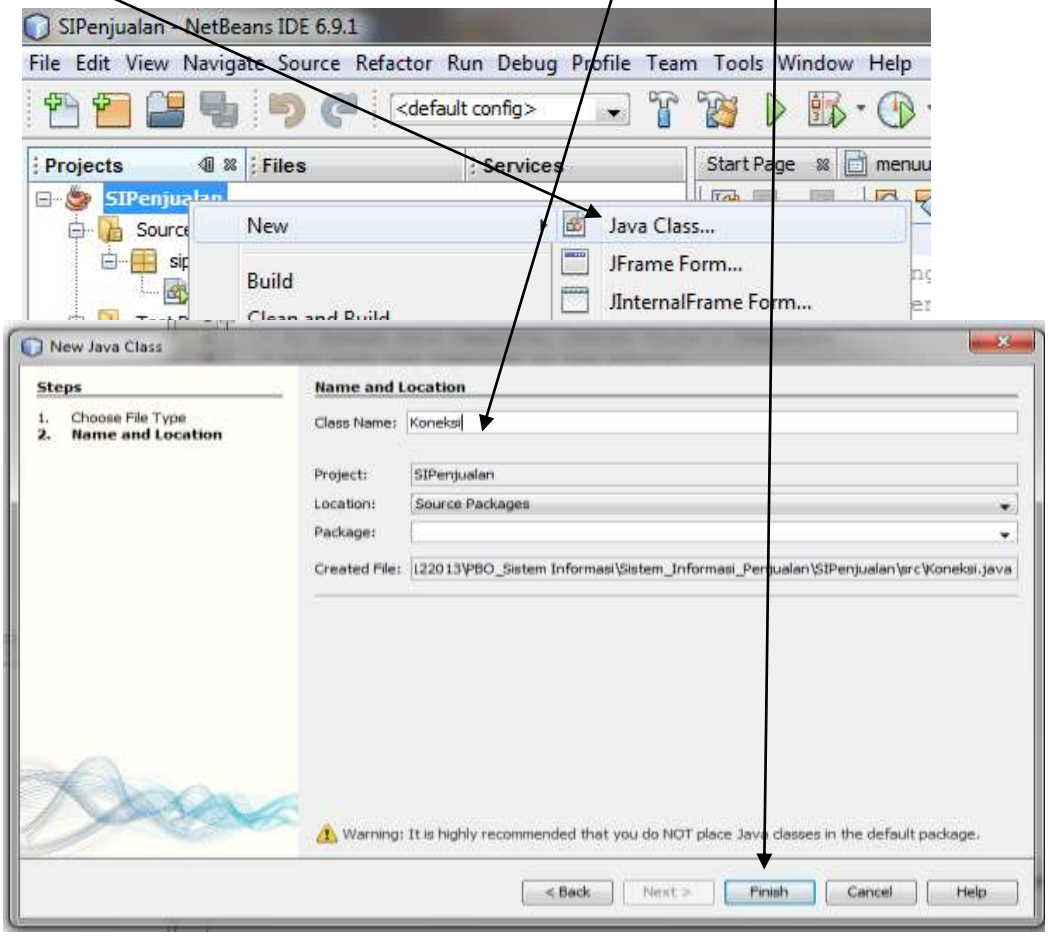


b. Buat Class Koneksi

Class koneksi ini berisi mengenai konfigurasi database yang akan digunakan/ hubungkan pada aplikasi yang akan dibuat pada NetBeans.

Adapun tujuan class ini dibuat adalah untuk memudahkan mengoneksikan antar modul dengan database. Artinya setiap modul tidak perlu membuat ulang bagaimana cara menghubungkan dengan database, tetapi cukup memanggil class koneksi ini saja. Dengan class koneksi ini juga secara tidak langsung konsep object oriented programmingnya terbentuk. Langkah-langkahnya sebagai berikut :

1) Pada project baru yang telah dibuat, tambahkan sebuah class baru (**Java Class**). Class tersebut beri dengan nama **Koneksi.** Klik **Finish.** Klik kanan pada project **SIPenjualan**.



2) Pada editor class **Koneksi**, tambahkan listing program sebagai berikut (**listing program yang ditambahkan terdapat dalam kotak segi empat**) :

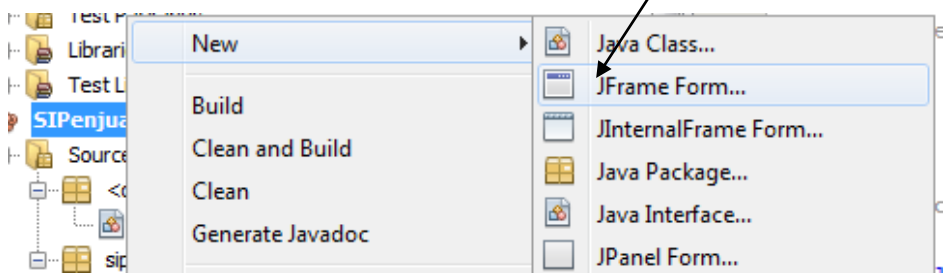
```
import java.sql.*;

public class Koneksi {
    public Koneksi() {}
    public Connection openKoneksi() throws SQLException
    {
        Connection con = null;
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:condbjual","","");
            return con;
        }
        catch (SQLException se)
        {
            System.out.print("No Connection Open");
            return null;
        }
        catch (Exception e)
        {
            System.out.println("Cannot Open Connection");
            return null;
        }
    }
}
```

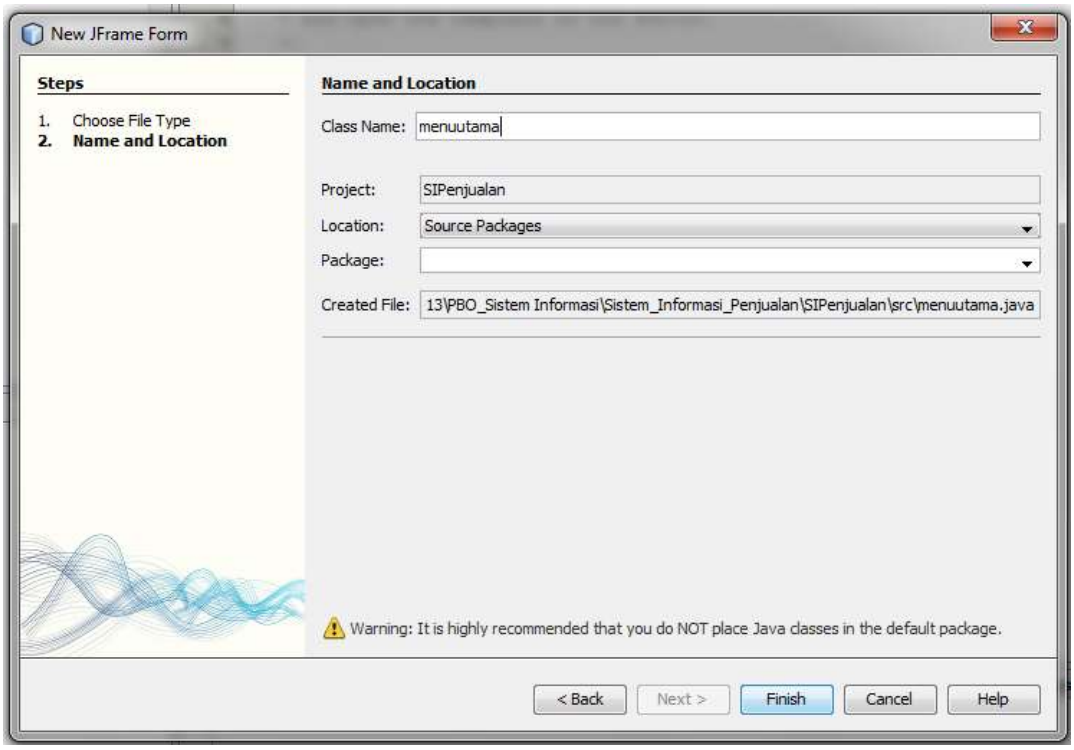
3) Pastikan project sudah disimpan/save ()

c. Buat Menu Utama

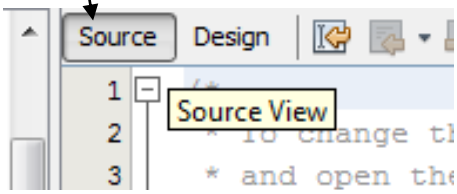
Menu bermanfaat untuk mengintegrasikan beberapa modul program/form menjadi satu bagian. Dengan menggunakan menu, beberapa modul yang dimiliki tidak perlu di running satu persatu. Cukup dengan menjalankan menu dan secara otomatis modul yang telah terintegrasi dengan menu tersebut akan dijalankan. Pada java beberapa class yang berhubungan dengan menu adalah class JMenuBar, JMenu, JMenuItem dan JDesktopPane. Langkah-langkahnya sebagai berikut :

1) Pada project tersebut, buat sebuah form baru (**JFrame Form**), beri dengan nama **menuutama.java**. klik **Finish**.



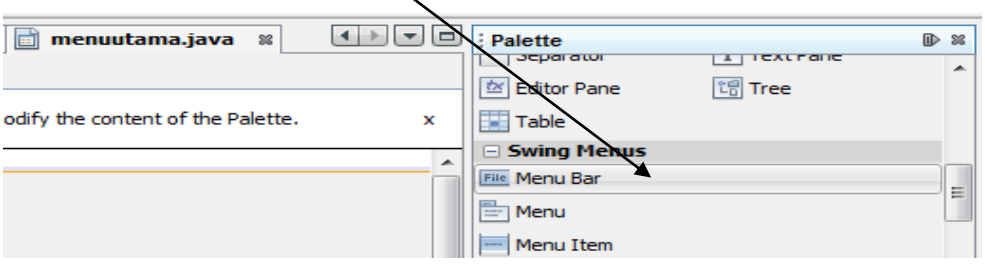


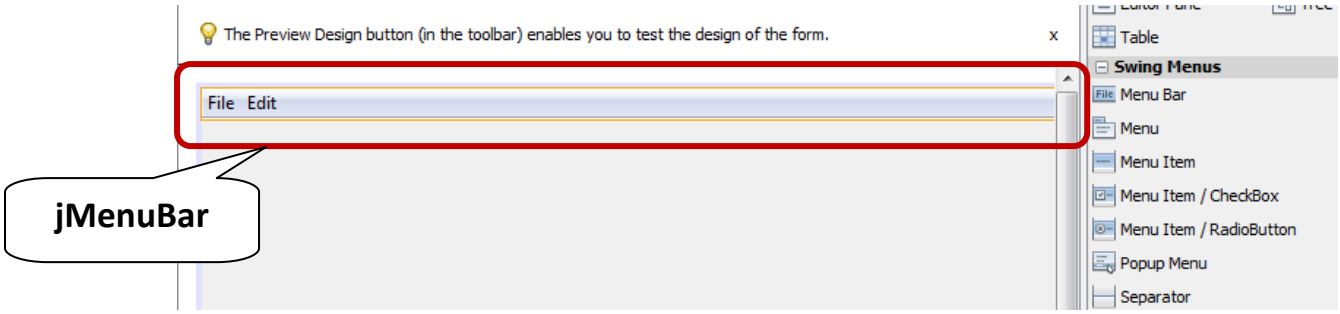
2) Klik tabulasi **Source,** tambahkan perintah berikut untuk memberikan judul dan lokasi form pada layar (**listing program yang ditambahkan terdapat dalam kotak segi empat**).



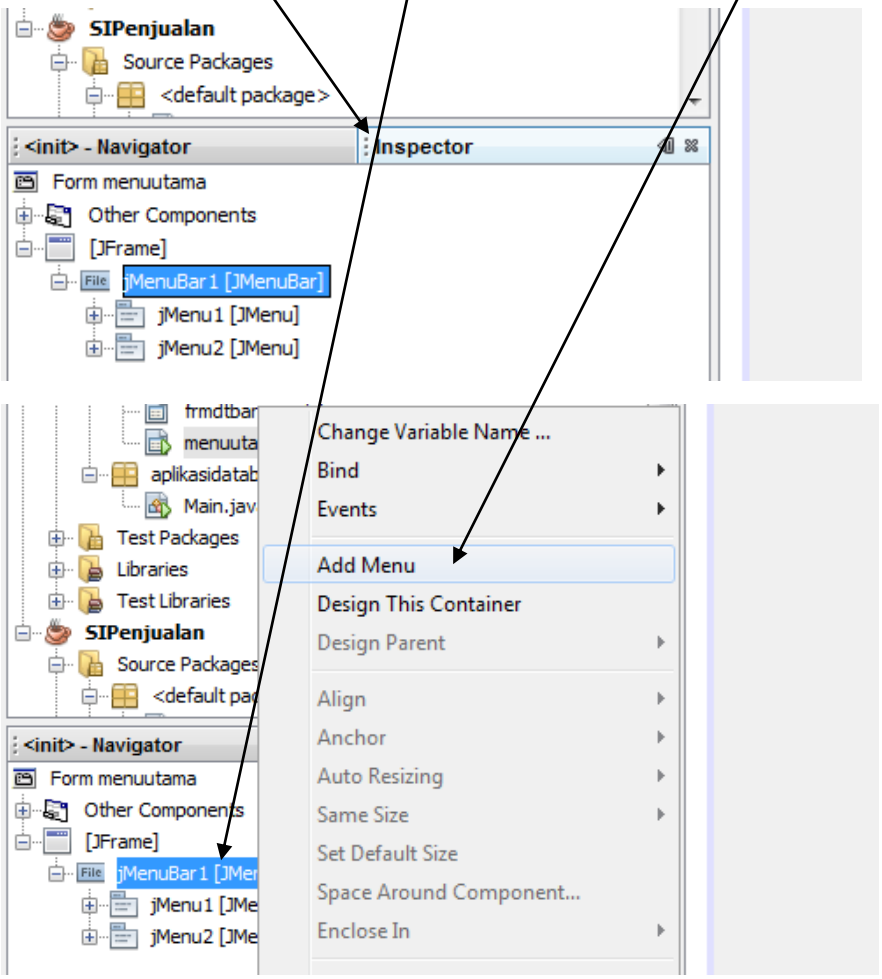

```
*/
public class menuutama extends javax.swing.JFrame {

    /** Creates new form menuutama */
    public menuutama() {
        super("Sistem Informasi Penjualan Tunai");
        setLocation(100,100);
        initComponents();
    }
```

3) Klik tabulasi **Design**, ubah ukuran form menjadi sedikit lebih lebar, karena akan digunakan sebagai menu utama.
4) Masukkan komponen **JMenuBar** dari **pallet** kedalam form.

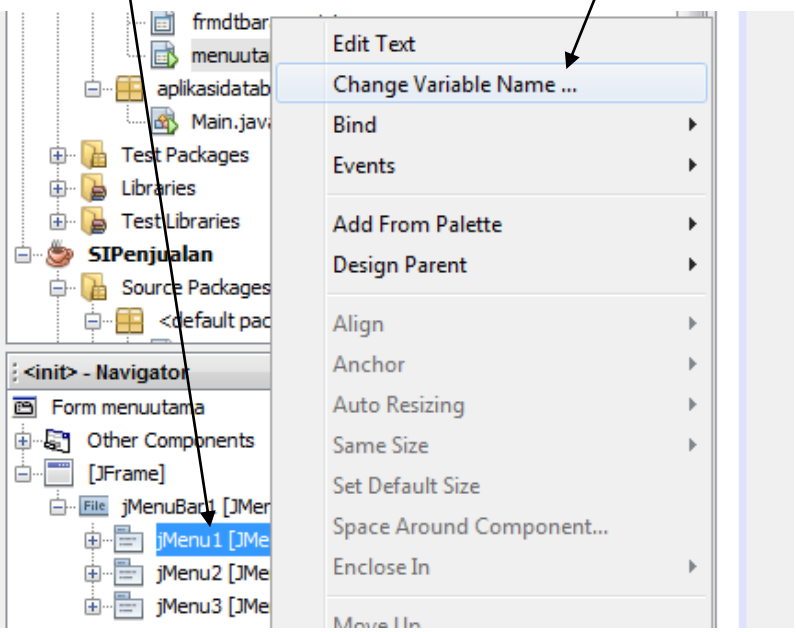




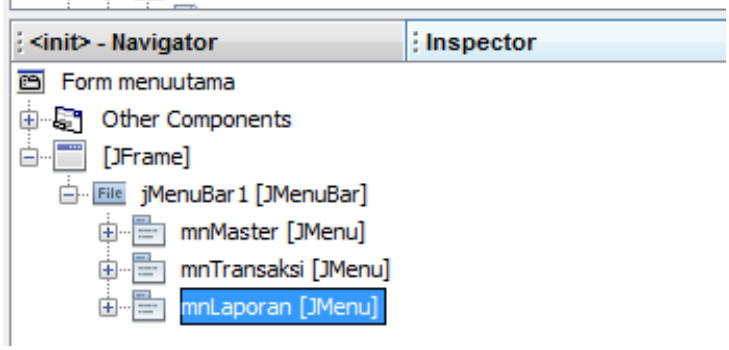
5) Pada Jendela **Inspector**, tambahkan sebuah menu baru (**Add Menu**) dengan cara mengklik kanan pada **JMenuBar** yang sudah disisipkan ke Form.



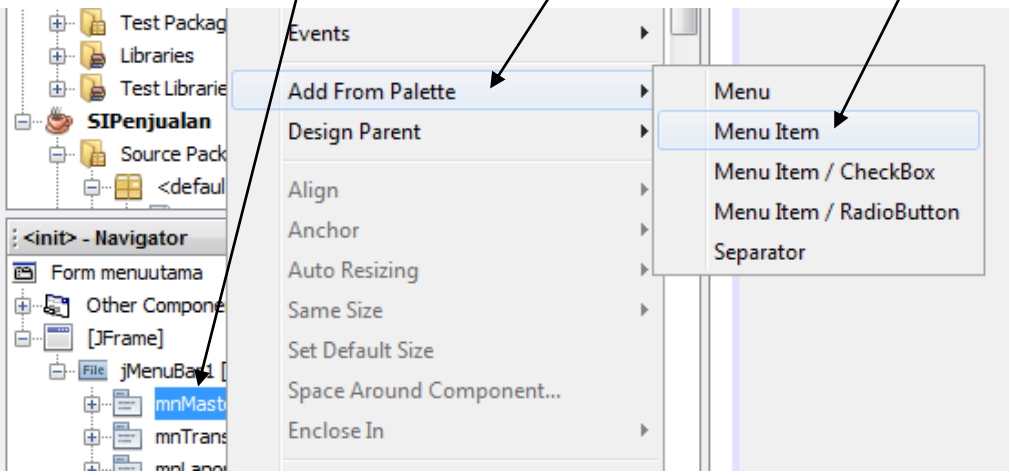
6) Masih di Jendela **Inspector**, ubah nama masing-masing menu dengan nama **mnMaster**, **mnTransaksi**, **mnLaporan** dengan cara mengklik kanan pada masing-masing **JMenu** tersebut dan pilih **Change Variable Name**.



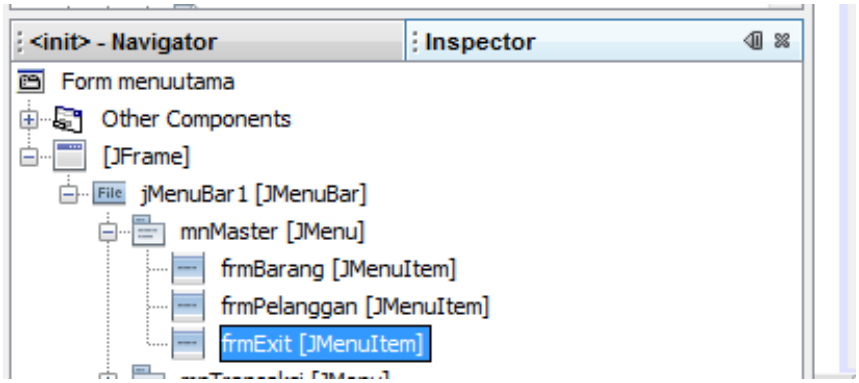
Tampilan akhir :



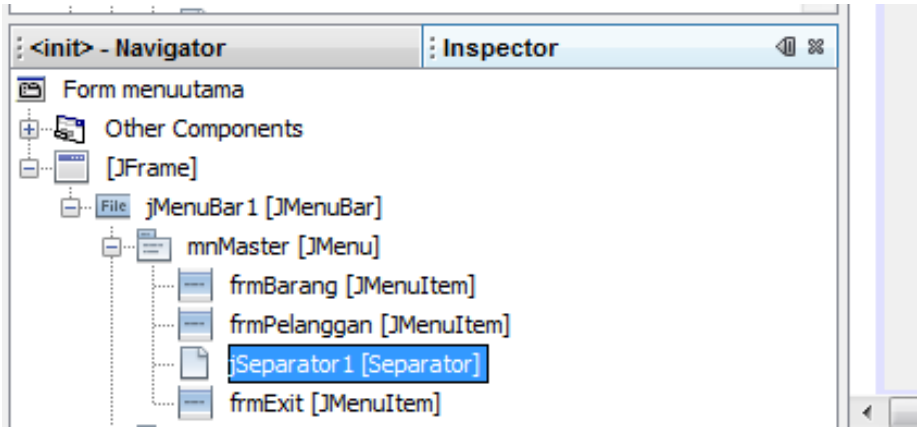
7) Masing-masing menu tersebut mempunyai sub menu. Untuk membuat sub menu klik kanan pada **menu** dan pilih **Add From Palette** dan pilih **Menu Item**. Buat sub menu untuk setiap menu dengan sebagai berikut :



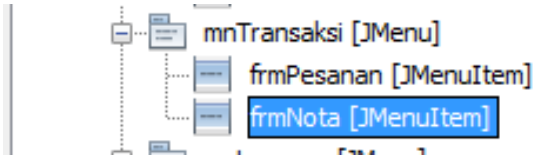
- **mnMaster** dengan sub menu sebagai berikut :



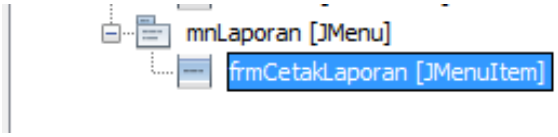
Tambahkan **separator** untuk memberikan jarak pada **frmExit (klik kanan mnMaster->Add From Palette->)**. **Separator** dapat dinaikkan atau diturunkan, sehingga tampilan akhir sebagai berikut :



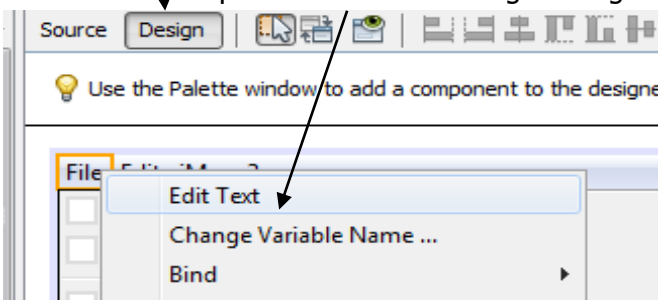
- **mnTransaksi** dengan sub menu sebagai berikut :



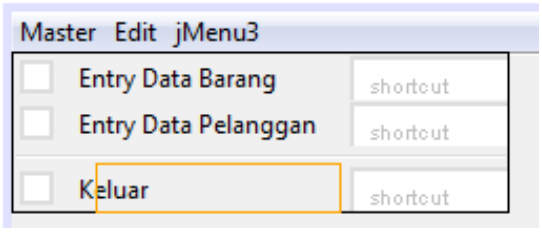
- **mnTransaksi** dengan sub menu sebagai berikut :



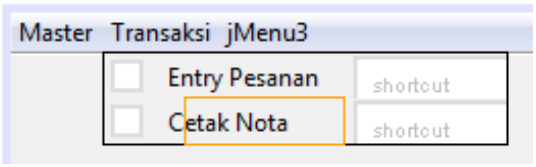
8) Kembali ke tabulasi **Design**, ubah nama masing-masing menu dan sub menu dengan cara klik kanan dan pilih **Edit Text**. Masing-masing sebagai berikut :



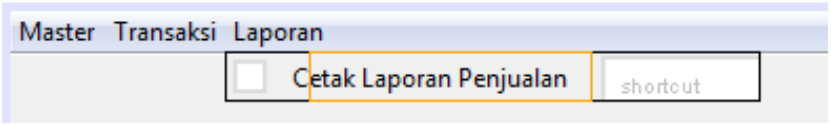
- Menu **Master** dan sub menunya sebagai berikut :



- Menu **Transaksi** dan sub menunya sebagai berikut :



- Menu **Laporan** dan sub menunya sebagai berikut :

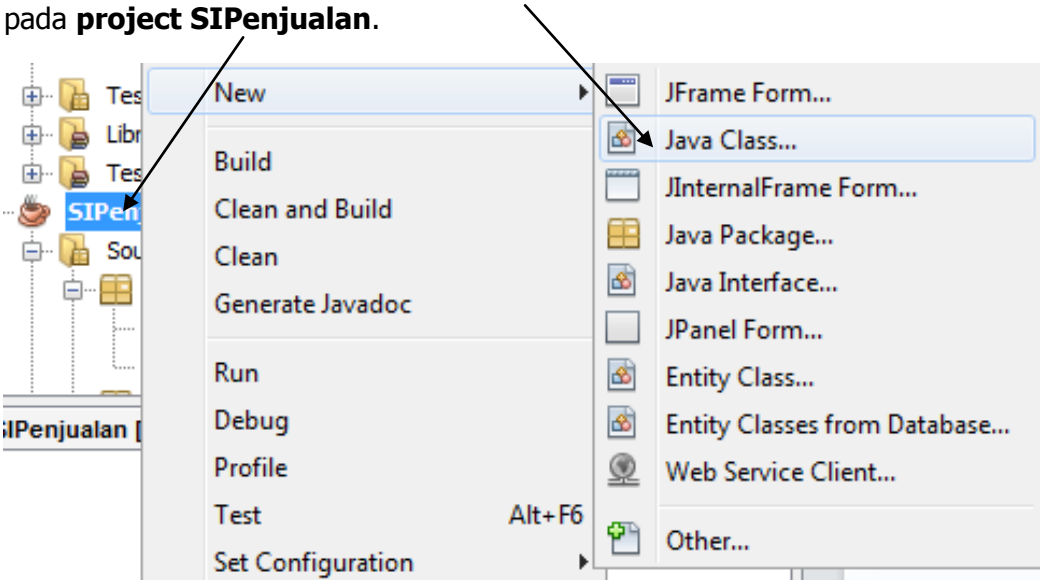


9) Pastikan Project disimpan ulang/save () dan **untuk sementara** form menu utama sudah selesai dibuat. **Mohon diingat**, **untuk sementara** form menu utama sudah selesai dibuat.

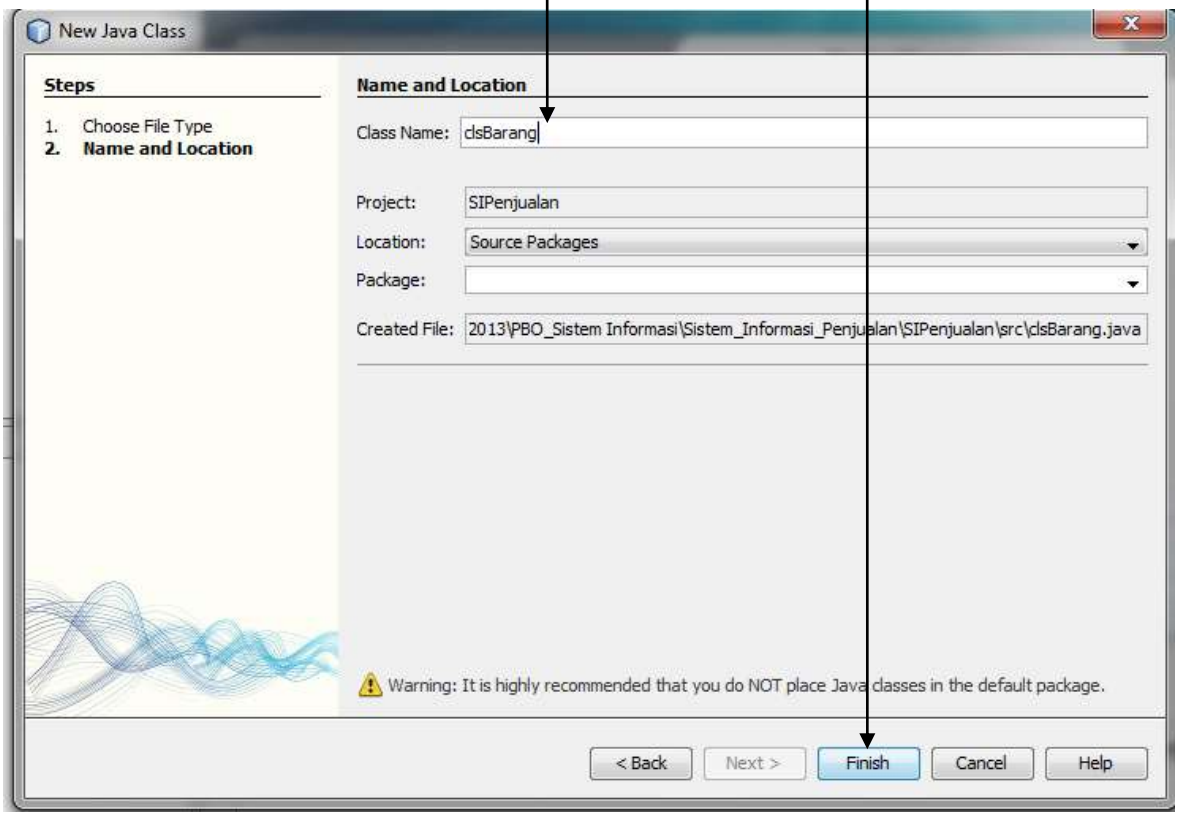
d. Buat class barang

Langkah-langkahnya sebagai berikut :

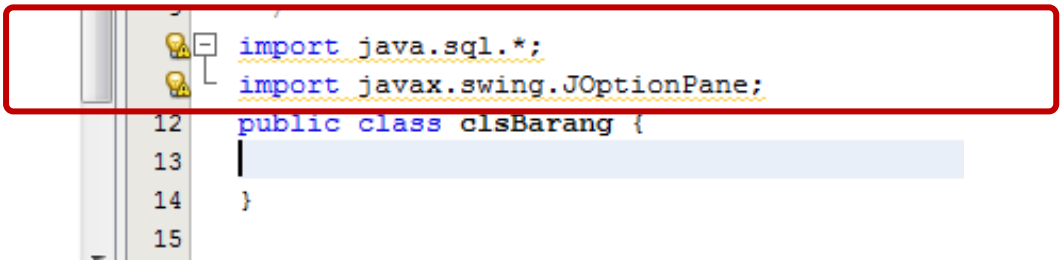
1) Tambahkan sebuah class baru (**Java Class**) dengan cara mengklik kanan pada **project SIPenjualan**.



2) Class tersebut beri dengan nama **clsBarang**, kemudian klik **finish**.



3) Pada Class tersebut, tambahkan sintaks untuk mengimport beberapa library yang diperlukan (**Listing program yang ditambahkan terdapat pada kotak segi empat**).



4) Didalam class **clsBarang** tersebut, tambahkan perintah untuk membuat variabel/atribut dan methode yang diperlukan untuk menentukan atau mengirimkan nilai dari atau ke variabel tersebut (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
public class clsBarang {
  protected String kode, nama, sat;
  protected int hrg, stok, flag;

  public void setKode (String Kd)
   { kode = Kd;}
  public void setNama (String Nm)
   { nama = Nm;}
  public void setSatuan(String St)
   { sat = St;}
  public void setHarga(int Hr)
   { hrg = Hr;}
  public void setStok(int Sk)
   { stok = Sk;}
  public String getKode()
   { return(kode);}
  public String getNama()
   { return(nama);}
  public String getSatuan()
   { return(sat);}
  public int getHarga()
   { return(hrg);}
  public int getStok()
   { return(stok);}
  public void setFlag(int F)
   { flag = F;}
  public int getFlag()
   { return(flag);}
}
```

5) Didalam class clsBarang tersebut, tambahkan methode **simpan** yang berfungsi untuk menyimpan data ke database. Methode ditambahkan dibawah metode **getFlag()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
public int getFlag()
 { return(flag);}

public void simpan()
 {
    try
    {
        Koneksi k=new Koneksi();
        Connection cn=k.openKoneksi();
        Statement st=cn.createStatement();
        String sql="insert into barang values(";
               sql+="'"+getKode()+"','"+getNama()+"',";
               sql+="'"+getSatuan()+"','"+getHarga()+"',";
               sql+="'"+getStok()+"');";

            st.executeUpdate(sql);
            setFlag(1);
            st.close();
            cn.close();
            JOptionPane.showMessageDialog(null, "Data Berhasil disimpan",
                    "SIMPAN",JOptionPane.INFORMATION_MESSAGE);

    }
    catch (SQLException sqe)
    {
        JOptionPane.showMessageDialog(null, "Gagal Simpan","Gagal Simpan",
                JOptionPane.WARNING_MESSAGE);
    }
 }
}
```

6) Didalam class clsBarang tersebut, tambahkan methode **ubah** yang berfungsi untuk mengubah data di database. Methode ditambahkan dibawah metode **simpan()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch (SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Gagal Simpan","Gagal Simpan",
                    JOptionPane.WARNING_MESSAGE);
        }
    }

    public void ubah()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="update barang set ";
                   sql+="nmbrg='"+getNama()+"',";
                   sql+="sat='"+getSatuan()+"',";
                   sql+="hrgbrg='"+getHarga()+"',";
                   sql+="stok='"+getStok()+"'";
                   sql+="where kdbrg='"+getKode()+"'";
            st.executeUpdate(sql);
            setFlag(2);
            st.close();
            cn.close();
            JOptionPane.showMessageDialog(null, "Data Berhasil diUbah",
                    "UBAH",JOptionPane.INFORMATION_MESSAGE);
        }
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data Gagal Ubah",
                    "Gagal Ubah",JOptionPane.WARNING_MESSAGE);
        }
    }
}
```

Perhatikan : antara set dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh :
set” (SALAH),
set ” (BETUL).

7) Didalam class clsBarang tersebut, tambahkan methode **hapus** yang berfungsi untuk menghapus data di database. Methode ditambahkan dibawah metode **ubah()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data Gagal Ubah",
                    "Gagal Ubah",JOptionPane.WARNING_MESSAGE);
        }
    }

    public void hapus()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="delete from barang ";
                   sql+="where kdbrg='"+getKode()+"'";

            st.executeUpdate(sql);
            setFlag(3);
            st.close();
            cn.close();
            JOptionPane.showMessageDialog(null, "Data Berhasil di Hapus",
                    "HAPUS",JOptionPane.INFORMATION_MESSAGE);

        }
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data GAGAL diHapus",
                    "Gagal Hapus",JOptionPane.WARNING_MESSAGE);
        }
    }
}
```

Perhatikan : antara barang dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!.
contoh :
barang” (SALAH),
barang ” (BETUL).

8) Didalam class clsBarang tersebut, tambahkan methode **tampil** yang berfungsi untuk menampilkan data dari database. Methode ditambahkan dibawah metode **hapus()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data GAGAL diHapus",
                    "Gagal Hapus",JOptionPane.WARNING_MESSAGE);
        }
    }

    public void tampil()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select * from barang ";
                   sql+="where kdbrg='"+getKode()+"'";
            ResultSet rs=st.executeQuery(sql);

            if(rs.next())
            {
                setFlag(4);
                setKode(rs.getString("kdbrg"));
                setNama(rs.getString("nmbrg"));
                setSatuan(rs.getString("sat"));
                setHarga(rs.getInt("hrgbrg"));
                setStok(rs.getInt("stok"));
                st.close();
                rs.close();
            }
        }
        catch(SQLException sqe)
        {}
    }
}
```

Perhatikan : antara barang dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!.
contoh :
barang” (SALAH),
barang ” (BETUL).

9) Didalam class clsBarang tersebut, tambahkan methode **autoKode** yang berfungsi untuk membuat kode barang secara otomatis. Methode ditambahkan dibawah metode **tampil()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        {}
    }

    public void autoKode()
    {
        try
        {
            int hit=0;

            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select count(kdbrg)from barang";
            ResultSet rs=st.executeQuery(sql);

            if(rs.next())
            {
                if(Integer.parseInt(rs.getString(1))==0)
                {
                  setKode("B0001");
                  st.close();
                  rs.close();
                }
                else
                {
                    sql="select Max(mid(kdbrg,2,4))from barang";
                    rs=st.executeQuery(sql);
                    rs.next();
                    hit = (Integer.parseInt(rs.getString(1)))+1;
                     if (hit<10)
                     { setKode("B000"+hit);}
                     else if (hit<100)
                     { setKode("B00"+hit);}
                     else if (hit<1000)
                     { setKode("B0"+hit);}
                     else
                     { setKode("B"+hit);}
                    st.close();
                    rs.close();
                 }
            }
        }
        catch(SQLException sqe)
        { }
    }
}
```

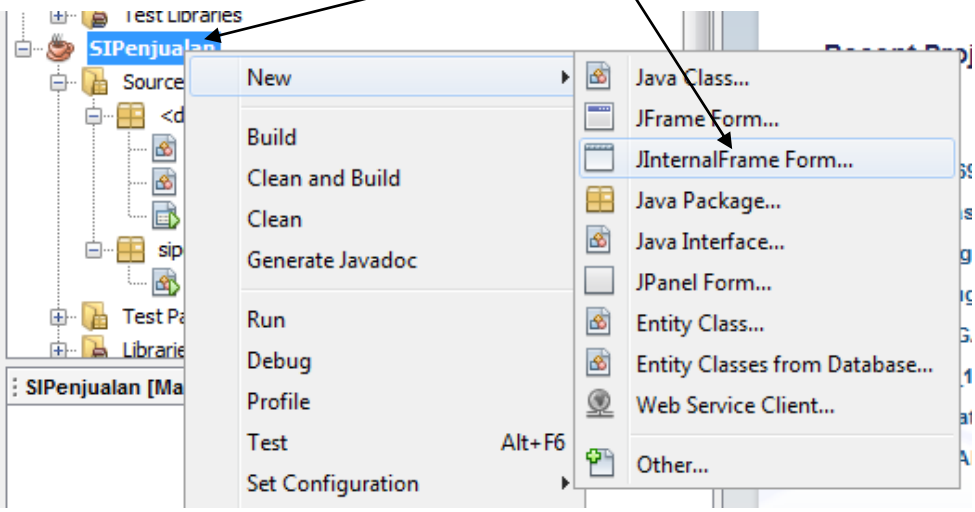
10) Secara keseluruhan class clsBarang sudah selesai dibuat. Project dapat disimpan ulang/disave ().

e. Buat Form Entry Data Barang

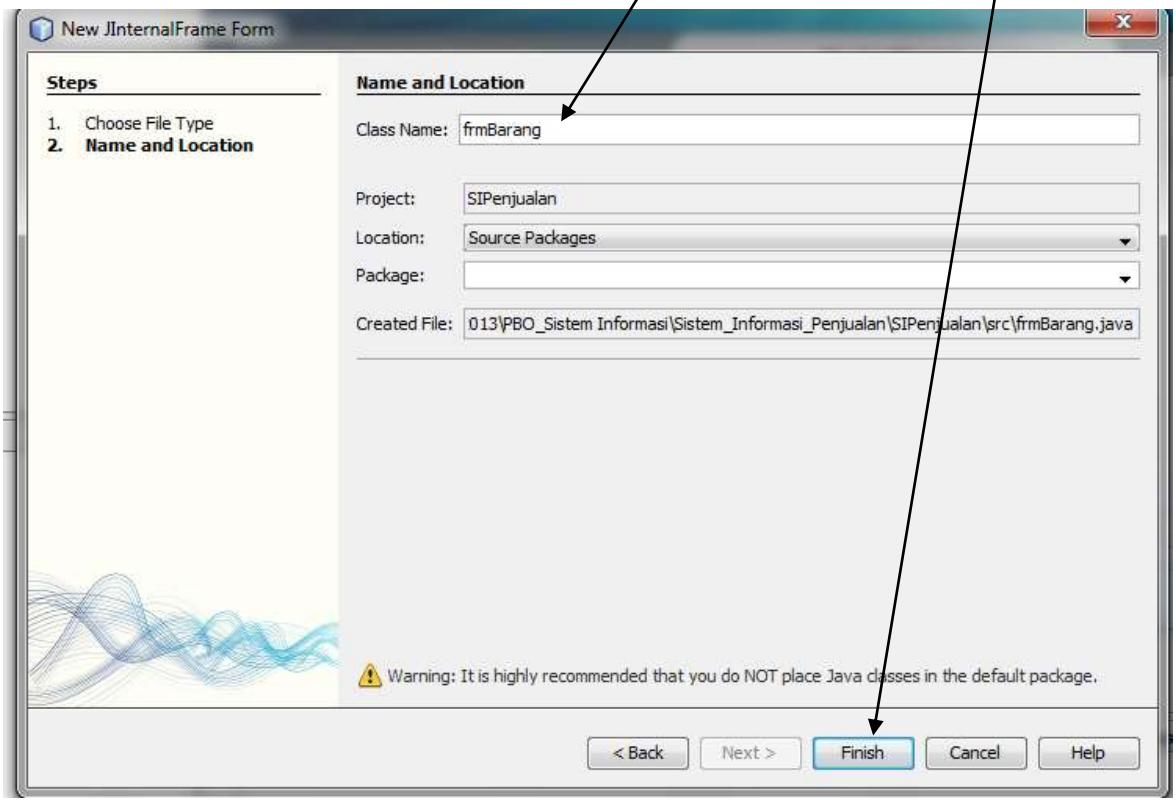
Dikarenakan menggunakan menu utama sebagai integrasinya, maka form yang akan dibuat JInternalFrame, sehingga tanpa melalui menu utama, maka form tidak akan bisa dijalankan.

Langkah-langkahnya sebagai berikut :

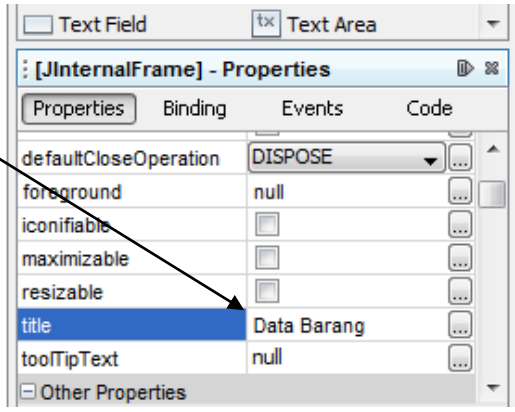
1) Tambahkan sebuah form baru (**JInternalFrame Form**), dengan cara mengklik kanan pada project **SIPenjualan**.



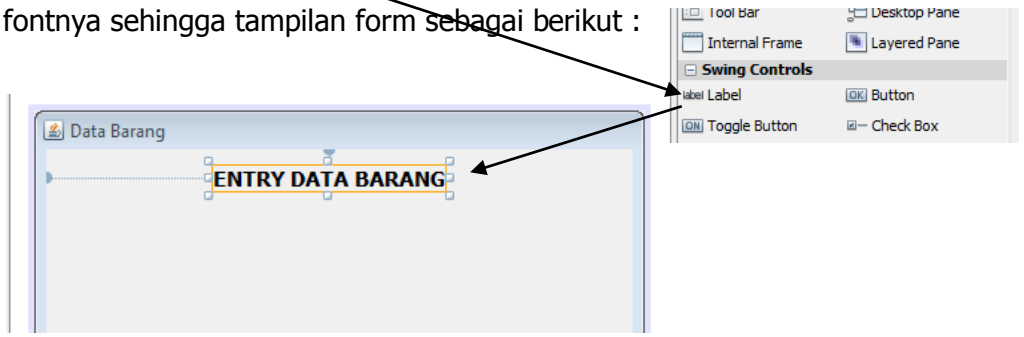
2) Form baru tersebut beri dengan nama **frmBarang**, kemudian klik **Finish**.



3) Beri judul form tersebut pada kolom yang sudah disediakan dengan nama : **Data Barang**.



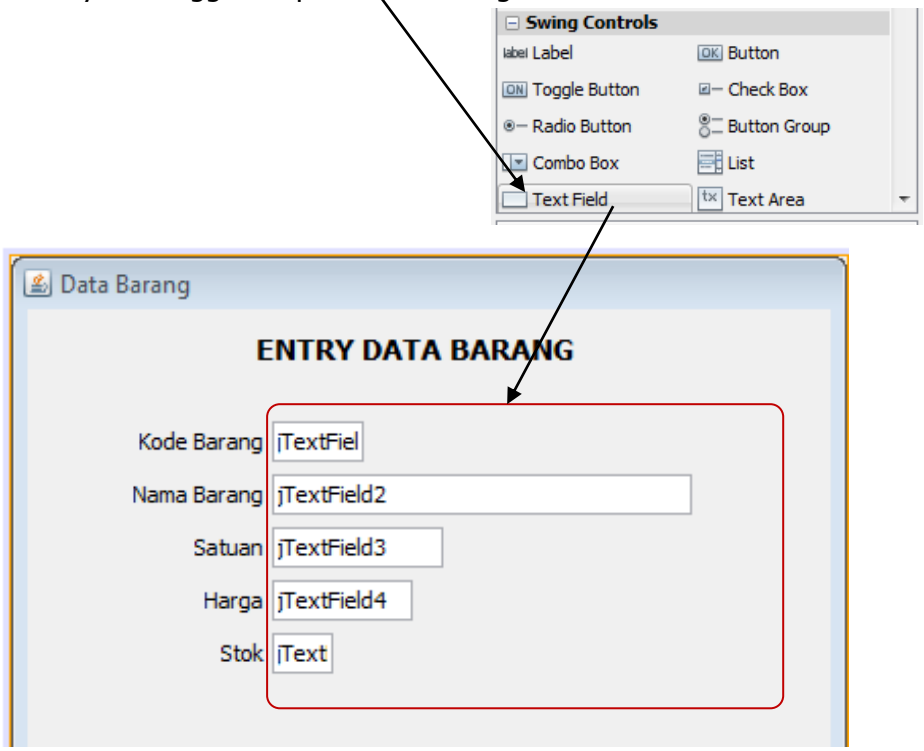
4) Tambahkan komponen **label** dari **palette swing controls**, atur huruf dan fontnya sehingga tampilan form sebagai berikut :



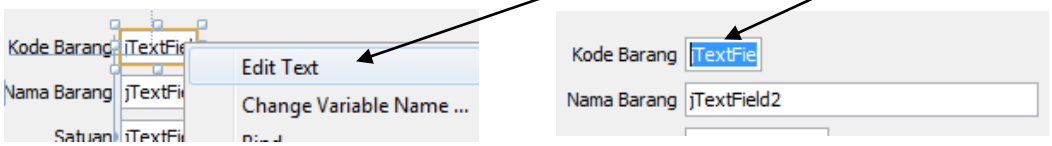
Klik kanan pada komponen **label** tersebut dan pilih **Edit Text** untuk mengubah tulisan.

5) Tambahkan komponen **Label** dari **palette swing controls**, atur huruf dan fontnya sehingga tampilan form sebagai berikut :

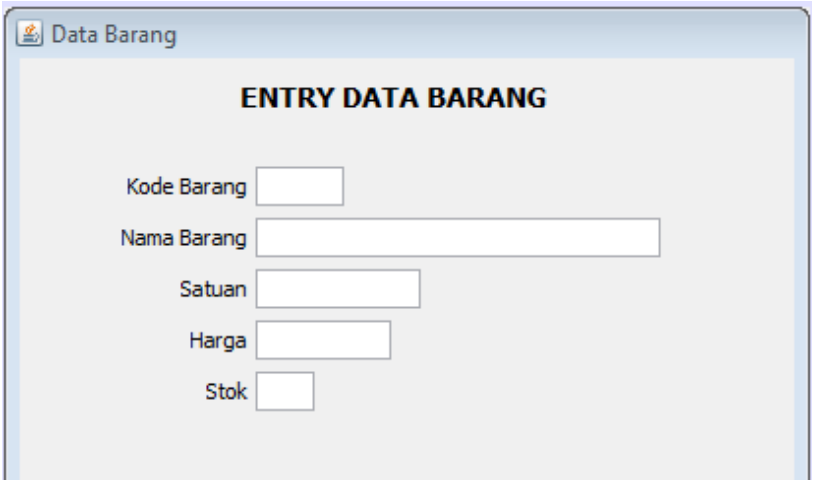
6) Tambahkan komponen **Text Field** dari **palette swing controls,** atur ukuran komponennya sehingga tampilan form sebagai berikut :



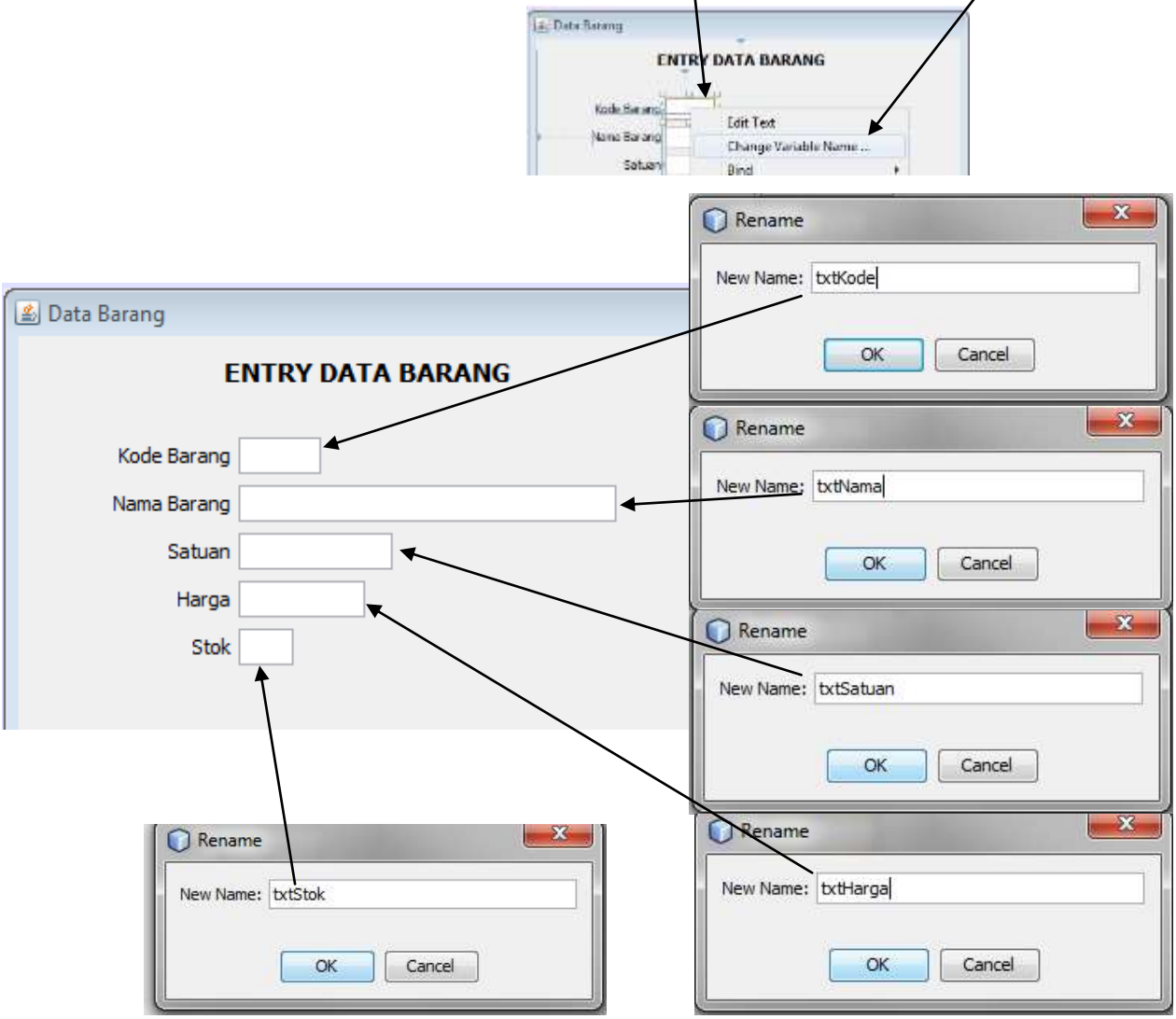
7) Hapus tulisan pada jTextField tersebut dengan cara mengklik kanan pada masing-masing-masing jTextField, pilih **Edit Text** dan di**hapus text**nya.



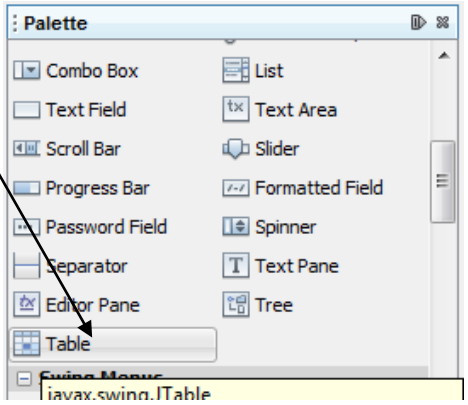
Tampilan sebagai berikut :

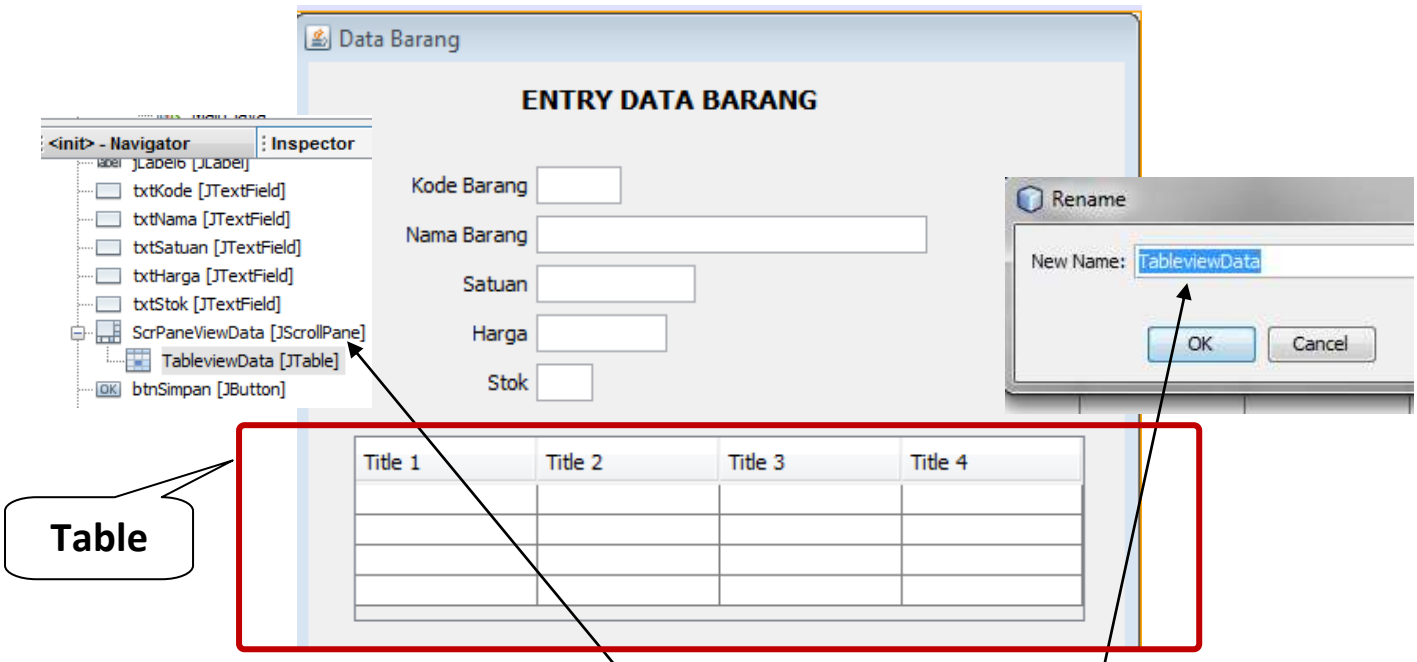


8) Ubah nama JTextField untuk memudahkan penulisan program dengan cara mengklik kanan pada masing-masing **JTextField** dan pilih **Change Variabel Name**. Ubah sesuai dengan contoh berikut :



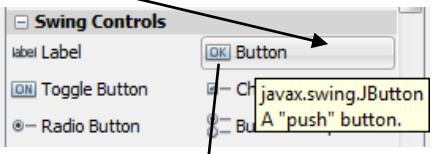
9) Tambahkan komponen **Table** dari **palette swing controls,** atur ukuran komponennya sehingga tampilan form sebagai berikut :

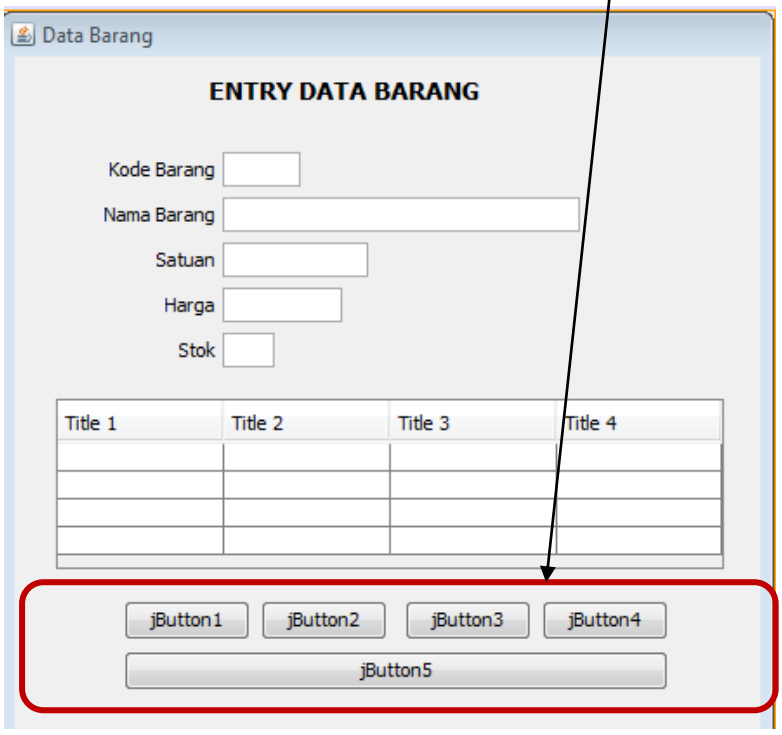




Ubah nama Tablenya (**change variable name**) menjadi : **TableviewData** dan JScrollPane1 menjadi **ScrPaneViewData** (lihat **di Inspector**).

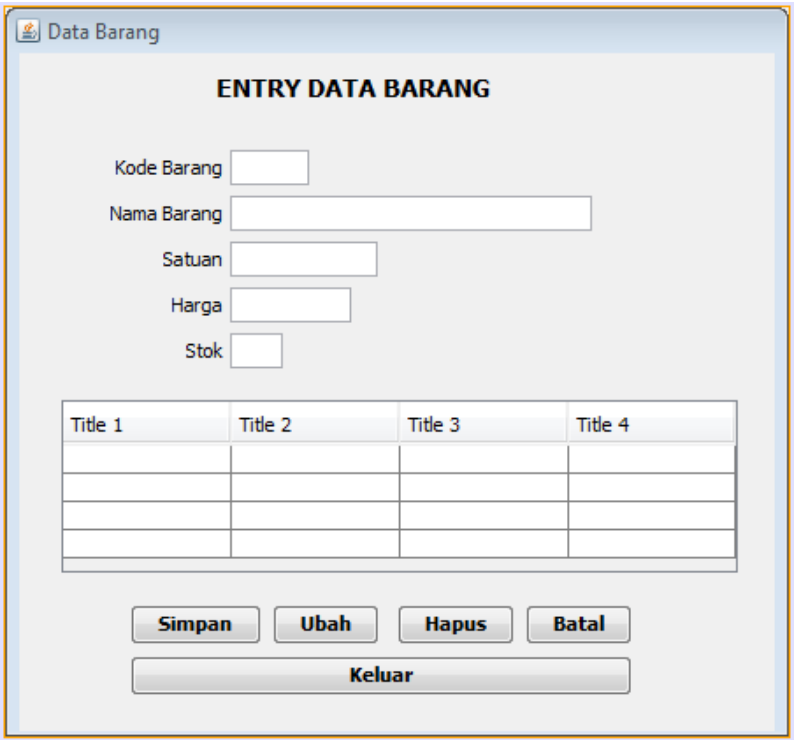
10) Tambahkan komponen **Button** dari **palette swing controls,** atur ukuran komponennya sehingga tampilan form sebagai berikut





11) Ubah text dan nama komponen button masing-masing sebagai berikut :
   - **jButton1**, Text diganti menjadi = **Simpan**, Nama Variabel = **btnSimpan**
   - **jButton2**, Text diganti menjadi = **Ubah**, Nama Variabel = **btnUbah**
   - **jButton3**, Text diganti menjadi = **Hapus**, Nama Variabel = **btnHapus**
   - **jButton4**, Text diganti menjadi = **Batal**, Nama Variabel = **btnBatal**
   - **jButton5**, Text diganti menjadi = **Keluar**, Nama Variabel = **btnKeluar**

   Sehingga bentuk akhir form Entry Data Barang sebagai berikut :



12) Klik tabulasi source dan tambahkan perintah berikut untuk mengimport beberapa library yang diperlukan dari packagenya (**Sintaks program yang ditambahkan terdapat pada kotak segi empat**).

```
14      * @author okkinn
15      */
16     import java.sql.*;
17     import javax.swing.JOptionPane;
18     import javax.swing.table.DefaultTableModel;
19
20     public class frmBarang extends javax.swing.JInternalFrame {
21
22         /** Creates new form frmBarang */
23         public frmBarang() {
24             initComponents();
25         }
26
```

13) Tambahkan perintah pada bagian deklarasi class dan konstruktor class dengan perintah berikut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmBarang extends javax.swing.JInternalFrame {

    /** Creates new form frmBarang */
    DefaultTableModel tabMode;

    public frmBarang() {
        initComponents();

        String [] row={"Kode","Nama","Satuan","Harga","Stok"};
        tabMode = new DefaultTableModel(null,row);

        TableviewData.setModel(tabMode);
        ScrPaneViewData.getViewport().add(TableviewData,null);
    }
```

14) Tambahkan methode kosong() dibawah konstruktor class tersebut. Methode tersebut berfungsi untuk membersihkan komponen dari data yang telah dinput/diolah sebelumnya sekaligus sebagai langkah awal untuk proses berikutnya. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
        String [] row={"Kode","Nama","Satuan","Harga","Stok"};
        tabMode = new DefaultTableModel(null,row);

        TableviewData.setModel(tabMode);
        ScrPaneViewData.getViewport().add(TableviewData,null);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code
    public void kosong()
    {
        clsBarang brg = new clsBarang();

        txtKode.setText("");
        txtNama.setText("");
        txtSatuan.setText("");
        txtHarga.setText("");
        txtStok.setText("");
        txtKode.requestFocus();
        btnSimpan.setEnabled(false);
        btnUbah.setEnabled(false);
        btnHapus.setEnabled(false);

        brg.autoKode();
        txtKode.setText(brg.getKode());

    }

    // Variables declaration - do not modify
    private javax.swing.JScrollPane ScrPaneViewData;
    private javax.swing.JTable TableviewData;
    private javax.swing.JButton btnBatal;
    private javax.swing.JButton btnHapus;
    private javax.swing.JButton btnKeluar;
    private javax.swing.JButton btnSimpan;
```

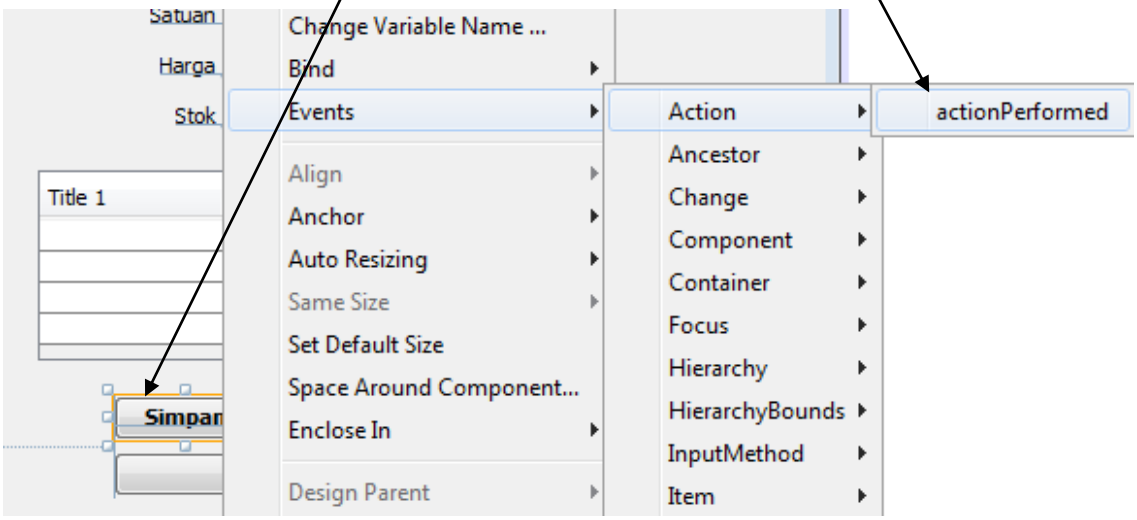
15) Tambahkan methode **tampilTabel()** yang berfungsi untuk menampilkan data ke tabel yang sudah disiapkan di form. Methode ini ditambahkan dibawah methode **kosong()**. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
        btnHapus.setEnabled(false);

        brg.autoKode();
        txtKode.setText(brg.getKode());
        txtKode.requestFocus();
    }

    public void tampilTabel()
    {   String kd, nm, sa, hr, sk;
        int baris=tabMode.getRowCount();
        for (int i=0;i<baris;i++)
        {
            tabMode.removeRow(0);
        }
        try
        {   Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select * from barang order by kdbrg asc";
            ResultSet rs=st.executeQuery(sql);
            while(rs.next())
            {
                kd=rs.getString("kdbrg");
                nm=rs.getString("nmbrg");
                sa=rs.getString("sat");
                hr=rs.getString("hrgbrg");
                sk=rs.getString("stok");
                String [] data={kd,nm,sa,hr,sk};
                tabMode.addRow(data);
            }
            rs.close();
            st.close();
        }
        catch(SQLException sqe)
        {}
    }

    // Variables declaration - do not modify
    private javax.swing.JScrollPane ScrPaneViewData;
    private javax.swing.JTable TableviewData;
    private javax.swing.JButton btnBatal;
    private javax.swing.JButton btnHapus;
    private javax.swing.JButton btnKeluar;
```

16) Kembali ke konstruktor clsBarang, tambahkan perintah berikut untuk memanggil methode **kosong()** dan **tampilTabel()**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmBarang extends javax.swing.JInternalFrame {

    /** Creates new form frmBarang */
    DefaultTableModel tabMode;

    public frmBarang() {
        initComponents();

        String [] row={"Kode","Nama","Satuan","Harga","Stok"};
        tabMode = new DefaultTableModel(null,row);

        TableviewData.setModel(tabMode);
        ScrPaneViewData.getViewport().add(TableviewData,null);

        kosong();
        tampilTabel();
    }
```

17) Tambahkan perintah melalui event **actionPerformed** pada object **btnSimpan** (tombol **simpan**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)



```
private void btnSimpanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clsBarang brg = new clsBarang();

    brg.setKode(txtKode.getText());
    brg.setNama(txtNama.getText());
    brg.setSatuan(txtSatuan.getText());
    brg.setHarga(Integer.parseInt(txtHarga.getText()));
    brg.setStok(Integer.parseInt(txtStok.getText()));
    brg.simpan();

    if (brg.getFlag()==1)
    {
        kosong();
        tampilTabel();
    }
}
```

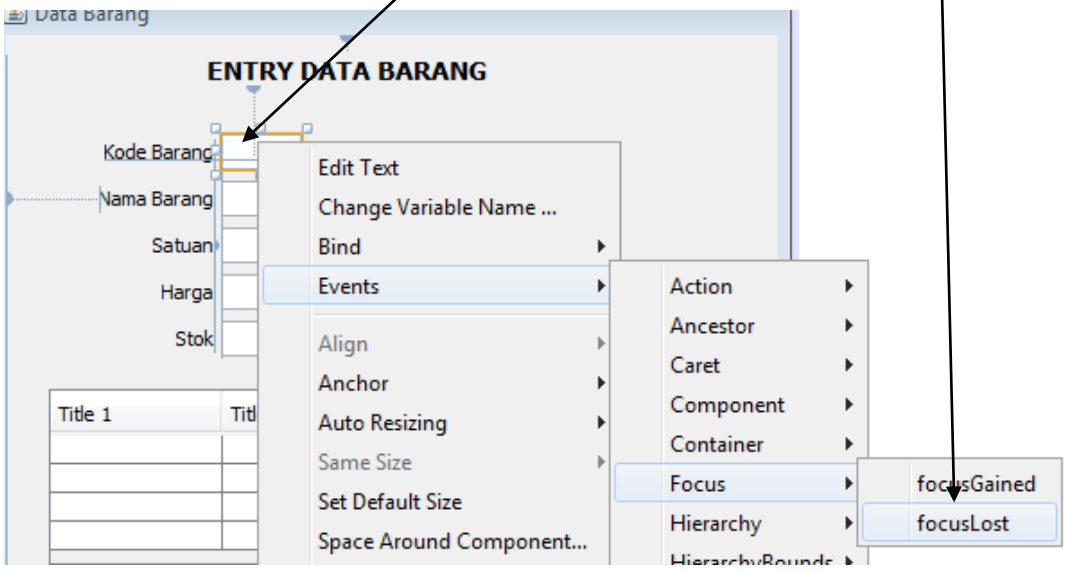
18) Tambahkan perintah melalui event **actionPerformed** pada object **btnUbah** (tombol **Ubah**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)

```
private void btnUbahActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clsBarang brg=new clsBarang();

    brg.setKode(txtKode.getText());
    brg.setNama(txtNama.getText());
    brg.setSatuan(txtSatuan.getText());
    brg.setHarga(Integer.parseInt(txtHarga.getText()));
    brg.setStok(Integer.parseInt(txtStok.getText()));
    brg.ubah();

    if(brg.getFlag()==2)
    {
        kosong();
        tampilTabel();
    }
}
```

19) Tambahkan perintah melalui event **actionPerformed** pada object **btnHapus** (tombol **Hapus**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clsBarang brg=new clsBarang();

    brg.setKode(txtKode.getText());
    brg.hapus();

    if (brg.getFlag()==3)
    {
        kosong();
        tampilTabel();
    }
}
```

20) Tambahkan perintah pada object **txtKode** melalui event **focusLost** dengan cara klik kanan pada object tersebut. Perintah ini diperlukan untuk memeriksa apakah kode yang diinput sudah ada atau belum di database. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).



```
private void txtKodeFocusLost(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:
        clsBarang brg=new clsBarang();

        brg.setKode(txtKode.getText());
        brg.tampil();

        if(brg.getFlag()==4)
        {
            txtNama.setText(brg.getNama());
            txtSatuan.setText(brg.getSatuan());
            txtHarga.setText(""+brg.getHarga());
            txtStok.setText(""+brg.getStok());
            txtNama.requestFocus();
            btnSimpan.setEnabled(false);
            btnUbah.setEnabled(true);
            btnHapus.setEnabled(true);
        }
        else
        {
            txtNama.setText("");
            txtSatuan.setText("");
            txtHarga.setText("");
            txtStok.setText("");
            txtNama.requestFocus();
            btnSimpan.setEnabled(true);
            btnUbah.setEnabled(false);
            btnHapus.setEnabled(false);
        }
    }
```

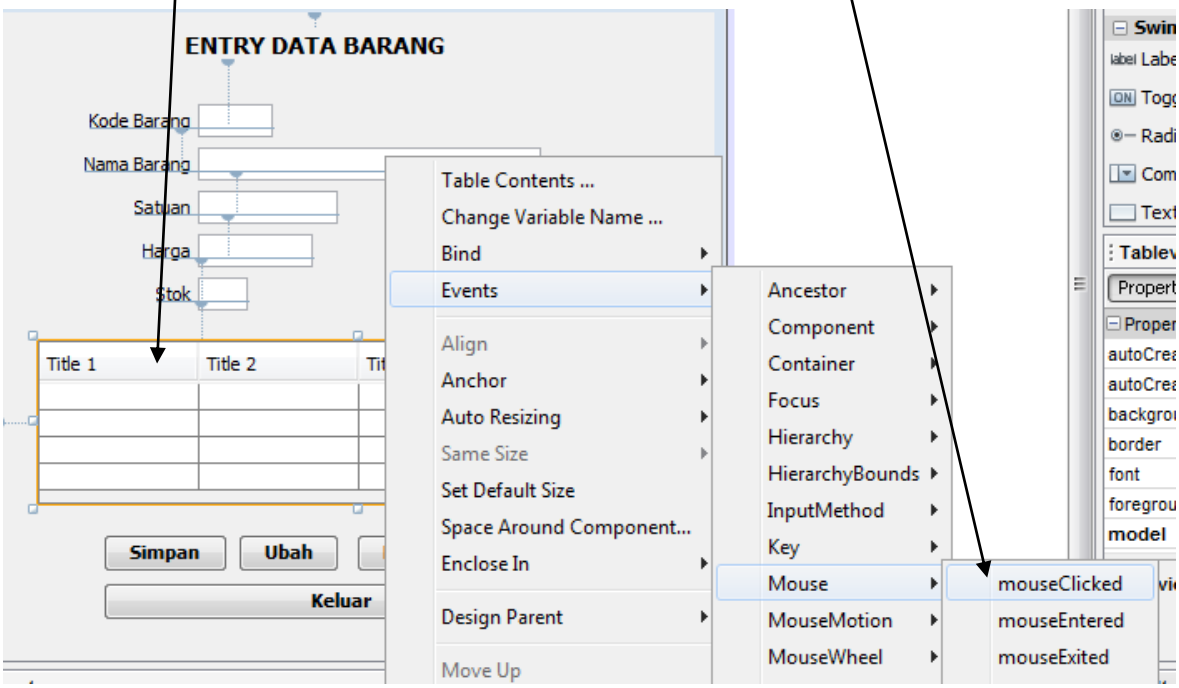
21) Tambahkan perintah melalui event **actionPerformed** pada object **btnBatal** (tombol **Batal**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
        kosong();
        tampilTabel();
}
```

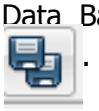
22) Tambahkan perintah melalui event **actionPerformed** pada object **btnKeluar** (tombol **Keluar**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

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

23) Tambahkan perintah melalui event **mouseClicked** pada object **TableviewData** dengan cara klik kanan pada object tersebut. Perintah ini diperlukan untuk menampilkan data ke objek yang telah ditentukan pada saat mouse di klik pada baris tertentu. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)



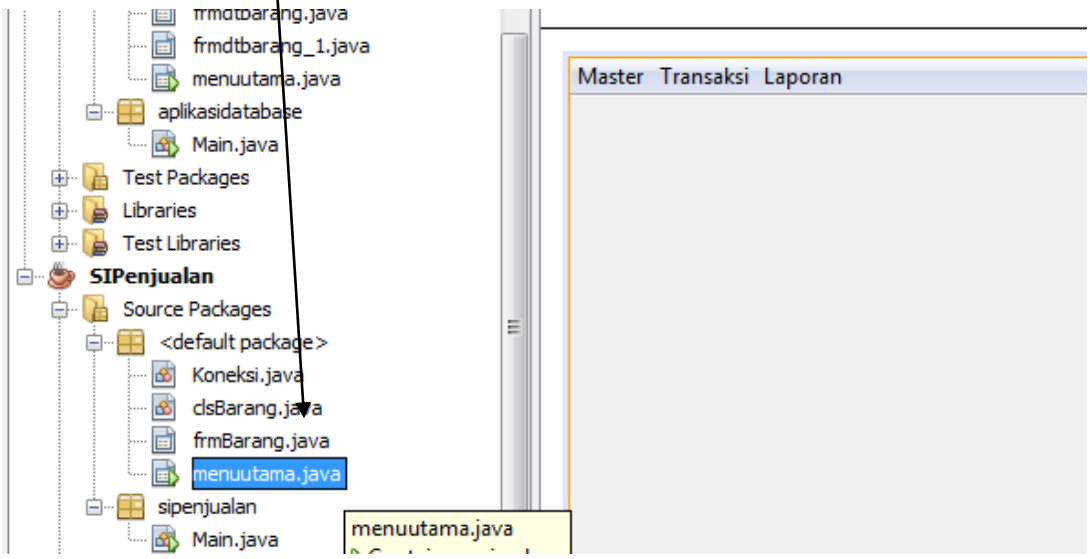

```
private void TableviewDataMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int tabelData=TableviewData.getSelectedRow();

    txtKode.setText(""+TableviewData.getValueAt(tabelData, 0));
    txtNama.setText(""+TableviewData.getValueAt(tabelData, 1));
    txtSatuan.setText(""+TableviewData.getValueAt(tabelData, 2));
    txtHarga.setText(""+TableviewData.getValueAt(tabelData, 3));
    txtStok.setText(""+TableviewData.getValueAt(tabelData, 4));
    txtNama.requestFocus();
    btnSimpan.setEnabled(false);
    btnUbah.setEnabled(true);
    btnHapus.setEnabled(true);
}
```

24) Secara Keseluruhan form Entry Data Barang sudah selesai dibuat, Project silahkan disimpan ulang /disave ( .

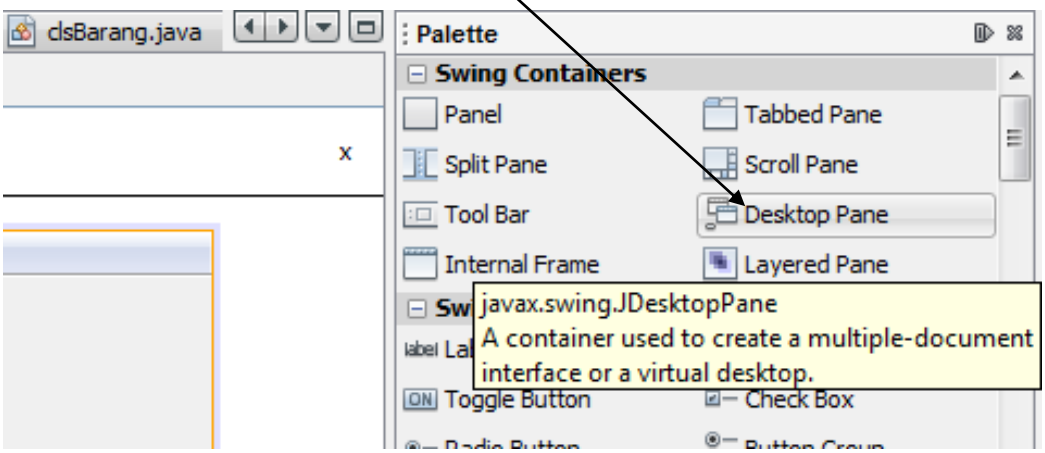
f. Integrasi Form Entry Data Barang dengan Menu Utama.

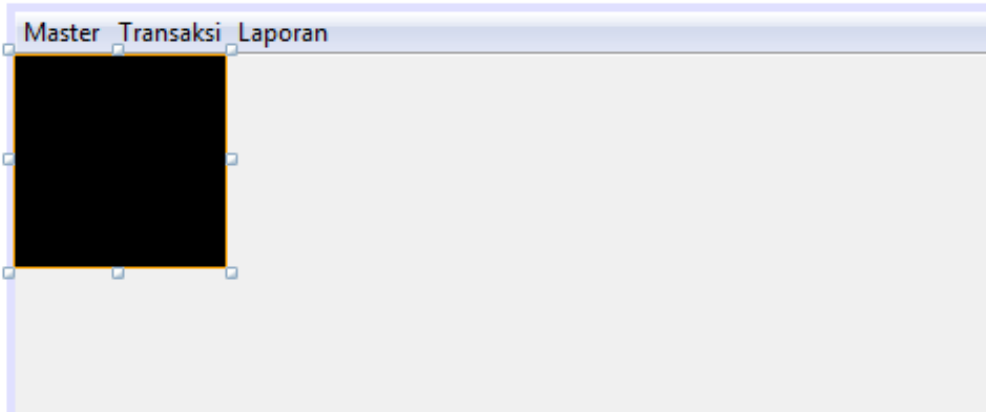
Proses ini bertujuan agar form entry data barang dapat digunakan apabila project sudah dijalankan. Seperti yang ditunjukkan pada proses pembuatan Form Entry Data Barang, form ini menggunakan JInternalFrame sebagai framenya dan mengakibatkan form ini tidak mempunyai methode **main()**, sehingga apabilaform akan dijalankan, maka harus merujuk pada form lain yang mempunyai methode main(). Dalam hal ini form yang mempunyai methode **main()** adalah menu utama sebagai induk dari project ini. Adapun langkah-langkahnya sebagai berikut :

1) Buka form **menu utama**. Form dapat dibuka dengan cara mendouble klik form tersebut pada jendela **project**. Pastikan tabulasi **design** dalam keadaan terpilih.



2) Tambahkan komponen **Desktop Pane** dari **Palette Swing Containers**. Dan tempatkan pada menu utama.



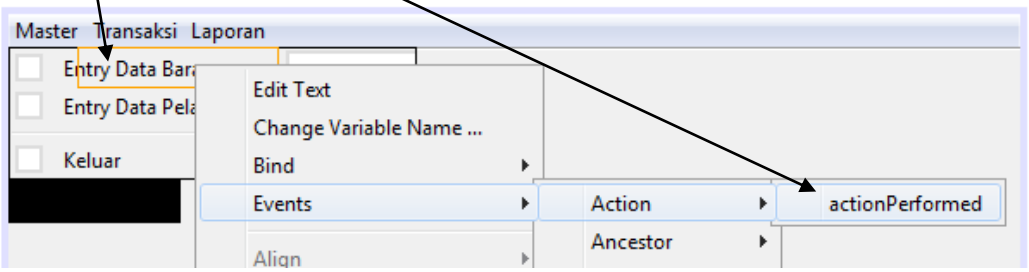


Ubah nama komponen tersebut menjadi **dskpane** (melalui **change variable name..**).

3) Klik tabulasi **source** dan tambahkan perintah pada konstruktor classnya untuk menempelkan objek dskpane ke frame. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
public class menuutama extends javax.swing.JFrame {

    /** Creates new form menuutama */
    public menuutama() {
        super("Sistem Informasi Penjualan Tunai");
        setLocation(100,100);
        initComponents();
        setContentPane(dskPane);
    }
```

4) Klik tabulasi **design**. Tambahkan perintah pada sub menu **Entry Data Barang** untuk menghubungkan ke class/form data barang. Perintah ditambahkan dengan cara klik kanan pada sub menu tersebut dan melalui event **ActionPerformed**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

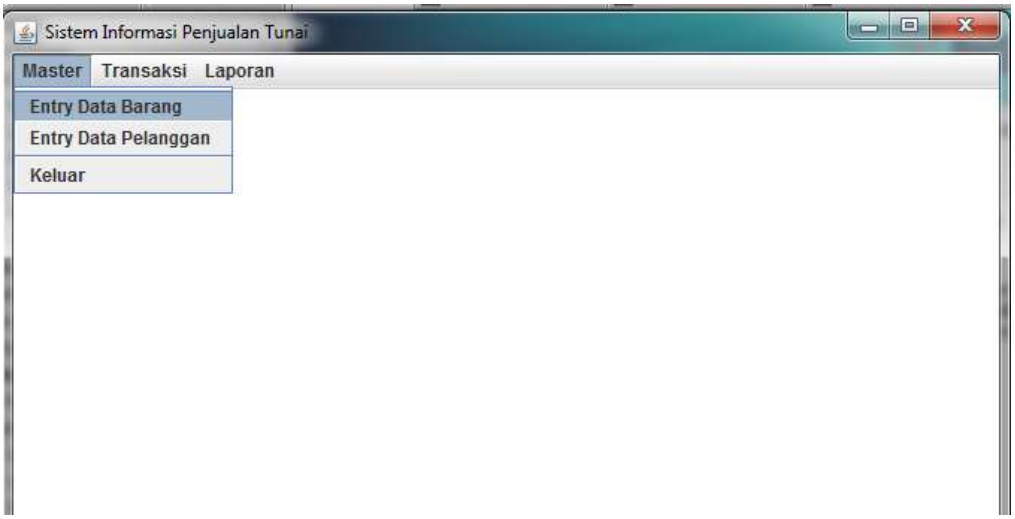


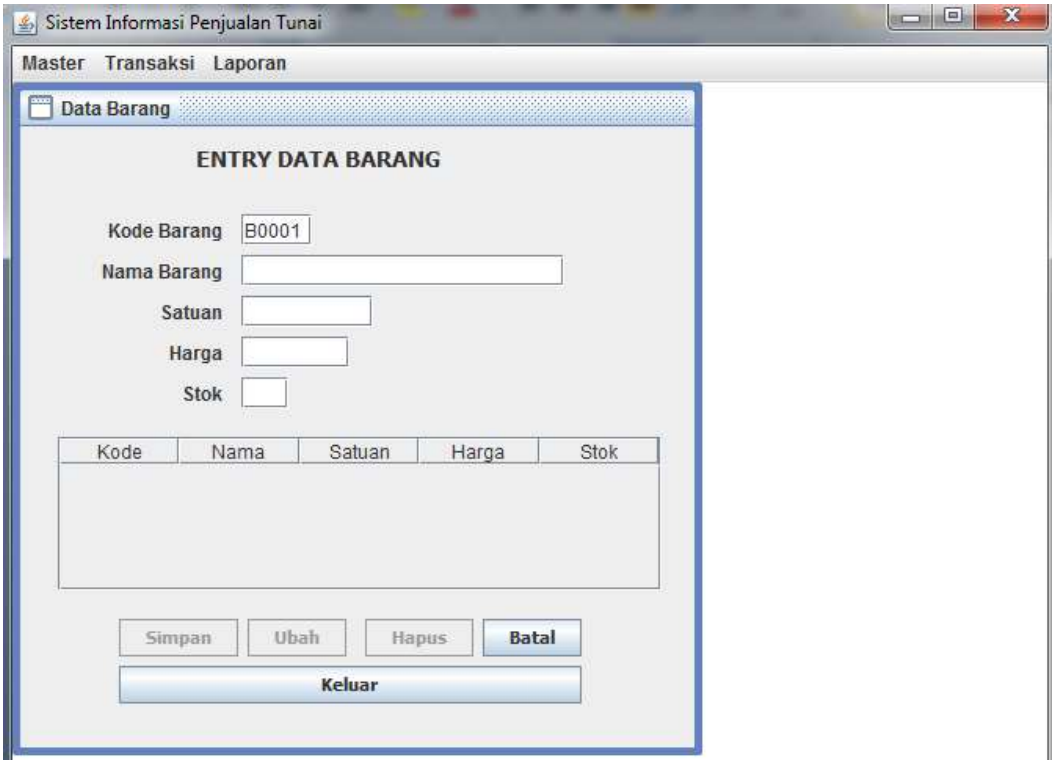
```
private void frmBarangActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    frmBarang barang = new frmBarang();

    dskPane.add(barang);
    barang.setVisible(true);
}
```

5) Klik tabulasi **design.** Tambahkan perintah pada sub menu **Keluar** untuk menutup/keluar dari program. Perintah ditambahkan dengan cara klik kanan pada sub menu tersebut dan melalui event **ActionPerformed**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
private void frmExitActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int keluar=JOptionPane.showConfirmDialog(null, "Yakin Ingin Keluar ??",
            "Keluar",JOptionPane.YES_NO_OPTION);
    if (keluar==JOptionPane.YES_OPTION)
    {
        System.exit(0);
    }
}
```

6) Integrasi form data barang dengan menu utama sudah selesai. Untuk melihat hasil programnya, pastikan **menu utama** dalam keadaan terpilih, kemudian klik **Run** dan pilih **Run File** pada netBeans atau dengan tombol **shift+F6**. Berikut hasil programnya :

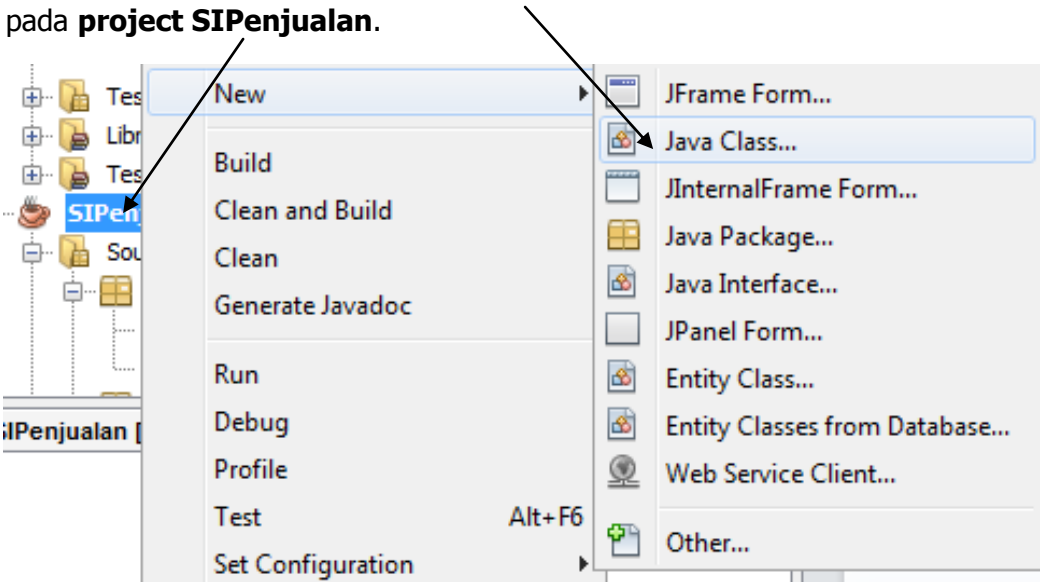




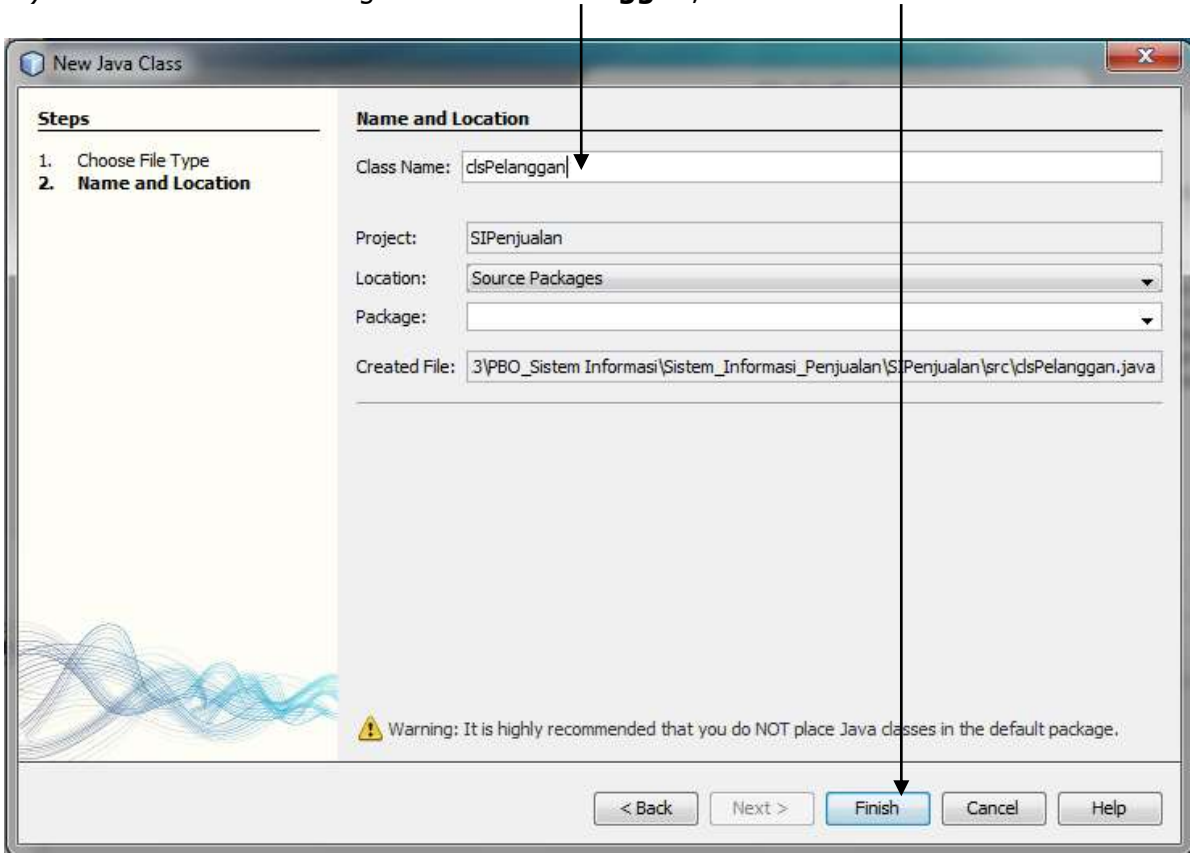
g. Buat class pelanggan

Adapun proses membuat class pelanggan hampir sama dengan class barang dan langkah-langkahnya sebagai berikut :

1) Tambahkan sebuah class baru (**Java Class**) dengan cara mengklik kanan pada **project SIPenjualan**.



2) Class tersebut beri dengan nama **clsPelanggan**, kemudian klik **finish**.



3) Pada Class tersebut, tambahkan sintaks untuk mengimport beberapa library yang diperlukan (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
public class clsPelanggan {

}
```

4) Didalam class **clsPelanggan** tersebut, tambahkan perintah untuk membuat variabel/atribut dan methode yang diperlukan untuk menentukan atau mengirimkan nilai dari atau ke variabel tersebut (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
public class clsPelanggan {
    protected String kode, nama,alamat, telepon;
    protected int flag;

    public void setKode (String kd)
    { kode = kd;}
    public void setNama (String nm)
    { nama = nm;}
    public void setAlamat (String alm)
    { alamat = alm;}
    public void setTelepon (String tlp)
    { telepon = tlp;}
    public String getKode()
    { return(kode); }
    public String getNama()
    { return(nama); }
    public String getAlamat()
    { return(alamat);}
    public String getTelepon()
    { return(telepon);}
    public void setFlag(int F)
    { flag = F;}
    public int getFlag()
    { return(flag);}
}
```

5) Didalam class **clsPelanggan** tersebut, tambahkan methode **simpan** yang berfungsi untuk menyimpan data ke database. Methode ditambahkan dibawah metode **getFlag()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
    public int getFlag()
    { return(flag);}

    public void simpan()
    {
        try
        {
            Koneksi k= new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="insert into pelanggan values(";
                   sql+="'"+getKode()+"','"+getNama()+"',";
                   sql+="'"+getAlamat()+"','"+getTelepon()+"');";

              st.executeUpdate(sql);
              setFlag(1);
              st.close();
              cn.close();
              JOptionPane.showMessageDialog(null, "Data Berhasil Disimpan",
                      "SIMPAN",JOptionPane.INFORMATION_MESSAGE);
        }
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data Gagal Disimpan",
                    "Gagal Simpan", JOptionPane.ERROR_MESSAGE);
        }
    }
}
```

6) Didalam class **clsPelanggan** tersebut, tambahkan methode **ubah** yang berfungsi untuk mengubah data di database. Methode ditambahkan dibawah metode **simpan()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data Gagal Disimpan",
                    "Gagal Simpan", JOptionPane.ERROR_MESSAGE);
        }
    }

    public void ubah()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="update pelanggan set ";
                   sql+="nmplg='"+getNama()+"',almplg='"+getAlamat()+"',";
                   sql+="tlpplg='"+getTelepon()+"' where ";
                   sql+="kdplg='"+getKode()+"'";

               st.executeUpdate(sql);
               setFlag(2);
               st.close();
               cn.close();
               JOptionPane.showMessageDialog(null, "Data Berhasil diUbah",
                       "UBAH",JOptionPane.INFORMATION_MESSAGE);
        }
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data GAGAL diUbah",
                    "GAGAL UBAH",JOptionPane.ERROR_MESSAGE);
        }
    }
}
```

Perhatikan : antara set dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh : set” (SALAH), set ” (BETUL).

Perhatikan : antara where dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh : where” (SALAH), where ” (BETUL).

7) Didalam class **clsPelanggan** tersebut, tambahkan methode **hapus** yang berfungsi untuk menghapus data di database. Methode ditambahkan dibawah metode **ubah()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data GAGAL diUbah",
                    "GAGAL UBAH",JOptionPane.ERROR_MESSAGE);
        }
    }

    public void hapus()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="delete from pelanggan ";
                   sql+="where kdplg='"+getKode()+"'";

              st.executeUpdate(sql);
              setFlag(3);
              st.close();
              cn.close();
              JOptionPane.showMessageDialog(null, "Data Berhasil diHAPUS",
                      "HAPUS",JOptionPane.INFORMATION_MESSAGE);
        }
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data GAGAL diHapus",
                    "GAGAL HAPUS",JOptionPane.ERROR_MESSAGE);
        }
    }
}
```

Perhatikan : antara pelanggan dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh :
pelanggan” (SALAH),
pelanggan ” (BETUL).

8) Didalam class **clsPelanggan** tersebut, tambahkan methode **tampil** yang berfungsi untuk menampilkan data dari database. Methode ditambahkan dibawah metode **hapus()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Data GAGAL diHapus",
                    "GAGAL HAPUS",JOptionPane.ERROR_MESSAGE);
        }
    }

    public void tampil()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select * from pelanggan ";
                   sql+="where kdplg='"+getKode()+"'";

            ResultSet rs=st.executeQuery(sql);
            if(rs.next())
            {
                setFlag(4);
                setKode(rs.getString("kdplg"));
                setNama(rs.getString("nmplg"));
                setAlamat(rs.getString("almplg"));
                setTelepon(rs.getString("tlpplg"));
                st.close();
                rs.close();
            }
        }
        catch(SQLException sqe)
        {}
    }
}
```

Perhatikan : antara pelanggan dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh : pelanggan” (SALAH), pelanggan ” (BETUL).

9) Didalam class **clsPelanggan** tersebut, tambahkan methode **autoKode** yang berfungsi untuk membuat kode barang secara otomatis. Methode ditambahkan dibawah metode **tampil()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        {}
    }

    public void autoKode()
    {
        try
        {
            int hit=0;

            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select count(kdplg) from pelanggan";
            ResultSet rs=st.executeQuery(sql);

            if (rs.next())
            {
                if(Integer.parseInt(rs.getString(1))==0)
                {
                    setKode("P0001");
                    rs.close();
                    st.close();
                }
                else
                {
                    sql="select Max(mid(kdplg,2,4)) from pelanggan";
                    rs=st.executeQuery(sql);
                    rs.next();
                    hit = (Integer.parseInt(rs.getString(1)))+1;
                    if (hit < 10)
                    { setKode("P000"+hit);}
                    else if (hit < 100)
                    { setKode("P00"+hit);}
                    else if (hit < 1000)
                    { setKode("P0"+hit);}
                    else
                    { setKode("P"+hit);}
                    st.close();
                    rs.close();
                }
            }
        }
        catch(SQLException sqe)
        {}
    }
}
```

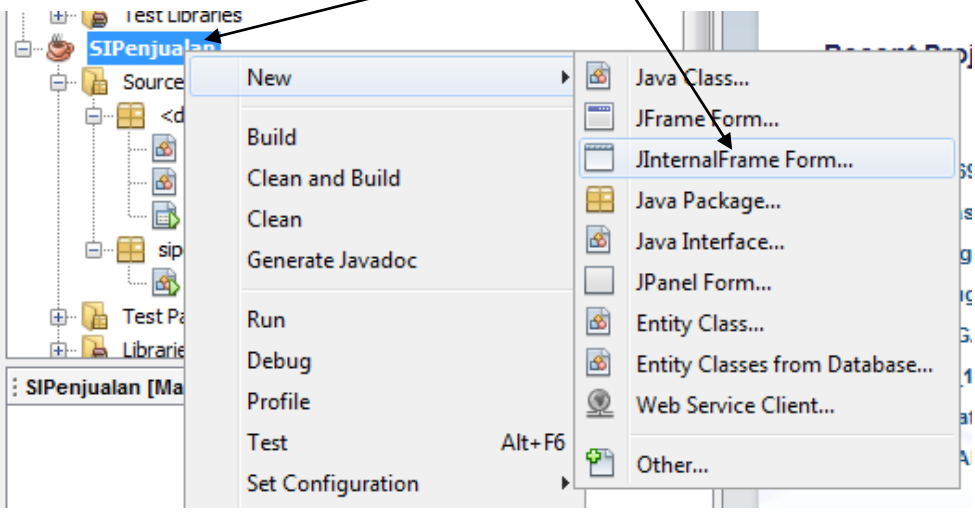
10) Secara keseluruhan class **clsPelanggan** sudah selesai dibuat. Project dapat disimpan ulang/disave ( .

h. Buat Form Entry Data Pelanggan

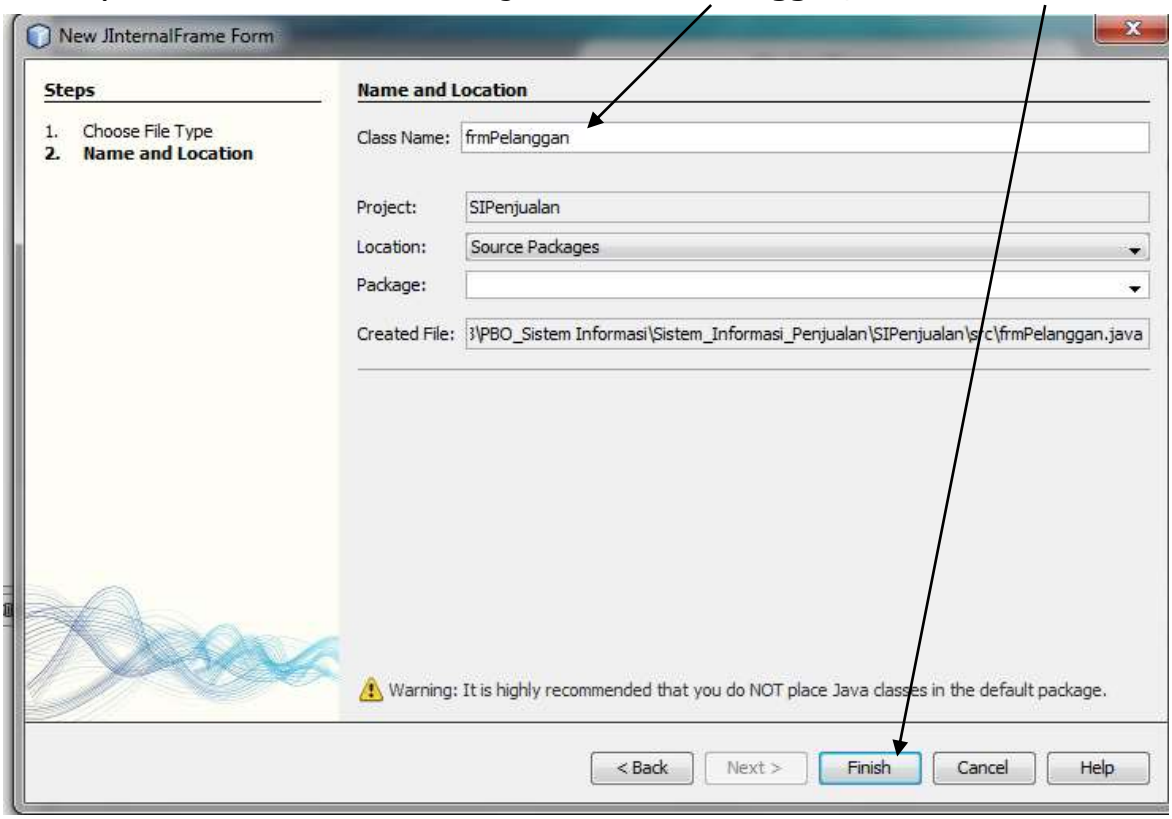
Dikarenakan menggunakan menu utama sebagai integrasinya, maka form yang akan dibuat JInternalFrame, sehingga tanpa melalui menu utama, maka form tidak akan bisa dijalankan. Langkah-langkahnya hampir sama dengan membuat form entry data barang.

Langkah-langkahnya sebagai berikut :

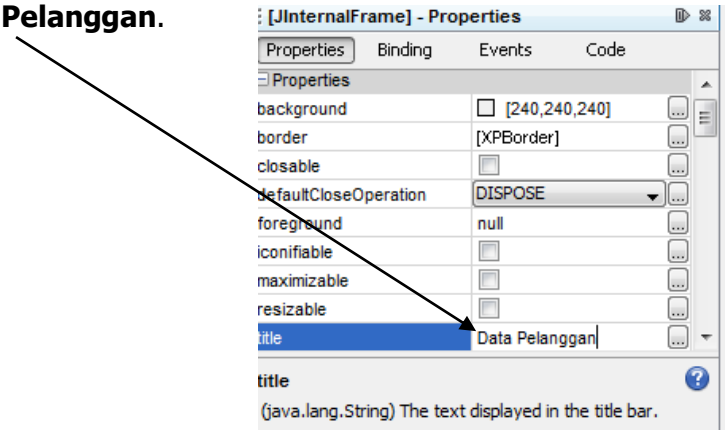
1) Tambahkan sebuah form baru (**JInternalFrame Form**), dengan cara mengklik kanan pada project **SIPenjualan**.



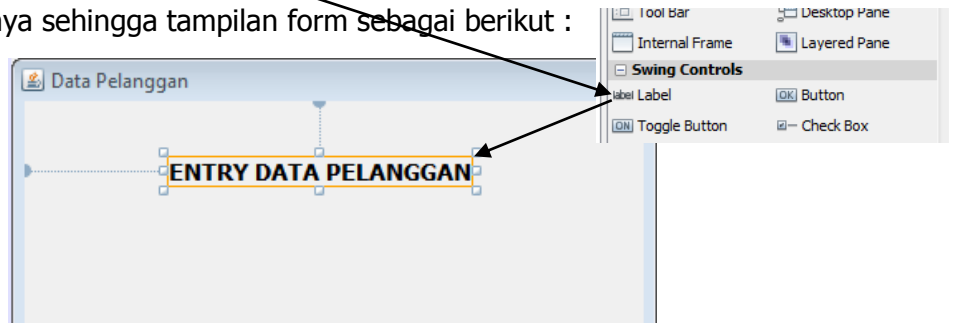
2) Form baru tersebut beri dengan nama **frmPelanggan**, kemudian klik **Finish**.



3) Beri judul form tersebut pada kolom yang sudah disediakan dengan nama : **Data Pelanggan**.



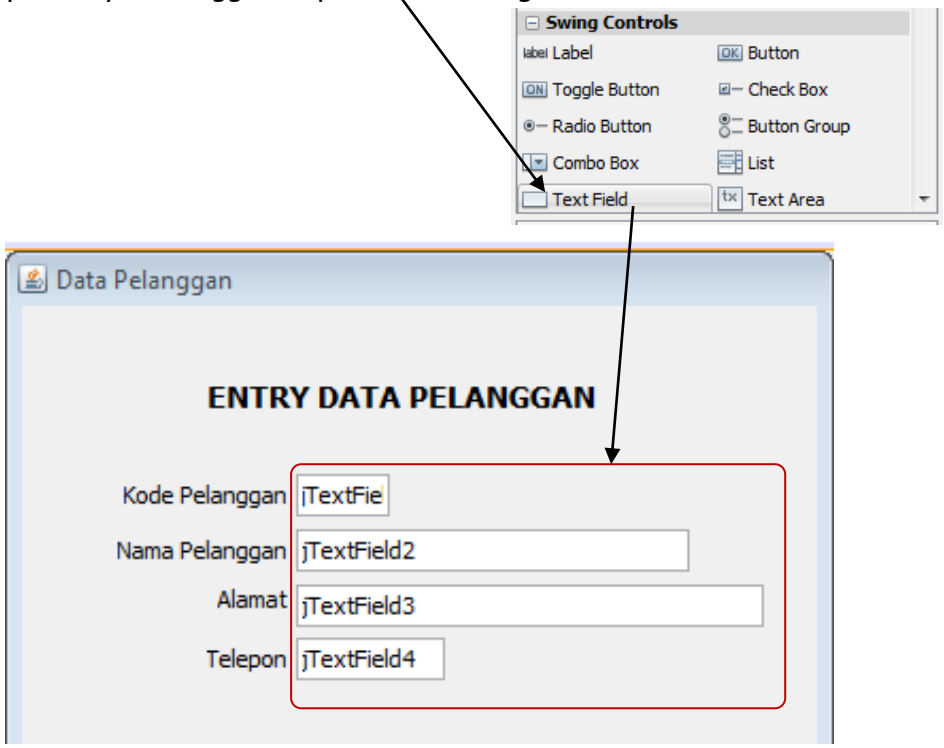
4) Tambahkan komponen **label** dari **palette swing controls**, atur huruf dan fontnya sehingga tampilan form sebagai berikut :



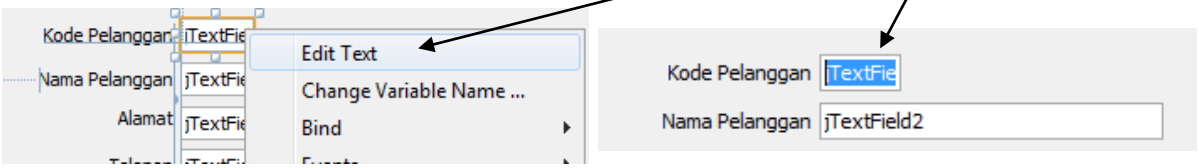
Klik kanan pada komponen **label** tersebut dan pilih **Edit Text** untuk mengubah tulisan.

5) Tambahkan komponen **Label** dari **palette swing controls**, atur huruf dan fontnya sehingga tampilan form sebagai berikut :

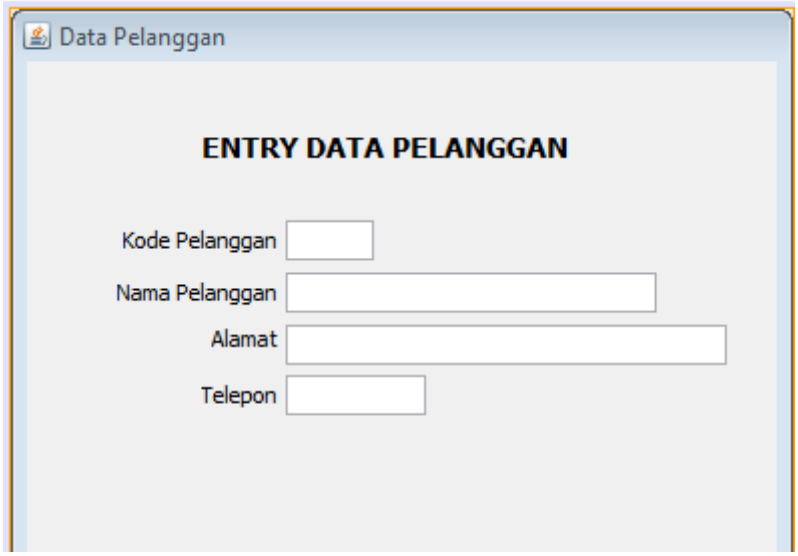
6) Tambahkan komponen **Text Field** dari **palette swing controls,** atur ukuran komponennya sehingga tampilan form sebagai berikut :



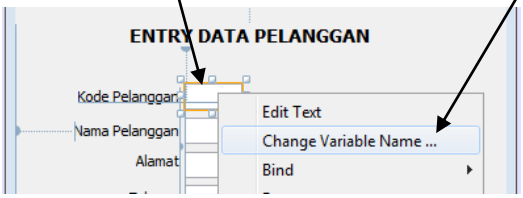
7) Hapus tulisan pada jTextField tersebut dengan cara mengklik kanan pada masing-masing-masing jTextField, pilih **Edit Text** dan di**hapus text**nya.

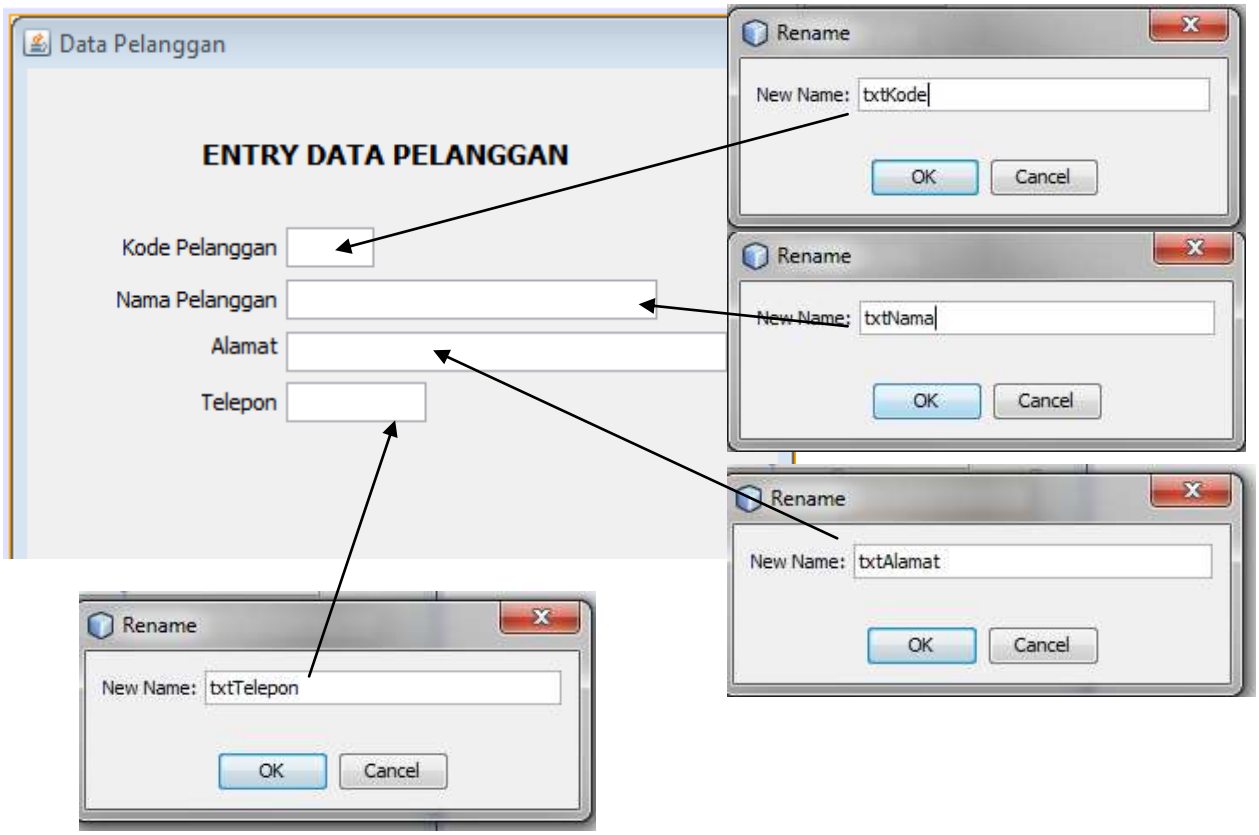


Tampilan sebagai berikut :

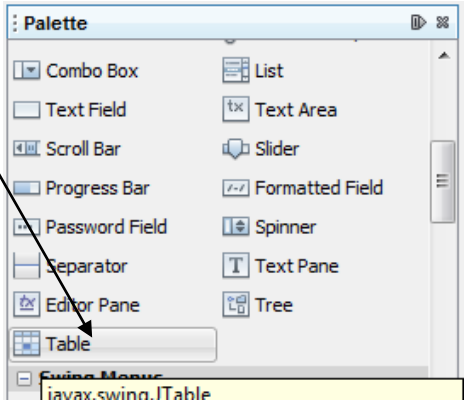


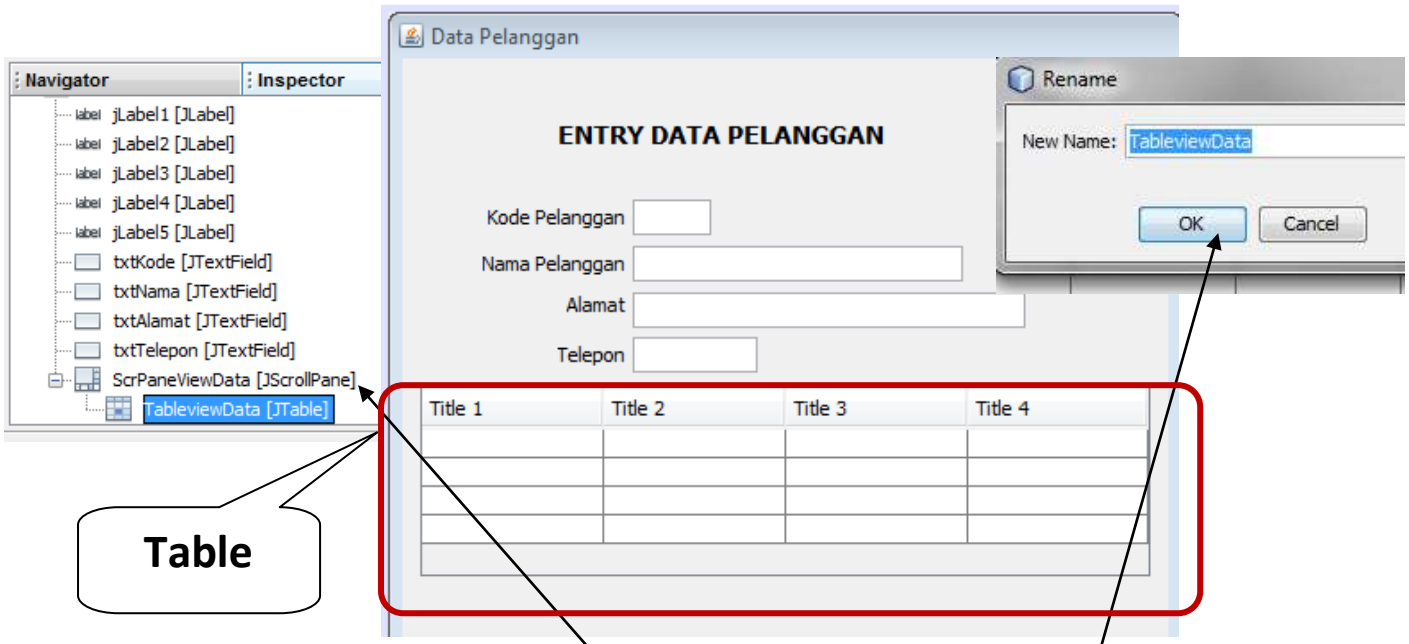
8) Ubah nama JTextField untuk memudahkan penulisan program dengan cara mengklik kanan pada masing-masing **JTextField** dan pilih **Change Variabel Name**. Ubah sesuai dengan contoh berikut :





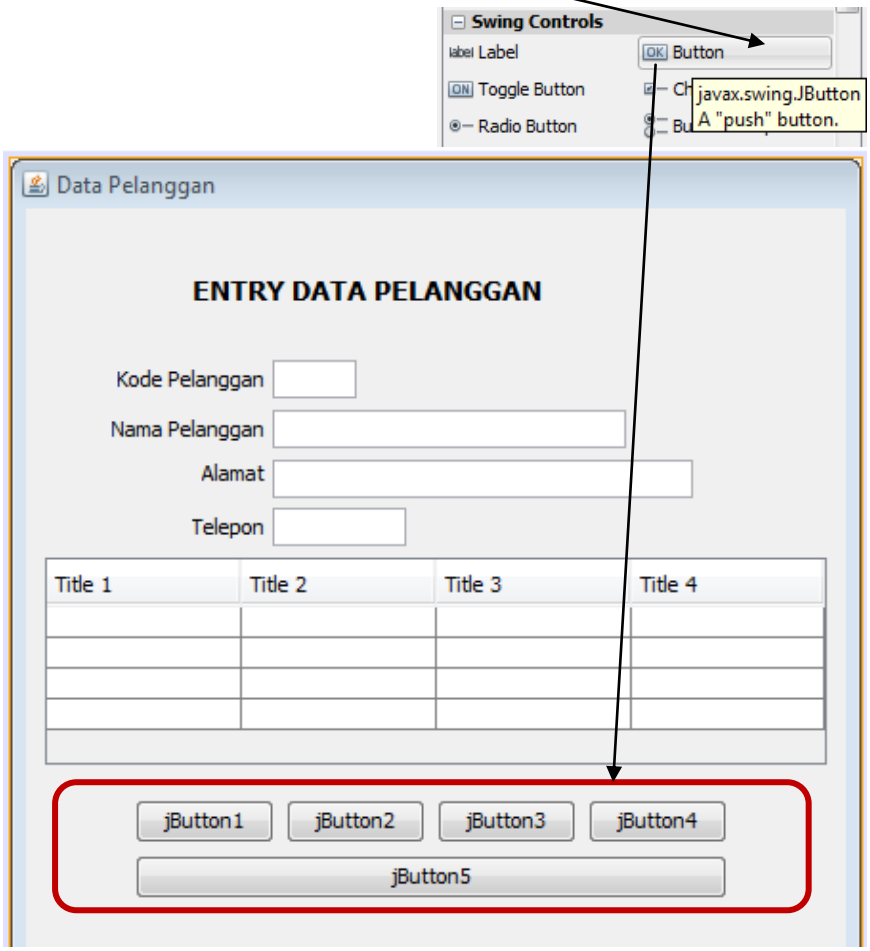
9) Tambahkan komponen **Table** dari **palette swing controls,** atur ukuran komponennya sehingga tampilan form sebagai berikut :





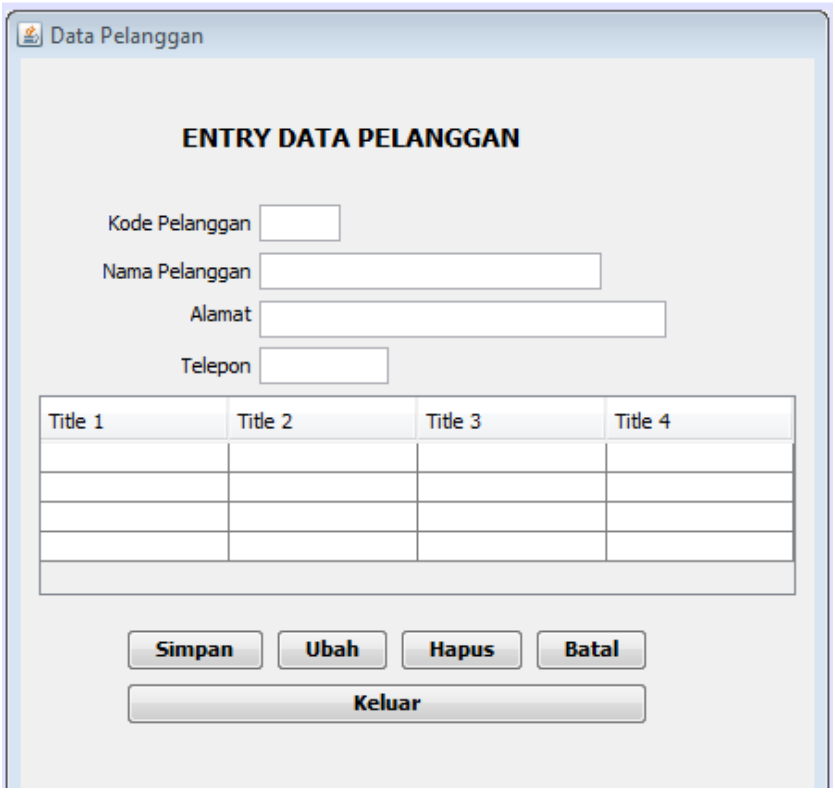
Ubah nama Tablenya (**change variable name**) menjadi : **TableviewData** dan JScrollPane1 menjadi **ScrPaneViewData** (lihat **di Inspector**).

10) Tambahkan komponen **Button** dari **palette swing controls,** atur ukuran komponennya sehingga tampilan form sebagai berikut



11) Ubah text dan nama komponen button masing-masing sebagai berikut :
   - **jButton1**, Text diganti menjadi = **Simpan**, Nama Variabel = **btnSimpan**
   - **jButton2**, Text diganti menjadi = **Ubah**, Nama Variabel = **btnUbah**
   - **jButton3**, Text diganti menjadi = **Hapus**, Nama Variabel = **btnHapus**
   - **jButton4**, Text diganti menjadi = **Batal**, Nama Variabel = **btnBatal**
   - **jButton5**, Text diganti menjadi = **Keluar**, Nama Variabel = **btnKeluar**

   Sehingga bentuk akhir form Entry Data Pelanggan sebagai berikut :



12) Klik tabulasi source dan tambahkan perintah berikut untuk mengimport beberapa library yang diperlukan dari packagenya (**Sintaks program yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmPelanggan extends javax.swing.JInternalFrame {

    /** Creates new form frmPelanggan */
    public frmPelanggan() {
        initComponents();
    }
```

13) Tambahkan perintah pada bagian deklarasi class dan konstruktor class dengan perintah berikut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmPelanggan extends javax.swing.JInternalFrame {

    /** Creates new form frmPelanggan */
    DefaultTableModel tabMode;

    public frmPelanggan() {
        initComponents();

        String [] row={"Kode","Nama","Alamat","Telepon"};
        tabMode = new DefaultTableModel(null,row);

        TableviewData.setModel(tabMode);
        ScrPaneViewData.getViewport().add(TableviewData,null);
    }

    /** This method is called from within the constructor to
```

14) Tambahkan methode kosong() dibawah konstruktor class tersebut. Methode tersebut berfungsi untuk membersihkan komponen dari data yang telah dinput/diolah sebelumnya sekaligus sebagai langkah awal untuk proses berikutnya. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
        String [] row={"Kode","Nama","Alamat","Telepon"};
        tabMode = new DefaultTableModel(null,row);

        TableviewData.setModel(tabMode);
        ScrPaneViewData.getViewport().add(TableviewData,null);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code
    public void kosong()
    {
        clsPelanggan plg=new clsPelanggan();

        txtKode.setText("");
        txtNama.setText("");
        txtAlamat.setText("");
        txtTelepon.setText("");
        txtKode.requestFocus();
        btnSimpan.setEnabled(false);
        btnUbah.setEnabled(false);
        btnHapus.setEnabled(false);

        plg.autoKode();
        txtKode.setText(plg.getKode());
    }

    // Variables declaration - do not modify
    private javax.swing.JScrollPane ScrPaneViewData;
    private javax.swing.JTable TableviewData;
    private javax.swing.JButton btnBatal;
    private javax.swing.JButton btnHapus;
    private javax.swing.JButton btnKeluar;
    private javax.swing.JButton btnSimpan;
    private javax.swing.JButton btnUbah;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
```

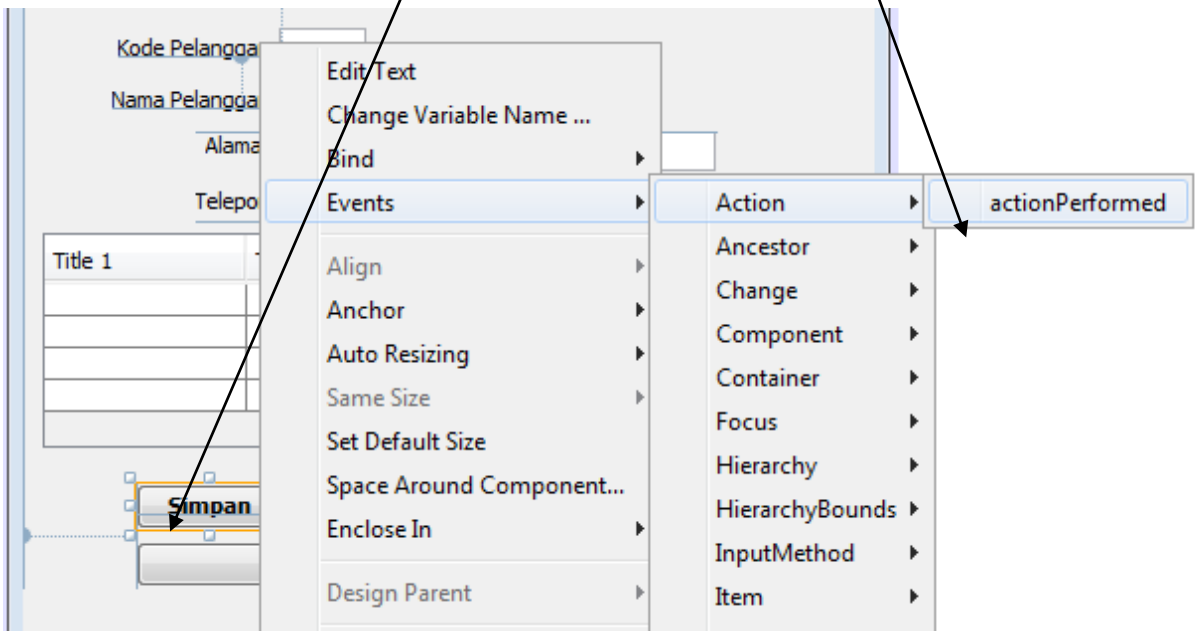
15) Tambahkan methode **tampilTabel()** yang berfungsi untuk menampilkan data ke tabel yang sudah disiapkan di form. Methode ini ditambahkan dibawah methode **kosong()**. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
        plg.autoKode();
        txtKode.setText(plg.getKode());
    }

    public void tampilTabel()
    {
        String kd, nm, al, tl;
        int baris=tabMode.getRowCount();
        for (int i=0;i<baris;i++)
        {
            tabMode.removeRow(0);
        }
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select * from pelanggan order by kdplg asc";
            ResultSet rs=st.executeQuery(sql);
            while(rs.next())
            {
                kd=rs.getString("kdplg");
                nm=rs.getString("nmplg");
                al=rs.getString("almplg");
                tl=rs.getString("tlpplg");
                String [] data={kd,nm,al,tl};
                tabMode.addRow(data);
            }
            rs.close();
            st.close();
        }
        catch (SQLException sqe)
        {}
    }

    // Variables declaration - do not modify
    private javax.swing.JScrollPane ScrPaneViewData;
    private javax.swing.JTable TableviewData;
    private javax.swing.JButton btnBatal;
    private javax.swing.JButton btnHapus;
    private javax.swing.JButton btnKeluar;
    private javax.swing.JButton btnSimpan;
    private javax.swing.JButton btnUbah;
    private javax.swing.JLabel jLabel1;
```

16) Kembali ke konstruktor **frmPelanggan**, tambahkan perintah berikut untuk memanggil methode **kosong()** dan **tampilTabel()**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmPelanggan extends javax.swing.JInternalFrame {

    /** Creates new form frmPelanggan */
    DefaultTableModel tabMode;

    public frmPelanggan() {
        initComponents();

        String [] row={"Kode","Nama","Alamat","Telepon"};
        tabMode = new DefaultTableModel(null,row);

        TableviewData.setModel(tabMode);
        ScrPaneViewData.getViewport().add(TableviewData,null);

        kosong();
        tampilTabel();
    }
```

17) Tambahkan perintah melalui event **actionPerformed** pada object **btnSimpan** (tombol **simpan**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)



```
private void btnSimpanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clsPelanggan plg=new clsPelanggan();

    plg.setKode(txtKode.getText());
    plg.setNama(txtNama.getText());
    plg.setAlamat(txtAlamat.getText());
    plg.setTelepon(txtTelepon.getText());
    plg.simpan();

    if(plg.getFlag()==1)
    {
        kosong();
        tampilTabel();
    }
}
```

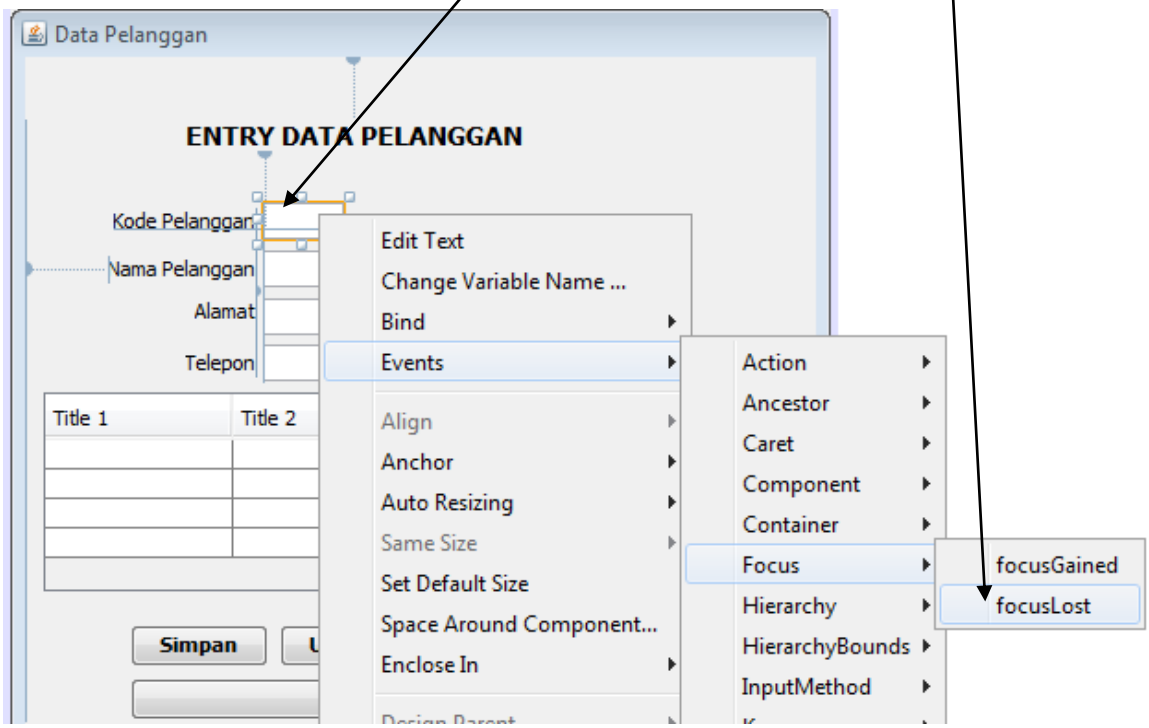
18) Tambahkan perintah melalui event **actionPerformed** pada object **btnUbah** (tombol **Ubah**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)

```
private void btnUbahActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clsPelanggan plg=new clsPelanggan();

    plg.setKode(txtKode.getText());
    plg.setNama(txtNama.getText());
    plg.setAlamat(txtAlamat.getText());
    plg.setTelepon(txtTelepon.getText());
    plg.ubah();

    if (plg.getFlag()==2)
    {
        kosong();
        tampilTabel();
    }

}
```

19) Tambahkan perintah melalui event **actionPerformed** pada object **btnHapus** (tombol **Hapus**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clsPelanggan plg=new clsPelanggan();

    plg.setKode(txtKode.getText());
    plg.hapus();

    if(plg.getFlag()==3)
    {
        kosong();
        tampilTabel();
    }
}
```

20) Tambahkan perintah pada object **txtKode** melalui event **focusLost** dengan cara klik kanan pada object tersebut. Perintah ini diperlukan untuk memeriksa apakah kode yang diinput sudah ada atau belum di database. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).



```
private void txtKodeFocusLost(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:
    clsPelanggan plg=new clsPelanggan();

    plg.setKode(txtKode.getText());
    plg.tampil();

    if(plg.getFlag()==4)
    {
        txtNama.setText(plg.getNama());
        txtAlamat.setText(plg.getAlamat());
        txtTelepon.setText(plg.getTelepon());
        txtNama.requestFocus();
        btnSimpan.setEnabled(false);
        btnUbah.setEnabled(true);
        btnHapus.setEnabled(true);
    }
    else
    {
        txtNama.setText("");
        txtAlamat.setText("");
        txtTelepon.setText("");
        txtNama.requestFocus();
        btnSimpan.setEnabled(true);
        btnUbah.setEnabled(false);
        btnHapus.setEnabled(false);
    }
}
```

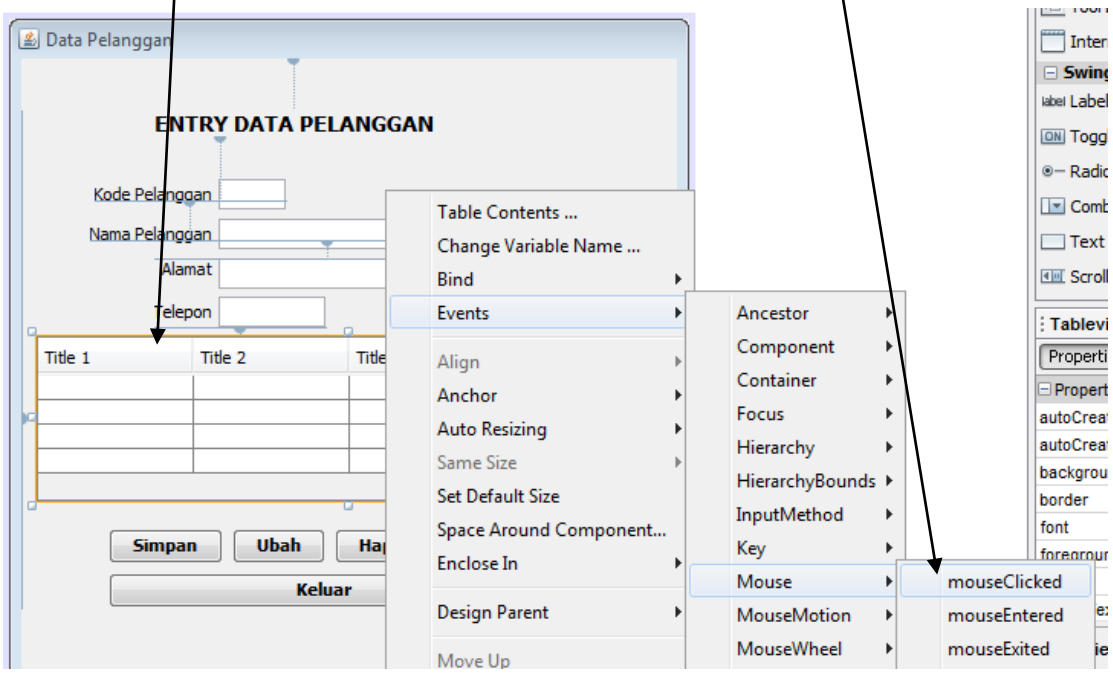
21) Tambahkan perintah melalui event **actionPerformed** pada object **btnBatal** (tombol **Batal**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    kosong();
    tampilTabel();
}
```

22) Tambahkan perintah melalui event **actionPerformed** pada object **btnKeluar** (tombol **Keluar**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

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

23) Tambahkan perintah melalui event **mouseClicked** pada object **TableviewData** dengan cara klik kanan pada object tersebut. Perintah ini diperlukan untuk menampilkan data ke objek yang telah ditentukan pada saat mouse di klik pada baris tertentu. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)

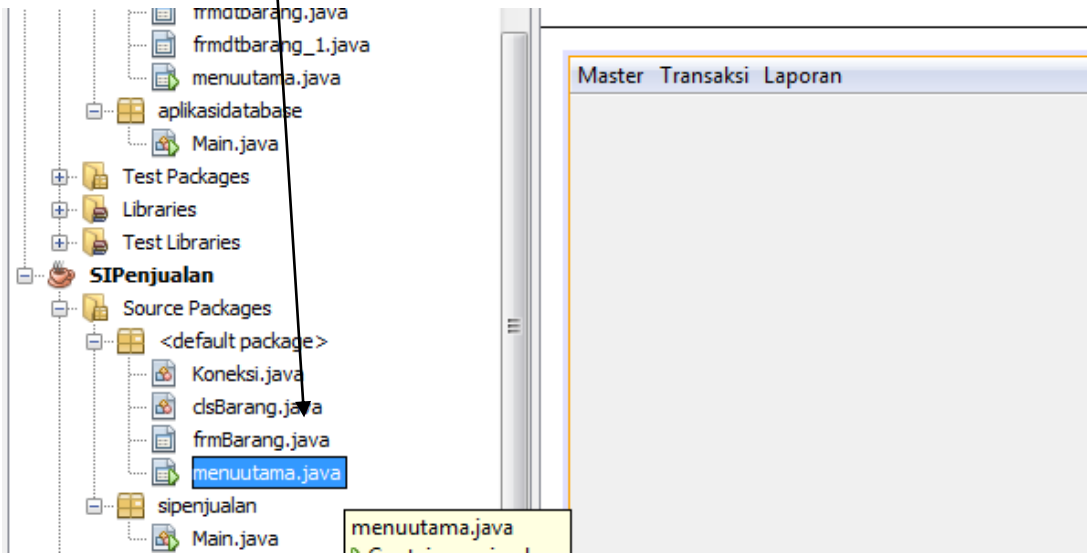



```
private void TableviewDataMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int tabelData=TableviewData.getSelectedRow();

    txtKode.setText(""+TableviewData.getValueAt(tabelData, 0));
    txtNama.setText(""+TableviewData.getValueAt(tabelData, 1));
    txtAlamat.setText(""+TableviewData.getValueAt(tabelData, 2));
    txtTelepon.setText(""+TableviewData.getValueAt(tabelData, 3));
    txtNama.requestFocus();
    btnSimpan.setEnabled(false);
    btnUbah.setEnabled(true);
    btnHapus.setEnabled(true);
}
```

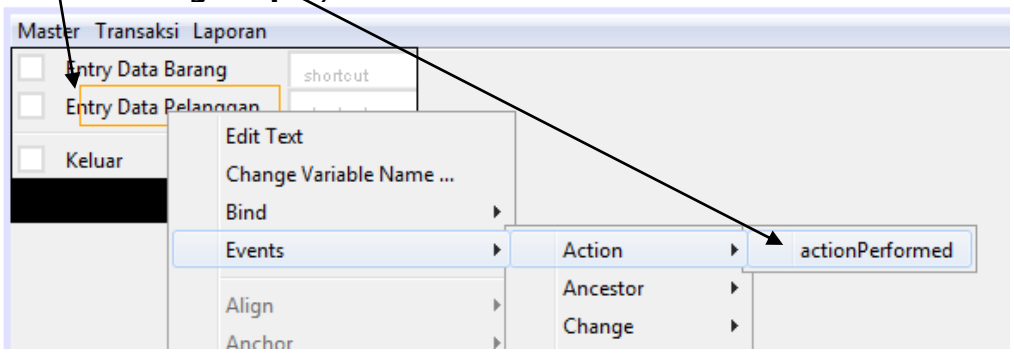
24) Secara Keseluruhan form Entry Data Pelanggan sudah selesai dibuat, Project silahkan disimpan ulang /disave ( .

i. Integrasi Form Entry Data Pelanggan dengan Menu Utama.
Proses ini bertujuan agar form entry data barang dapat digunakan apabila project sudah dijalankan. Seperti yang ditunjukkan pada proses pembuatan Form Entry Data Barang, form ini menggunakan JInternalFrame sebagai framenya dan mengakibatkan form ini tidak mempunyai methode **main()**, sehingga apabilaform akan dijalankan, maka harus merujuk pada form lain yang mempunyai methode main(). Dalam hal ini form yang mempunyai methode **main()** adalah menu utama sebagai induk dari project ini. Adapun langkah-langkahnya sebagai berikut :

1) Buka form **menu utama**. Form dapat dibuka dengan cara mendouble klik form tersebut pada jendela **project**. Pastikan tabulasi **design** dalam keadaan terpilih.

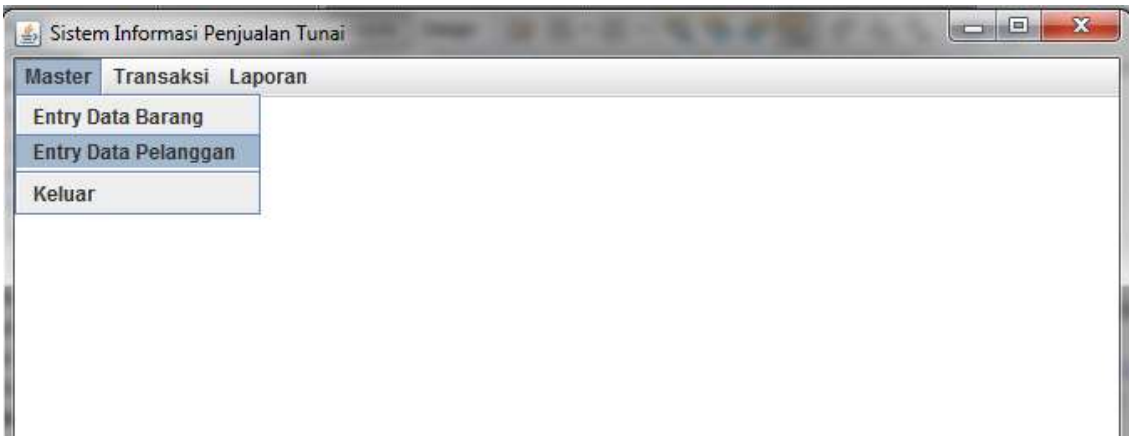


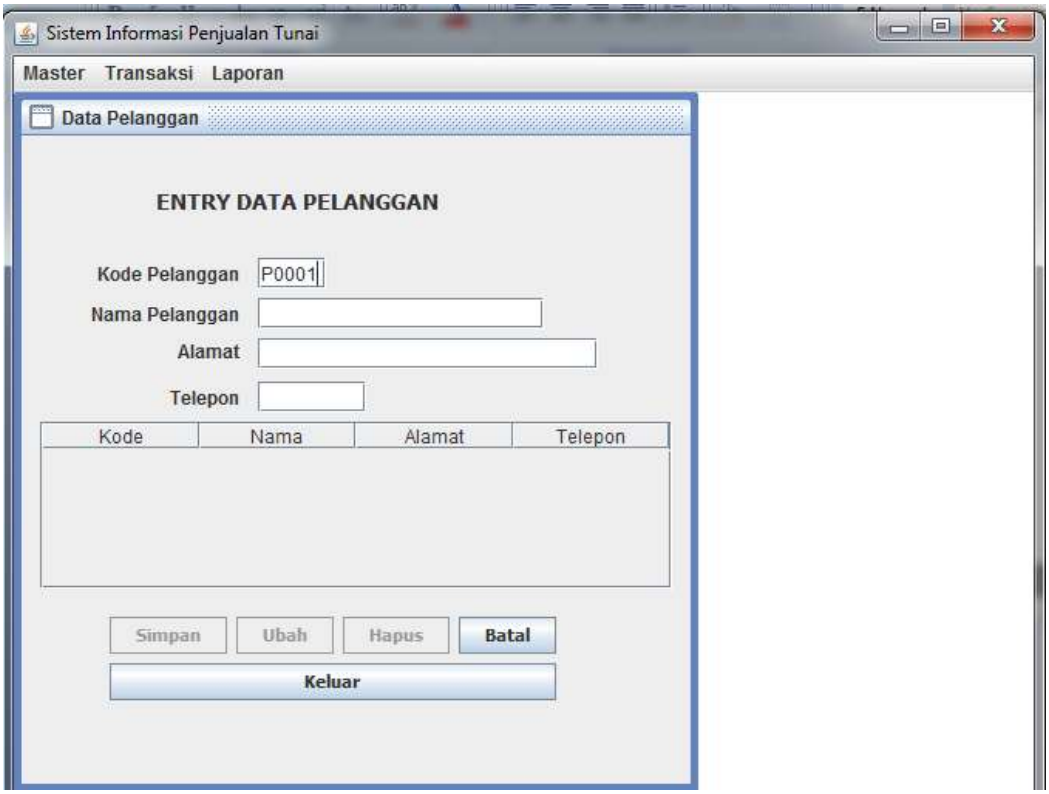
2) Klik tabulasi **design**. Tambahkan perintah pada sub menu **Entry Data Pelanggan** untuk menghubungkan ke class/form data barang. Perintah ditambahkan dengan cara klik kanan pada sub menu tersebut dan melalui event **ActionPerformed**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).



```
private void frmPelangganActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    frmPelanggan pelanggan = new frmPelanggan();

    dskPane.add(pelanggan);
    pelanggan.setVisible(true);
}
```

3) Integrasi form data pelanggan dengan menu utama sudah selesai. Untuk melihat hasil programnya, pastikan **menu utama** dalam keadaan terpilih, kemudian klik **Run** dan pilih **Run File** pada netBeans atau dengan tombol **shift+F6**. Berikut hasil programnya :

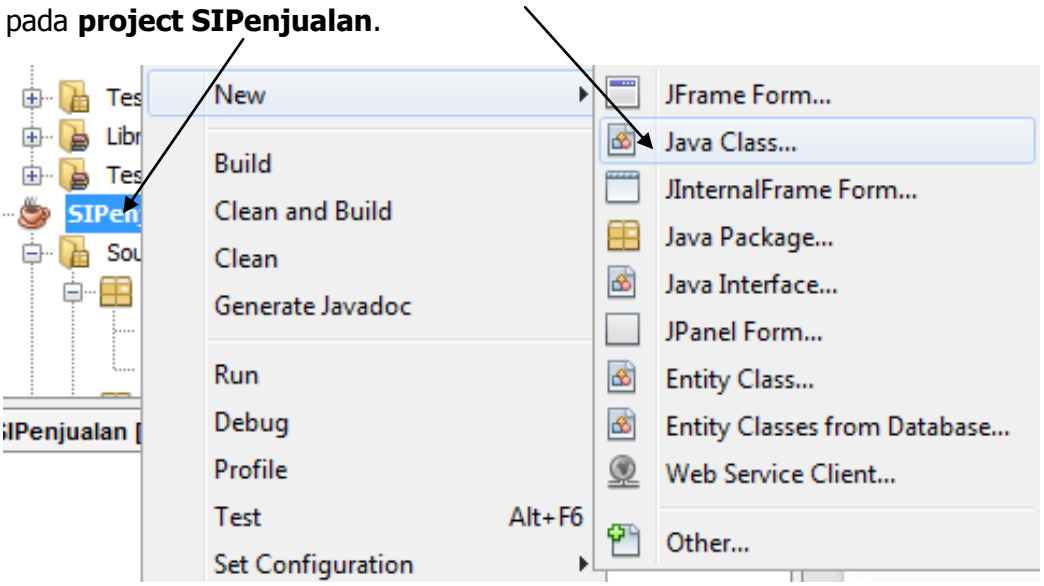




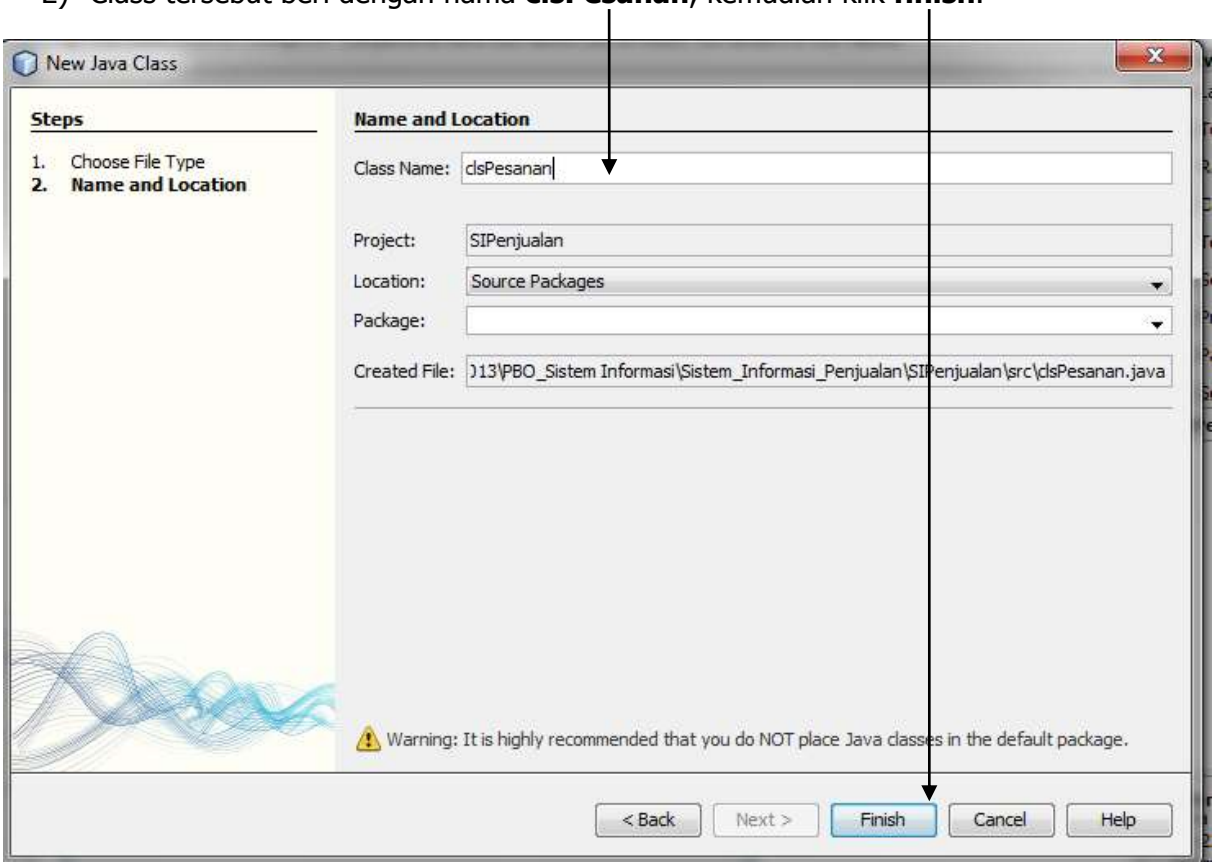
j. Buat Class Pesanan

Adapun langkah-langkahnya sebagai berikut :

1) Tambahkan sebuah class baru (**Java Class**) dengan cara mengklik kanan pada **project SIPenjualan**.



2) Class tersebut beri dengan nama **clsPesanan**, kemudian klik **finish**.



3) Pada Class tersebut, tambahkan sintaks untuk mengimport beberapa library yang diperlukan (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
public class clsPesanan {

}
```

4) Didalam class **clsPesanan** tersebut, tambahkan perintah untuk membuat variabel/atribut dan methode yang diperlukan untuk menentukan atau mengirimkan nilai dari atau ke variabel tersebut (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
public class clsPesanan {
    protected String nopsn, kodepel, kodebrg;
    protected int jumlah, harga, flag;
    protected Date tanggal;

    public void setNoPesan(String N)
    { nopsn = N; }
    public void setTanggal(Date Tg)
    { tanggal = Tg;}
    public void setKodePelanggan(String kdp)
    { kodepel = kdp;}
    public void setKodeBarang(String kdb)
    { kodebrg = kdb;}
    public void setJumlah(int jml)
    { jumlah = jml;}
    public void setHarga(int hr)
    { harga = hr;}
    public String getNoPesan()
    { return(nopsn);}
    public Date getTanggal()
    { return(tanggal);}
    public String getKodePelanggan()
    { return(kodepel);}
    public String getKodeBarang()
    { return(kodebrg);}
    public int getJumlah()
    { return(jumlah);}
    public int getHarga()
    { return(harga);}

    public void setFlag(int F)
    { flag=F;}
    public int getFlag()
    { return(flag);}
}
```

5) Didalam class **clsPesanan** tersebut, tambahkan methode **simpanPesanan()** yang berfungsi untuk menyimpan data ke database. Methode ditambahkan dibawah metode **getFlag()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
public int getFlag()
{ return(flag);}

public void simpanPesanan()
{
    try
    {
        Koneksi k=new Koneksi();
        Connection cn=k.openKoneksi();
        Statement st=cn.createStatement();
        String sql="insert into pesanan values(";
               sql+="'"+getNoPesan()+"','"+getTanggal()+"',";
               sql+="'"+getKodePelanggan()+"')";

           st.executeUpdate(sql);
           setFlag(1);
           st.close();
           cn.close();
           JOptionPane.showMessageDialog(null, "Data Berhasil Simpan",
                   "SIMPAN",JOptionPane.INFORMATION_MESSAGE);
    }
    catch(SQLException sqe)
    {
        JOptionPane.showMessageDialog(null, "Gagal Simpan",
                "GAGAL SIMPAN",JOptionPane.ERROR_MESSAGE);
    }
}
}
```

6) Didalam class **clsPesanan** tersebut, tambahkan methode **simpanDetailPesan()** yang berfungsi untuk menyimpan data ke database. Methode ditambahkan dibawah metode **simpanPesanan()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
            JOptionPane.showMessageDialog(null, "Gagal Simpan",
                    "GAGAL SIMPAN",JOptionPane.ERROR_MESSAGE);
        }
    }

    public void simpanDetailPesan()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="insert into detailpesan values(";
                   sql+="'"+getNoPesan()+"','"+getKodeBarang()+"',";
                   sql+="'"+getJumlah()+"','"+getHarga()+"')";

              st.executeUpdate(sql);
              setFlag(2);
              st.close();
              cn.close();
        }
        catch(SQLException sqe)
        {
            JOptionPane.showMessageDialog(null, "Gagal Simpan",
                    "GAGAL SIMPAN",JOptionPane.ERROR_MESSAGE);
        }
    }
}
```

7) Didalam class **clsPesanan** tersebut, tambahkan methode **updateStok()** yang berfungsi untuk mengupdate stok barang yang ada data di database. Methode ditambahkan dibawah metode **simpanDetailPesan()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
            JOptionPane.showMessageDialog(null, "Gagal Simpan",
                    "GAGAL SIMPAN",JOptionPane.ERROR_MESSAGE);
        }
    }

    public void updateStok()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="update barang set stok=stok-'"+getJumlah()+"' ";
                   sql+="where kdbrg='"+getKodeBarang()+"'";
              st.executeUpdate(sql);
              st.close();
              cn.close();
        }
        catch(SQLException sqe)
        { }
    }
}
```

Perhatikan : antara petik tunggal dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh :
“” (SALAH),
‘ ” (BETUL).

8) Didalam class **clsPesanan** tersebut, tambahkan methode **autoNoPsn()** yang berfungsi untuk menampilkan nomor pesan secara otomatis. Methode ditambahkan dibawah metode **updateStok()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
        catch(SQLException sqe)
        { }
    }
```

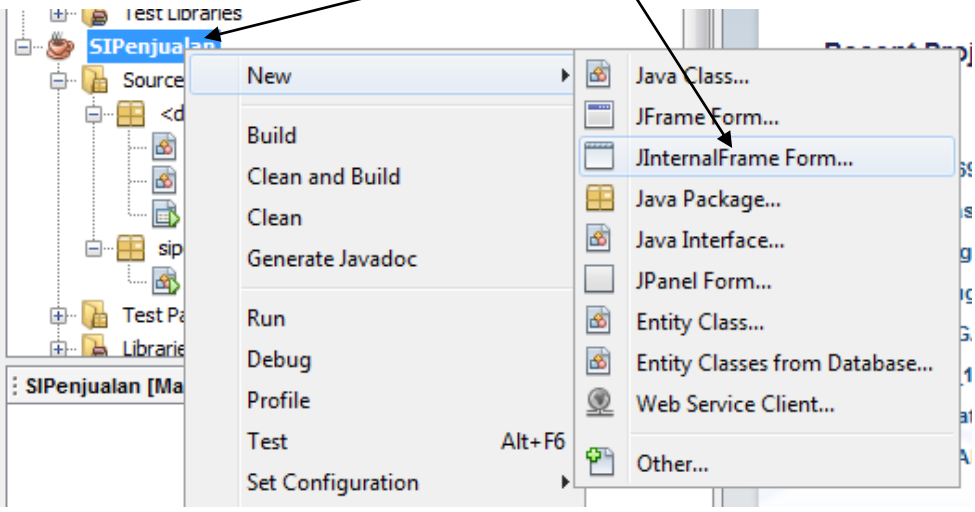
```
    public void autoNoPsn()
    {
        try
        {
            int hit=0;

            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select count(nopesan) from pesanan";
            ResultSet rs=st.executeQuery(sql);

            if(rs.next())
            {
                if(Integer.parseInt(rs.getString(1))==0)
                {
                    setNoPesan("PS001");
                    st.close();
                    cn.close();
                }
                else
                {
                    sql="select Max(mid(nopesan,3,3)) from pesanan";
                    rs=st.executeQuery(sql);
                    rs.next();
                    hit=(Integer.parseInt(rs.getString(1)))+1;
                    if (hit<10)
                    { setNoPesan("PS00"+hit);}
                    else if (hit<100)
                    { setNoPesan("PS0"+hit);}
                    else
                    { setNoPesan("PS"+hit);}
                    st.close();
                    cn.close();
                }
            }
        }
        catch(SQLException sqe)
        {}
    }
}
```

9) Secara keseluruhan class **clsPesanan** sudah selesai dibuat. Project dapat disimpan ulang/disave ().

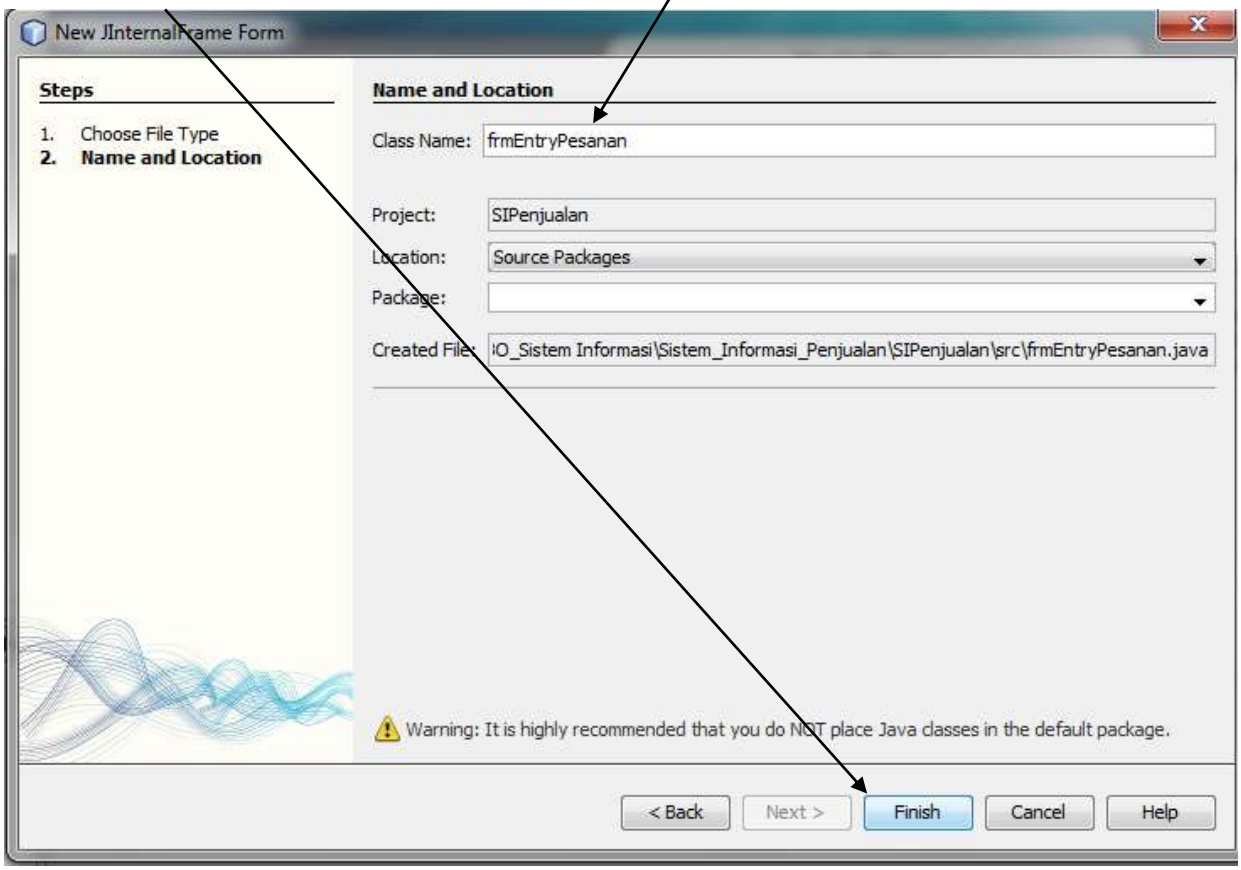
k. Buat Form Entry Pesanan

Langkah-langkahnya sebagai berikut.

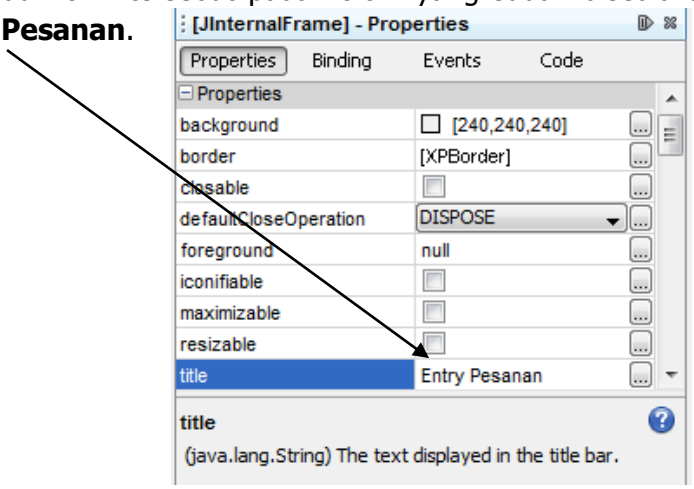
1) Tambahkan sebuah form baru (**JInternalFrame Form**), dengan cara mengklik kanan pada project **SIPenjualan**.



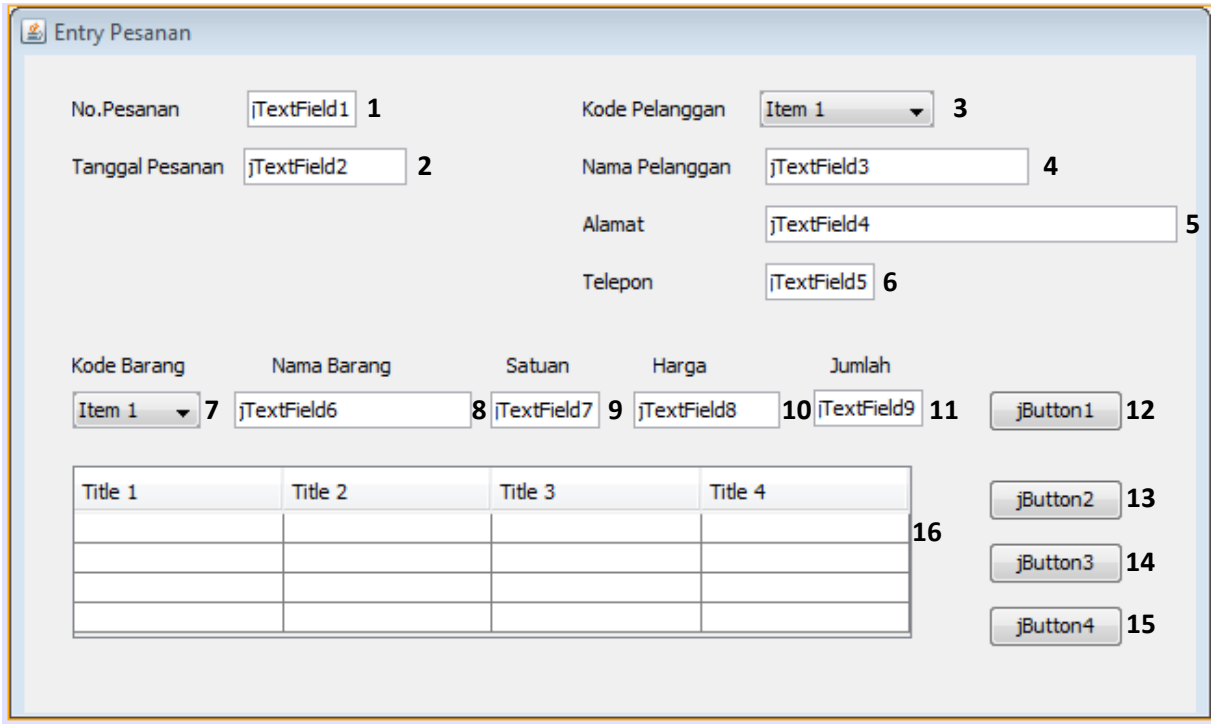
2) Form baru tersebut beri dengan nama **frmEntryPesanan**, kemudian klik **Finish**.

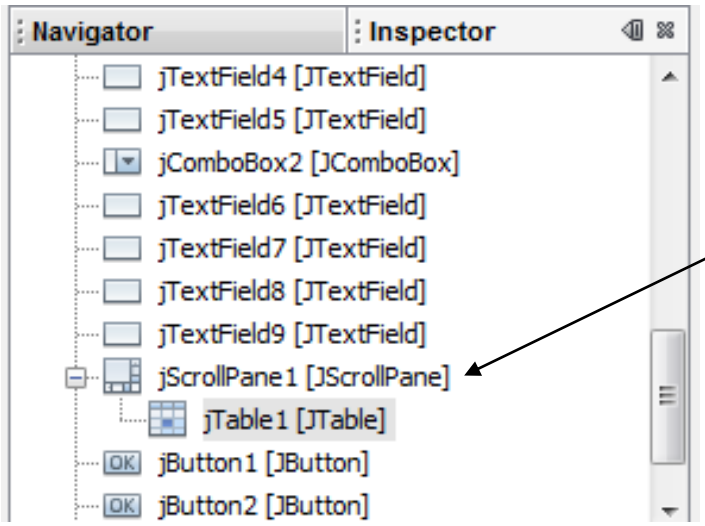


3) Beri judul form tersebut pada kolom yang sudah disediakan dengan nama : **Entry Pesanan**.



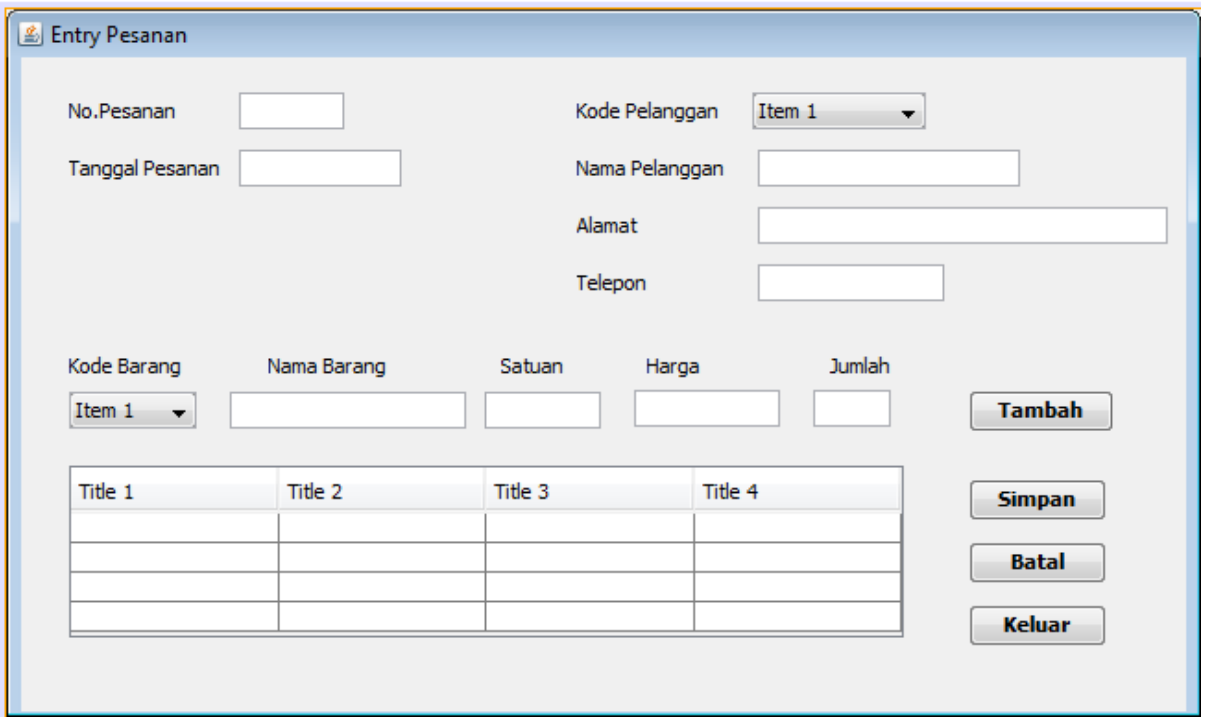
4) Atur ukuran form dan tambahkan objek seperti gambar berikut :





5) Berdasarkan form tersebut, ubah nama text dan nama variabelnya sesuai dengan penomorannya sebagai berikut :
   - **1**. **jTextField1**, Text dihapus, change variable name = **txtNoPsn**.
   - **2**. **jTextField2**, Text dihapus, change variable name = **txtTglPsn**.
   - **3**. **jComboBox1**, change variable name = **cmbKdPlg**.
   - **4**. **jTextField3**, Text dihapus, change variable name= **txtNmPlg**.
   - **5**. **jTextField4**, Text dihapus, change variable name = **txtAlmPlg**.
   - **6**. **jTextFIeld5**, Text dihapus, change variable name= **txtTlpPlg**.
   - **7**. **jComboBox2**, change variable name = **cmbKdBrg**.
   - **8**. **jTextField6**, Text dihapus, change variable name = **txtNmBrg**.
   - **9**. **jTextField7**, Text dihapus, change variable name = **txtSatuan**.
   - **10**. **jTextField8**, Text dihapus, change variable name = **txtHarga**.
   - **11**. **jTextField9**, Text dihapus, change variable name = **txtJumlah**.
   - **12**. **jButton1**, Text = **Tambah**, change variable name= **btnTambah**.
   - **13**. **jButton2**, Text = **Simpan**, change variable name=**btnSimpan**.
   - **14**. **jButton3**, Text = **Batal**, change variable name = **btnBatal**.
   - **15**. **jButton4**, Text = **Keluar**, change variable name = **btnKeluar**.
   - **16**. **jTable1**, change variable name=**viewDataPesan**.
   - **17**. **jScrollPane1**, change variable name= **scrPane**.

   Sehingga tampilan akhir form sebagai berikut :



6) Klik tabulasi **source** dan tambahkan perintah berikut untuk mengimport beberapa library yang diperlukan dari packagenya (**Sintaks program yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmEntryPesanan extends javax.swing.JInternalFrame {

    /** Creates new form frmEntryPesanan */
    public frmEntryPesanan() {
        initComponents();
    }
```

7) Tambahkan perintah pada bagian deklarasi class dan konstruktor class dengan perintah berikut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmEntryPesanan extends javax.swing.JInternalFrame {

    /** Creates new form frmEntryPesanan */
    DefaultTableModel tabMode;

    public frmEntryPesanan() {
        initComponents();

        String [] row={"Kode","Nama","Harga","Jumlah","Total"};
        tabMode = new DefaultTableModel(null,row);

        viewDataPesan.setModel(tabMode);
        scrPane.getViewport().add(viewDataPesan,null);
    }

    /** This method is called from within the constructor to
     * initialize the form.
```

8) Tambahkan methode **kosong**() dibawah konstruktor class tersebut. Methode tersebut berfungsi untuk membersihkan komponen dari data yang telah dinput/diolah sebelumnya sekaligus sebagai langkah awal untuk proses berikutnya. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
Generated Code
public void kosong()
{
    clsPesanan psn=new clsPesanan();
    int baris=tabMode.getRowCount();
    for (int i=0;i<baris;i++)
    {
        tabMode.removeRow(0);
    }
    txtNoPsn.setText("");
    txtTglPsn.setText("");
    cmbKdPlg.removeAllItems();
    txtNmPlg.setText("");
    txtAlmPlg.setText("");
    txtTlpPlg.setText("");
    cmbKdBrg.removeAllItems();
    txtNmBrg.setText("");
    txtSatuan.setText("");
    txtHarga.setText("");
    txtJumlah.setText("");
    btnSimpan.setEnabled(false);
    btnTambah.setEnabled(false);
    psn.autoNoPsn();
    txtNoPsn.setText(psn.getNoPesan());
}

// Variables declaration - do not modify
private javax.swing.JButton btnBatal;
private javax.swing.JButton btnKeluar;
private javax.swing.JButton btnSimpan;
private javax.swing.JButton btnTambah;
private javax.swing.JComboBox cmbKdBrg;
private javax.swing.JComboBox cmbKdPlg;
```

9) Tambahkan methode **comboPelanggan()** yang berfungsi untuk menampilkan data ke objek yang sudah disiapkan di form. Methode ini ditambahkan dibawah methode **kosong()**. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
        psn.autoNoPsn();
        txtNoPsn.setText(psn.getNoPesan());
    }

    public void comboPelanggan()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select kdplg from pelanggan ";
                   sql+="order by kdplg asc";
            ResultSet rs=st.executeQuery(sql);

            while (rs.next())
            {
                cmbKdPlg.addItem(rs.getString("kdplg"));
            }
            st.close();
            cn.close();
        }
        catch(SQLException sqe)
        {}
    }
```

Perhatikan : antara pelanggan dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh : pelanggan” (SALAH), pelanggan ” (BETUL).

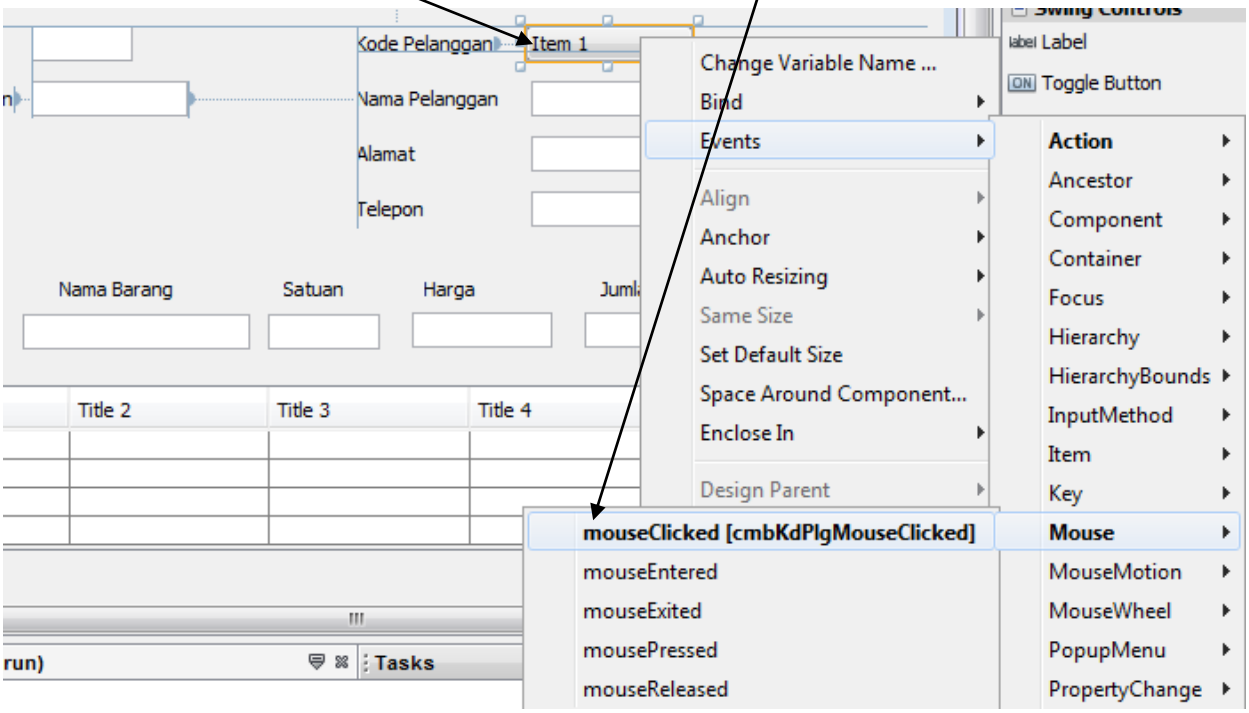
```
    // Variables declaration - do not modify
    private javax.swing.JButton btnBatal;
    private javax.swing.JButton btnKeluar;
```

10) Tambahkan methode **comboBarang()** yang berfungsi untuk menampilkan data ke tabel yang sudah disiapkan di form. Methode ini ditambahkan dibawah methode **comboPelanggan()**. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
    catch(SQLException sqe)
    {}
}

public void comboBarang()
{
    try
    {
        Koneksi k=new Koneksi();
        Connection cn=k.openKoneksi();
        Statement st=cn.createStatement();
        String sql="select kdbrg from barang ";
               sql+="order by kdbrg asc";
        ResultSet rs=st.executeQuery(sql);

        while (rs.next())
        {
            cmbKdBrg.addItem(rs.getString("kdbrg"));
        }
    }
    catch(SQLException sqe)
    {}
}

// Variables declaration - do not modify
private javax.swing.JButton btnBatal;
private javax.swing.JButton btnKeluar;
private javax.swing.JButton btnSimpan;
```

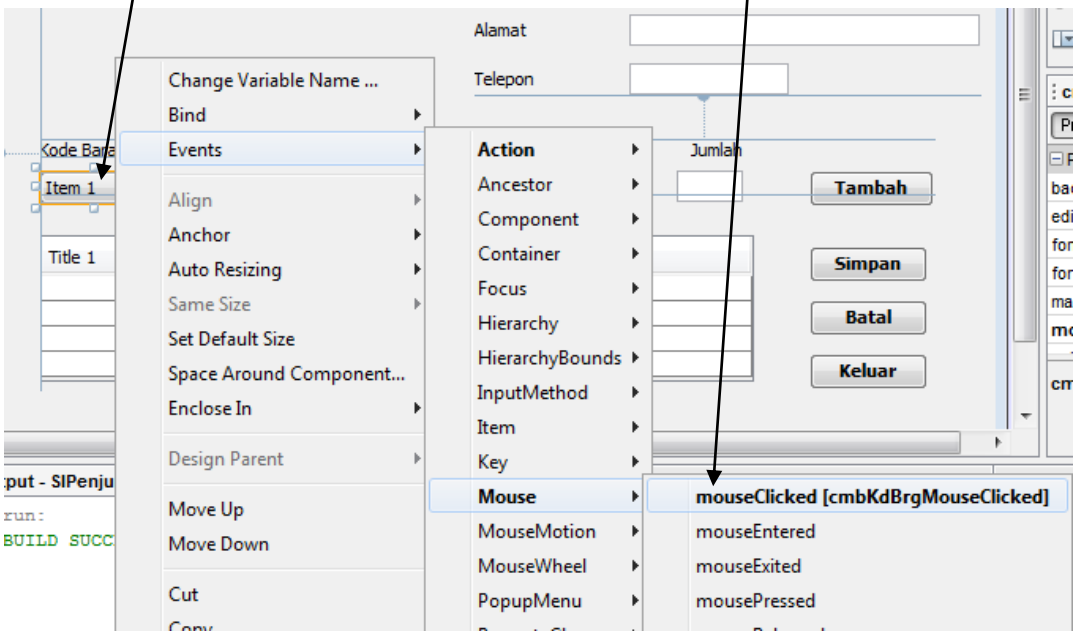
Perhatikan : antara barang dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh :
barang” (SALAH),
barang ” (BETUL).

11) Tambahkan perintah melalui event **mouseClicked** pada object **cmbKdPlg** (combo box kode pelanggan) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.



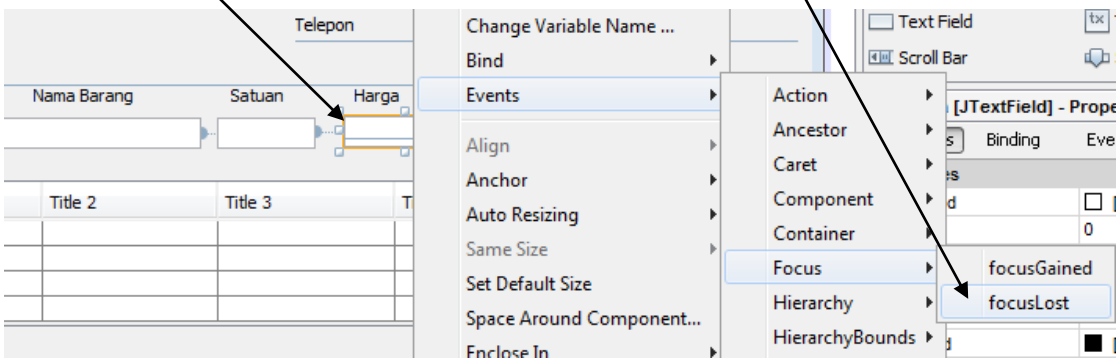

```
private void cmbKdPlgMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    clsPelanggan plg=new clsPelanggan();

    plg.setKode(""+cmbKdPlg.getSelectedItem());
    plg.tampil();
    txtNmPlg.setText(plg.getNama());
    txtAlmPlg.setText(plg.getAlamat());
    txtTlpPlg.setText(plg.getTelepon());
    cmbKdBrg.requestFocus();
}
```

12) Tambahkan perintah melalui event **mouseClicked** pada object **cmbKdBrg** (combo box kode barang) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.



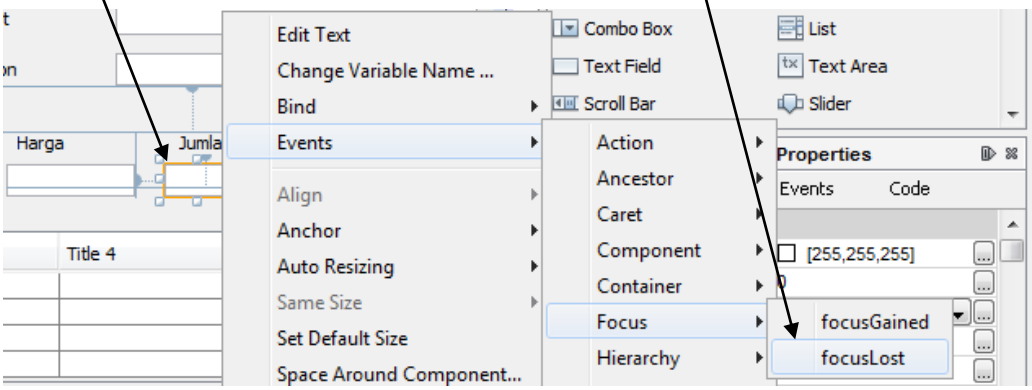

```
private void cmbKdBrgMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    clsBarang brg=new clsBarang();

    brg.setKode(""+cmbKdBrg.getSelectedItem());
    brg.tampil();
    txtNmBrg.setText(brg.getNama());
    txtSatuan.setText(brg.getSatuan());
    txtHarga.setText(""+brg.getHarga());
    txtHarga.requestFocus();
}
```

13) Tambahkan perintah melalui event **focustLost** pada object **txtHarga** (Text field untuk input harga) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.



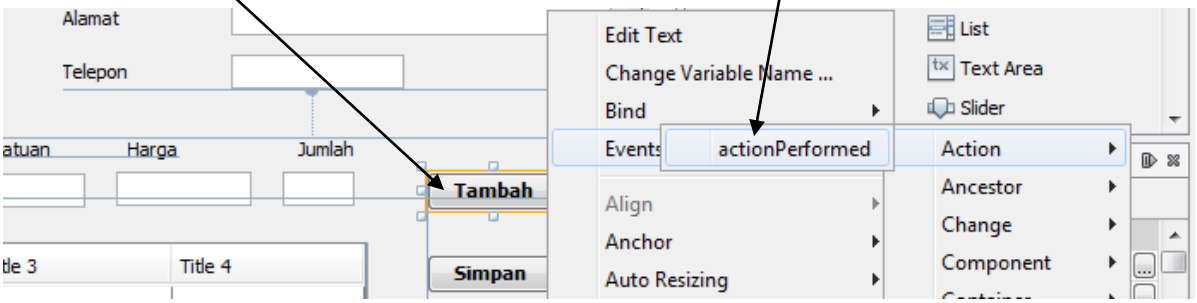

```
private void txtHargaFocusLost(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:
    txtJumlah.requestFocus();
}
```

14) Tambahkan perintah melalui event **focustLost** pada object **txtJumlah** (Text field untuk input jumlah beli) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.



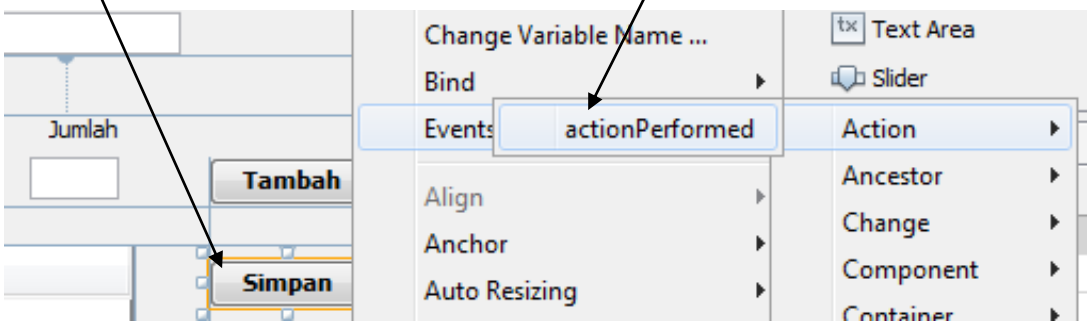

```
private void txtJumlahFocusLost(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:
    btnTambah.setEnabled(true);
    btnTambah.requestFocus();
}
```

15) Tambahkan perintah melalui event **actionPerformed** pada object **btnTambah** (Tombol **Tambah**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.



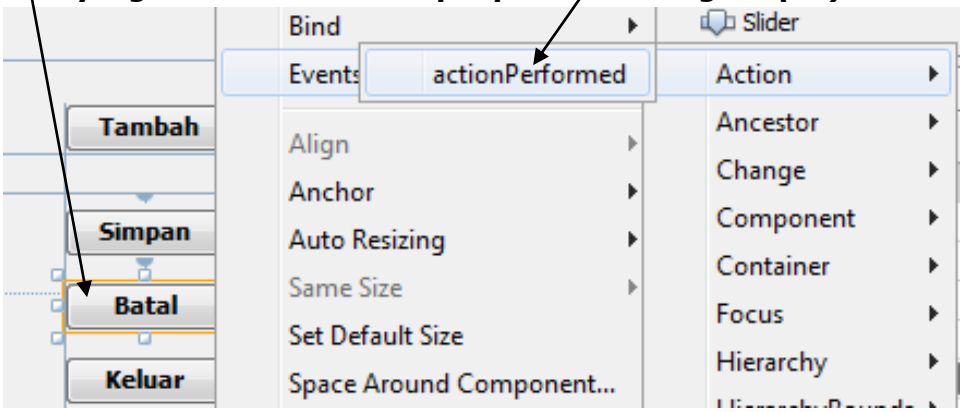

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String kd, nm, hr, jm, tt, cekkode;
    int hit=0;
    int baris=tabMode.getRowCount();

    kd=""+cmbKdBrg.getSelectedItem();
    nm=txtNmBrg.getText();
    hr=txtHarga.getText();
    jm=txtJumlah.getText();
    tt=""+(Integer.parseInt(txtHarga.getText())*Integer.parseInt(txtJumlah.getText()));

    for (int i=0;i<baris;i++)
    {
        cekkode=""+viewDataPesan.getValueAt(i, 0);
        if (kd.equals(cekkode))
        {  hit++; }
    }
    if(hit==0)
    {
        String [] data={kd,nm,hr,jm,tt};
        tabMode.addRow(data);

        txtNmBrg.setText("");
        txtSatuan.setText("");
        txtHarga.setText("");
        txtJumlah.setText("");
        btnSimpan.setEnabled(true);
        cmbKdBrg.requestFocus();
        btnTambah.setEnabled(false);
    }
    else
    {
        JOptionPane.showMessageDialog(null, "Kode Barang Sudah Ada !!!",
                "PESANAN",JOptionPane.INFORMATION_MESSAGE);
        cmbKdBrg.requestFocus();
    }
}
```

16) Tambahkan perintah melalui event **actionPerformed** pada object **btnSimpan** (Tombol **Simpan**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.



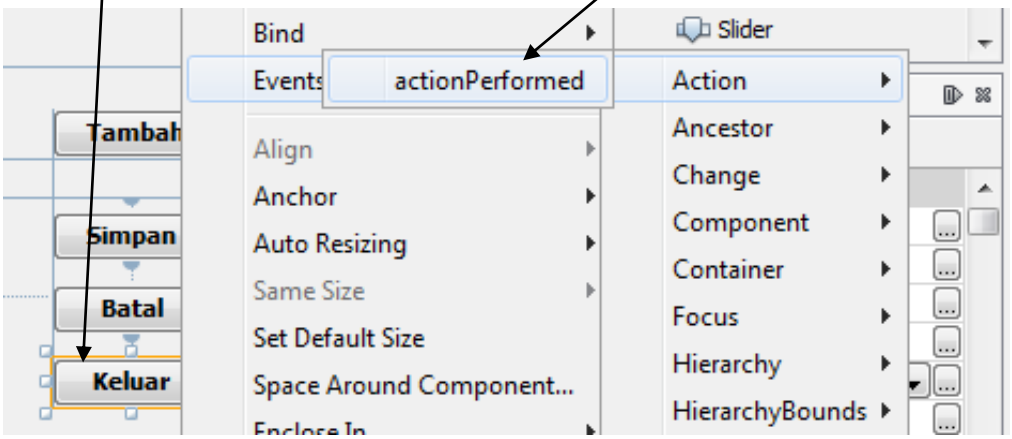

```
private void btnSimpanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clsPesanan psn = new clsPesanan();
    int baris=tabMode.getRowCount();

    for (int i=0;i<baris;i++)
    {
        psn.setNoPesan(txtNoPsn.getText());
        psn.setKodeBarang(""+viewDataPesan.getValueAt(i, 0));
        psn.setHarga(Integer.parseInt(""+viewDataPesan.getValueAt(i, 2)));
        psn.setJumlah(Integer.parseInt(""+viewDataPesan.getValueAt(i, 3)));
        psn.simpanDetailPesan();
        psn.updateStok();
    }

    psn.setNoPesan(txtNoPsn.getText());
    psn.setTanggal(Date.valueOf(txtTglPsn.getText()));
    psn.setKodePelanggan(""+cmbKdPlg.getSelectedItem());
    psn.simpanPesanan();
    kosong();
    comboBarang();
    comboPelanggan();
}
```

17) Tambahkan perintah melalui event **actionPerformed** pada object **btnBatal** (Tombol Batal) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.




```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    kosong();
    comboPelanggan();
    comboBarang();
}
```

18) Tambahkan perintah melalui event **actionPerformed** pada object **btnKeluar** (Tombol Keluar) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.




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

19) Kembali ke konstruktor **frmEntryPesanan**, tambahkan perintah berikut untuk memanggil methode **kosong()**, **comboPelanggan() dan comboBarang()**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**)

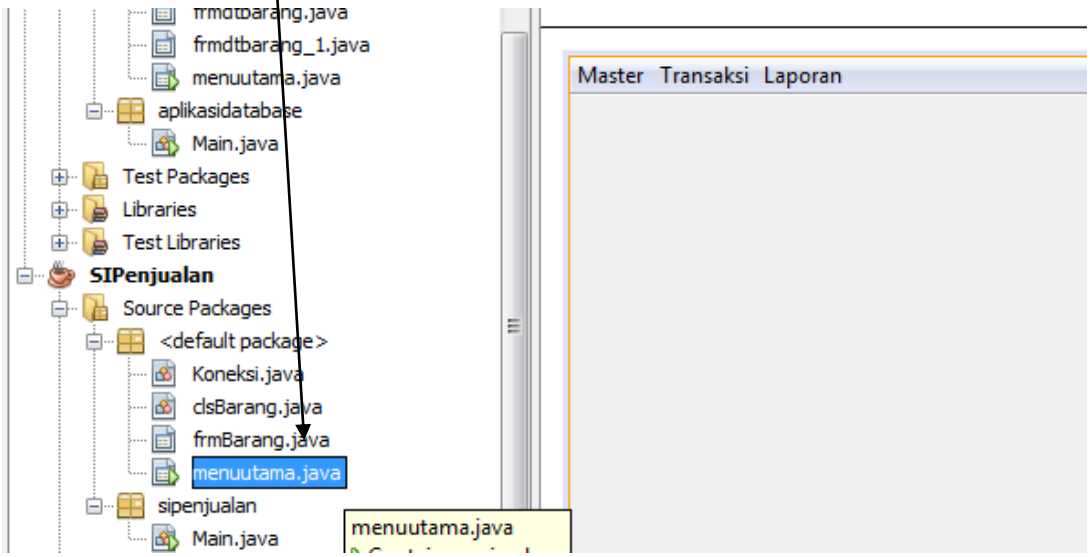
```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class frmEntryPesanan extends javax.swing.JInternalFrame {

    /** Creates new form frmEntryPesanan */
    DefaultTableModel tabMode;

    public frmEntryPesanan() {
        initComponents();

        String [] row={"Kode","Nama","Harga","Jumlah","Total"};
        tabMode = new DefaultTableModel(null,row);

        viewDataPesan.setModel(tabMode);
        scrPane.getViewport().add(viewDataPesan,null);

        kosong();
        comboPelanggan();
        comboBarang();
    }
```

20) Secara Keseluruhan form Entry Data Pesanan sudah selesai dibuat, Project silahkan disimpan ulang /disave ()

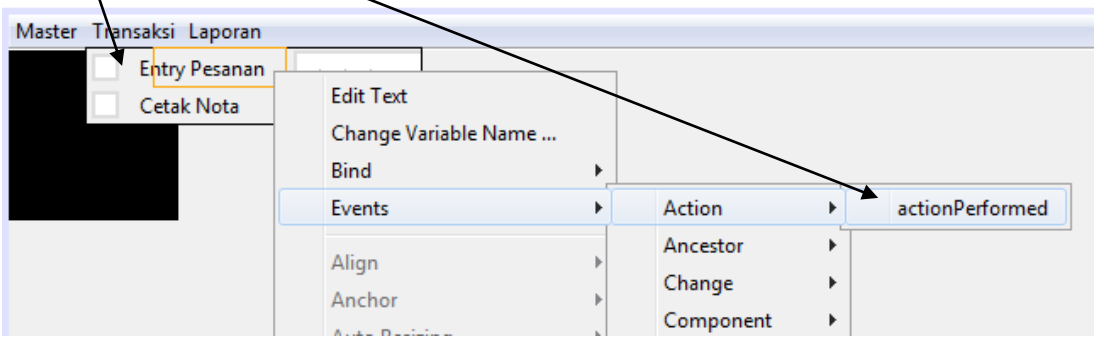
I. Integrasi Form Entry Pesanan dengan Menu Utama.

Proses ini bertujuan agar form entry data pesanan dapat digunakan apabila project sudah dijalankan. Seperti yang ditunjukkan pada proses pembuatan Form Entry Data Barang, form ini menggunakan JInternalFrame sebagai framenya dan mengakibatkan form ini tidak mempunyai methode **main()**, sehingga apabilaform akan dijalankan, maka harus merujuk pada form lain yang mempunyai methode main(). Dalam hal ini form yang mempunyai methode **main()** adalah menu utama sebagai induk dari project ini. Adapun langkah-langkahnya sebagai berikut :

1) Buka form **menu utama**. Form dapat dibuka dengan cara mendouble klik form tersebut pada jendela **project**. Pastikan tabulasi **design** dalam keadaan terpilih.

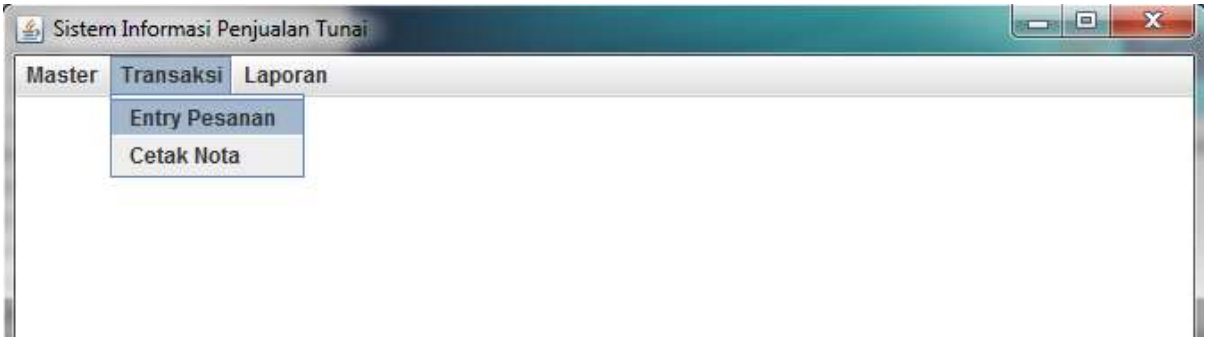


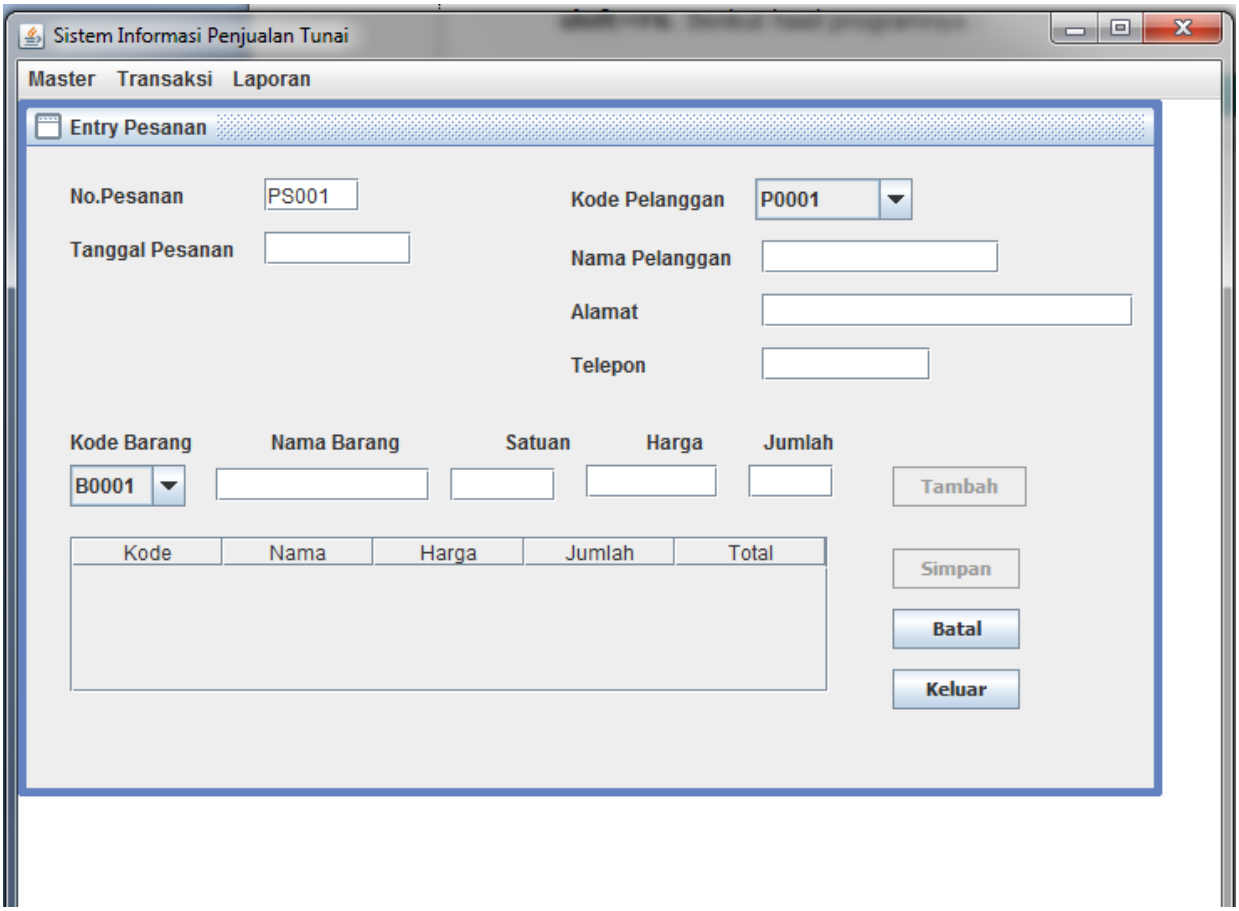
2) Klik tabulasi **design**. Tambahkan perintah pada sub menu **Entry Pesanan** untuk menghubungkan ke class/form data pesanan. Perintah ditambahkan dengan cara klik kanan pada sub menu tersebut dan melalui event **actionPerformed**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).




```
private void frmPesananActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    frmEntryPesanan frmpsn=new frmEntryPesanan();

    dskPane.add(frmpsn);
    frmpsn.setVisible(true);
}
```

3) Integrasi form data pesanan dengan menu utama sudah selesai. Untuk melihat hasil programnya, pastikan **menu utama** dalam keadaan terpilih, kemudian klik **Run** dan pilih **Run File** pada netBeans atau dengan tombol **shift+F6**. Berikut hasil programnya :





4. Buat report pada ireport

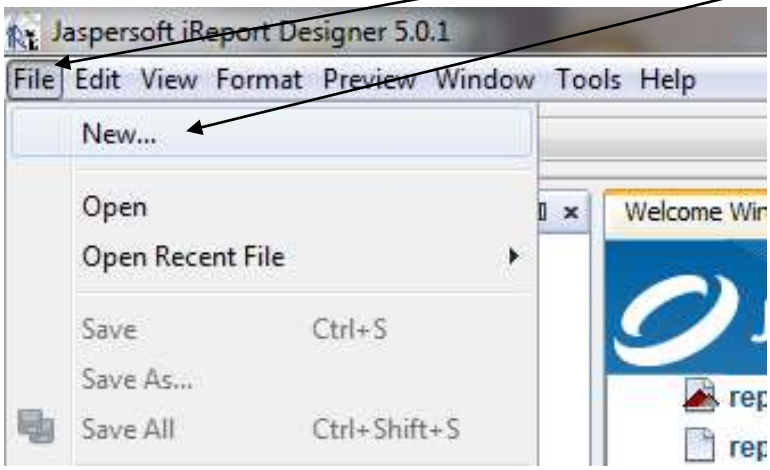
Pada aplikasi Sistem Informasi Penjualan Tunai ini, report yang akan dibuat ada dua macam, yaitu **NOTA**, sebagai bukti pembayaran dari hasil transaksi pesanan barang dan **LAPORAN PENJUALAN**, sebagai laporan jumlah penjualan yang telah terjadi dalam periode tertentu.

a. Buat NOTA

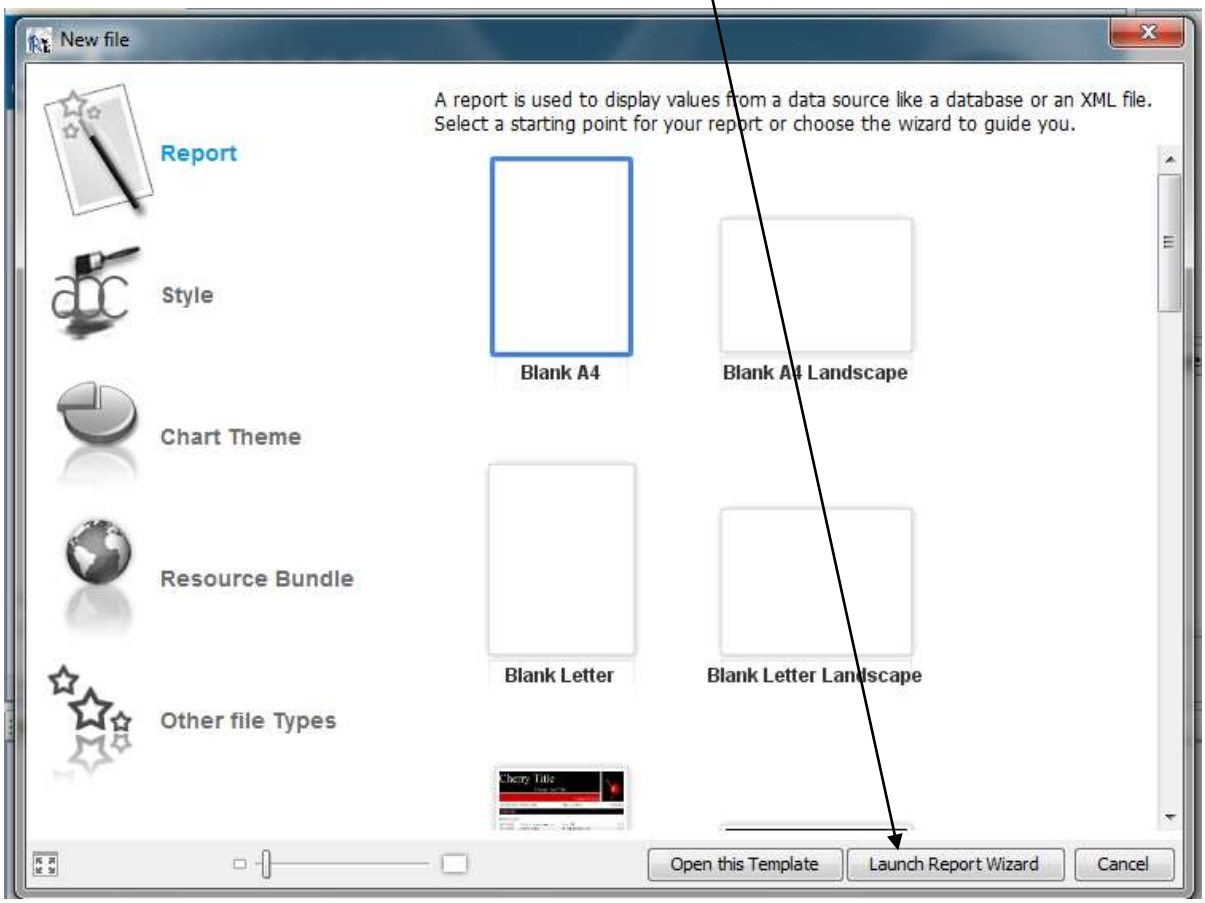
1) Buka aplikasi iReport (

dan tampilan awalnya sebagai berikut :

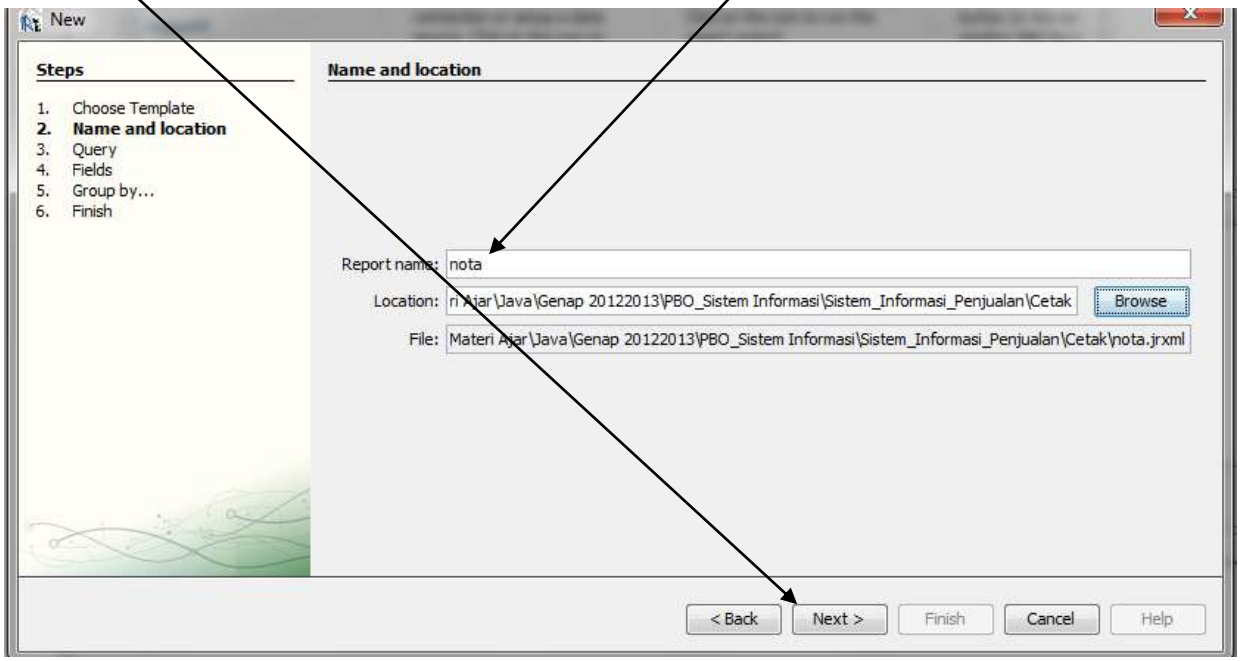
2) Buka report baru dengan cara mengklik menu **File** dan sub menu **New**.



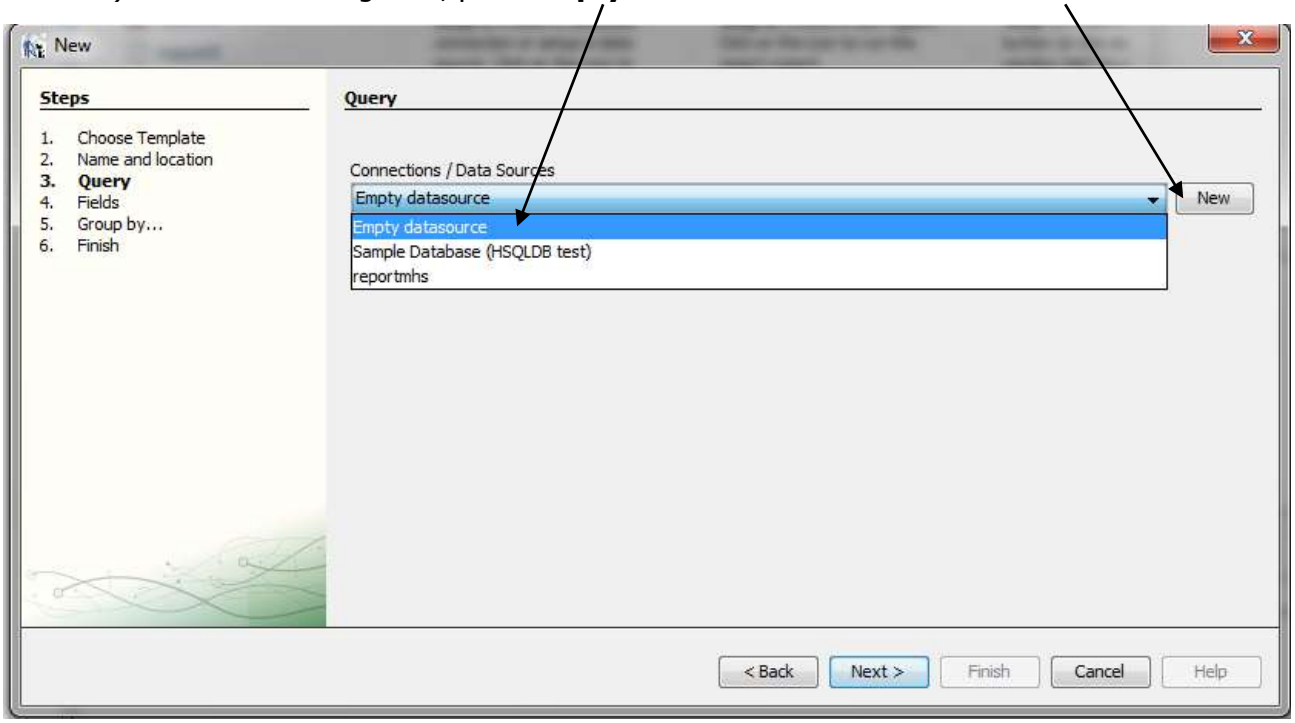
3) Pada kotak dialog **New file**, klik **Launch Report Wizard**.



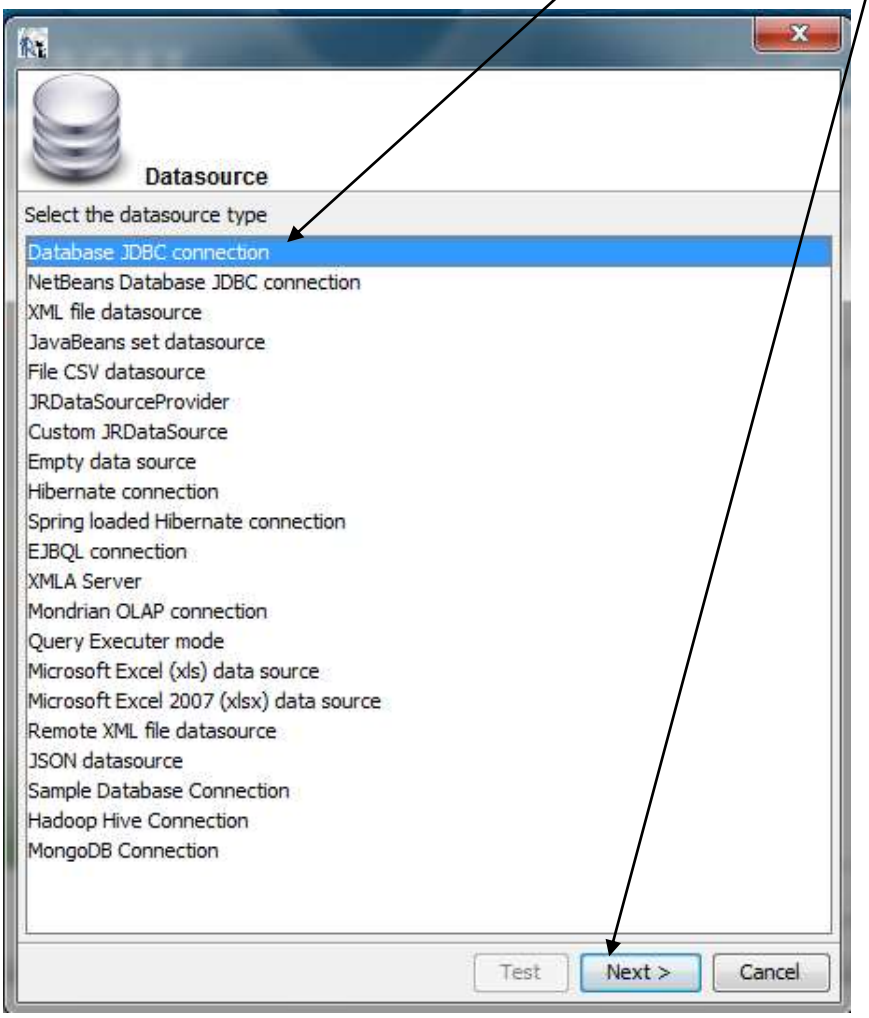
4) Tentukan nama dan lokasi penyimpanan reportnya sesuai dengan keinginan. Pada contoh aplikasi ini, file diberi dengan nama : **nota**. Kemudian klik tombol **next**.



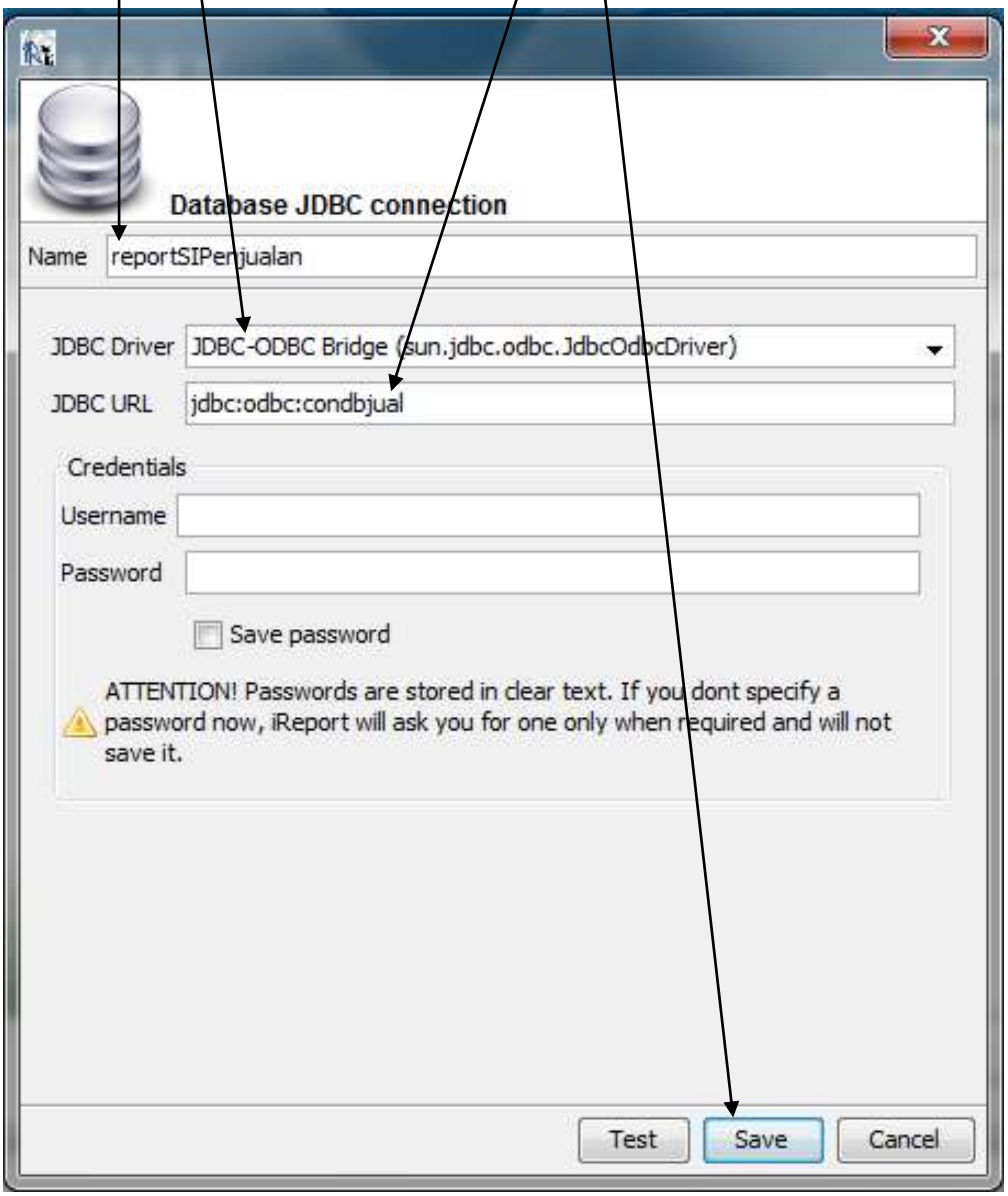
5) Pada kotak dialog New, pilih **Empty datasource** . Kemudian klik **New**.



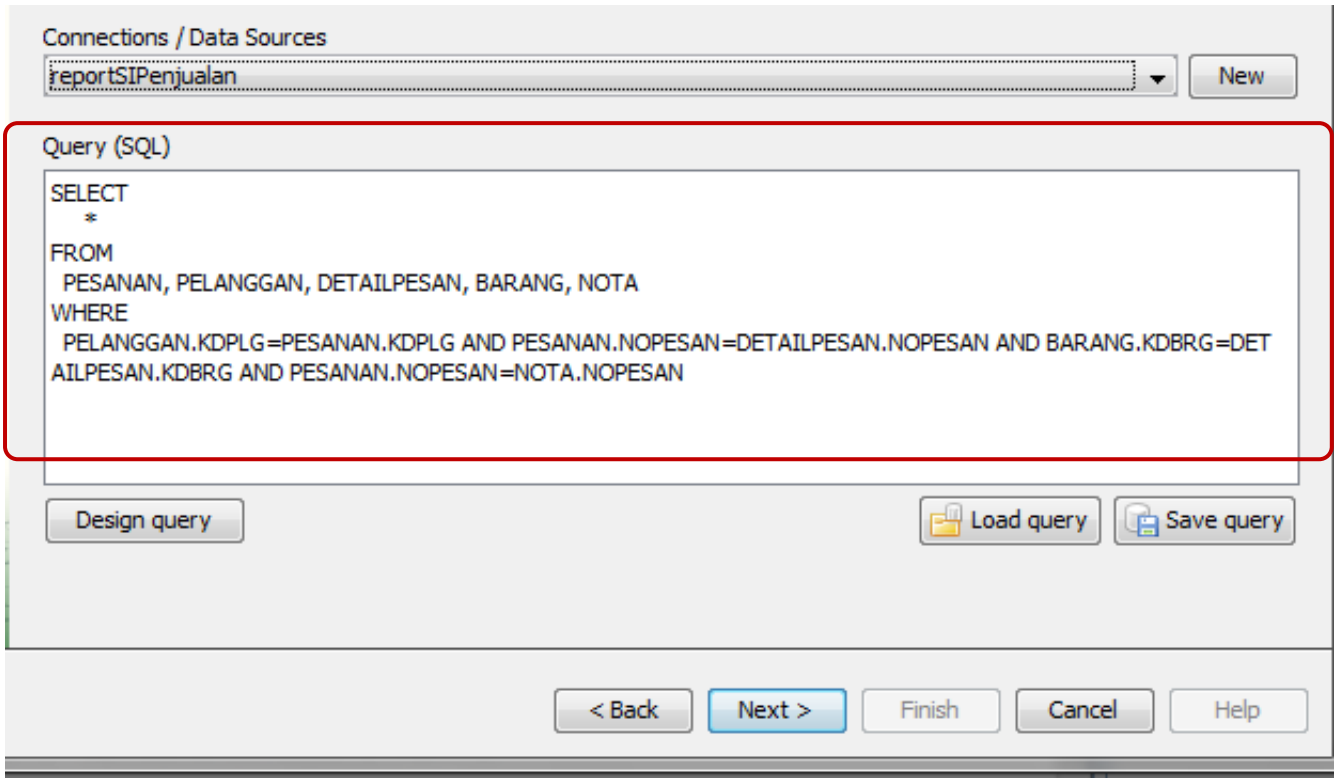
6) Pada Kotak Dialog **DataSource**, pastikan **Database JDBC connection** dalam keadaan terpilih, kemudian dilanjukan dengan mengklik tombol **Next** .



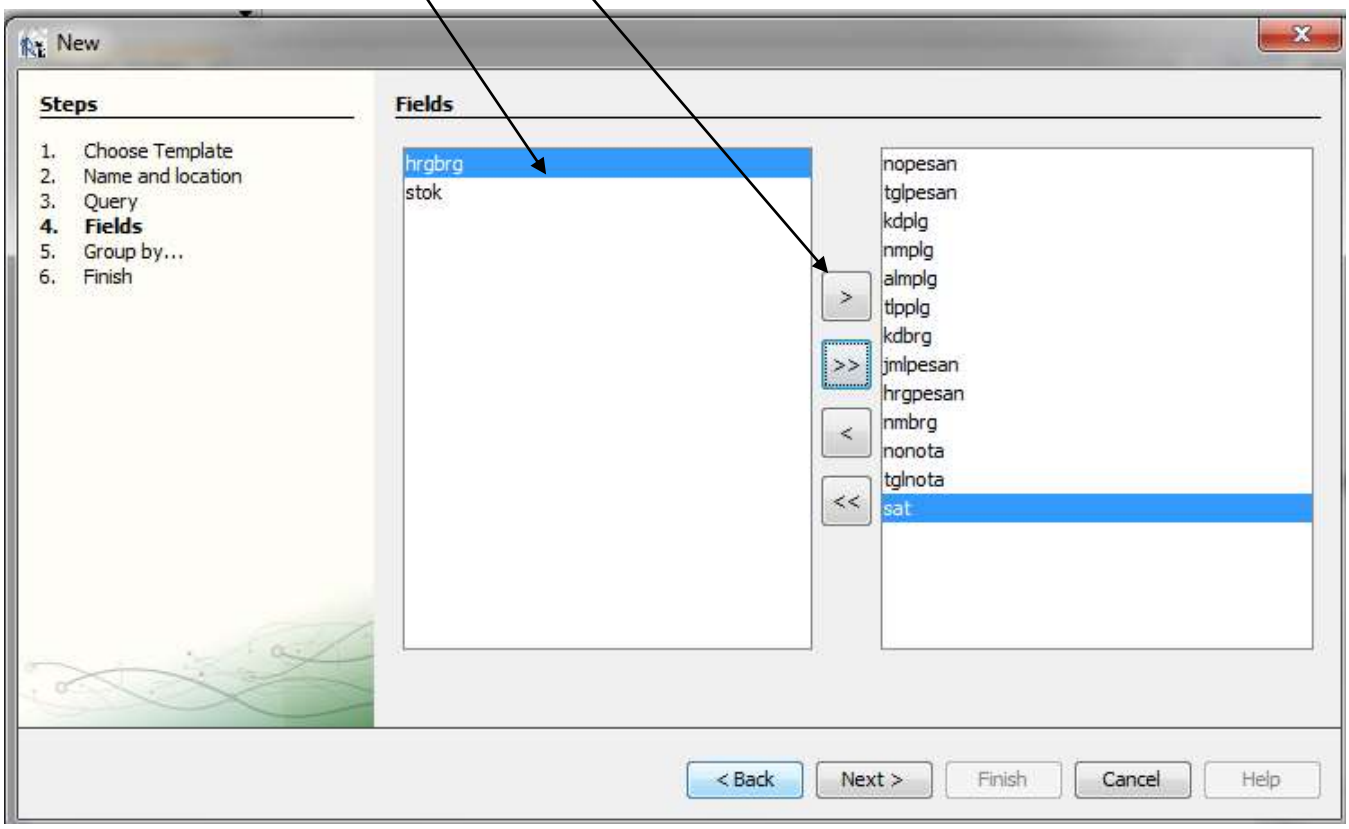
7) Pada kotak dialog Database JDBC connection, buat nama koneksinya dengan nama : **reportSIPenjualan**. Pastikan pada JDBC Driver terpilih Driver : **JDBC-ODBC Bridge (sun.jdbc.odbc.JdbcOdbcDriver)**. Pada JDBC URL buat nama databasenya sesuai dengan yang dibuat pada ODBC Control Panel atau sesuai dengan class Koneksi.java yang dibuat pada NetBeans. Pada contoh ini nama koneksi databasenya : **condbjual**. Username dan Password biarkan saja kosong, kemudian klik tombol **save**.



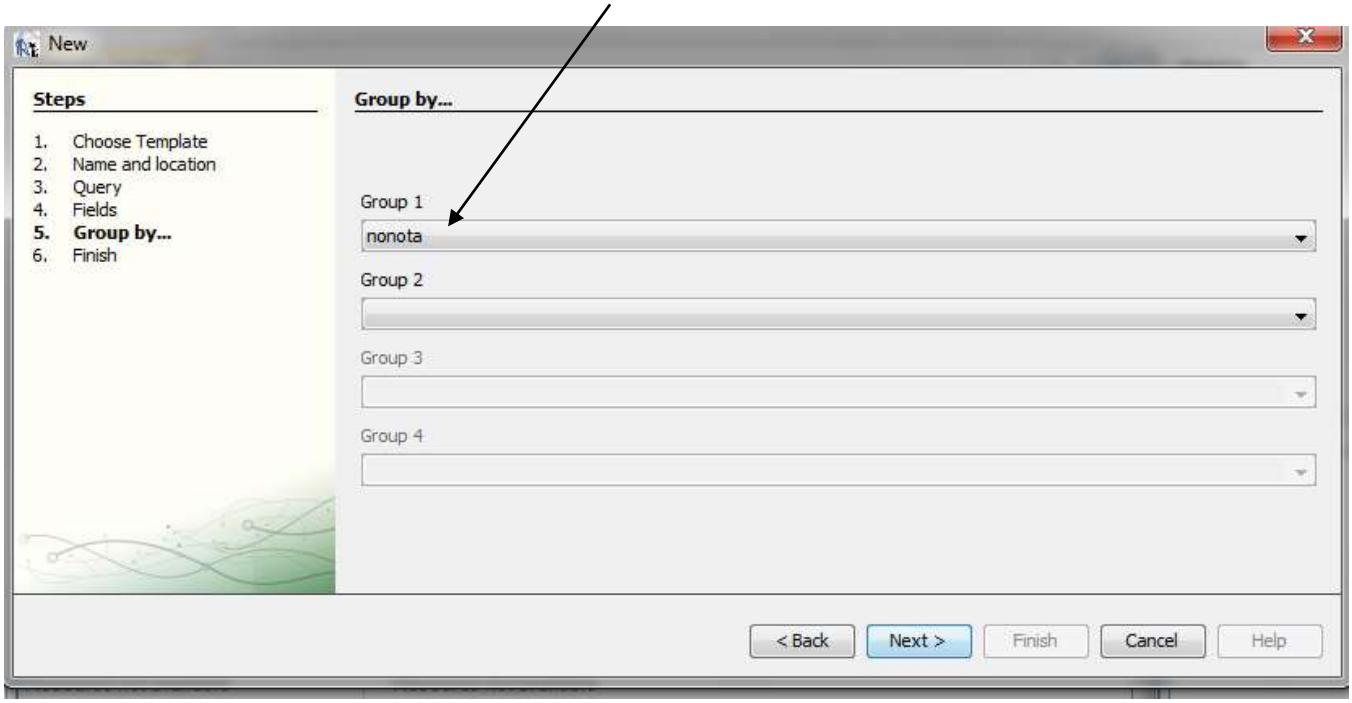
8) Kembali pada kotak dialog New. Ketik perintah berikut pada **Query (SQL)**. Untuk melanjutkan ke langkah-langkah berikutnya klik **Next**.



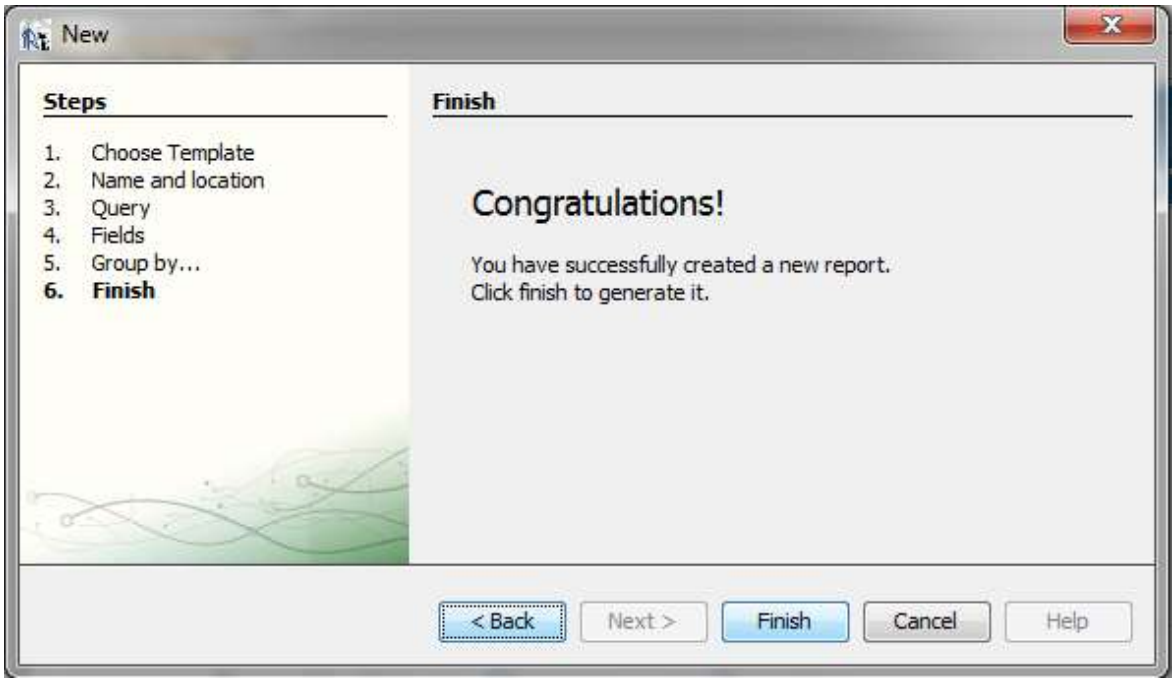
9) Pilih field-field yang diperlukan untuk ditampilkan direport dengan cara mengklik **field** dan **tombol** yang sudah disediakan. Field-field yang diperlukan seperti yang ditampilkan. Kemudian klik **Next.**



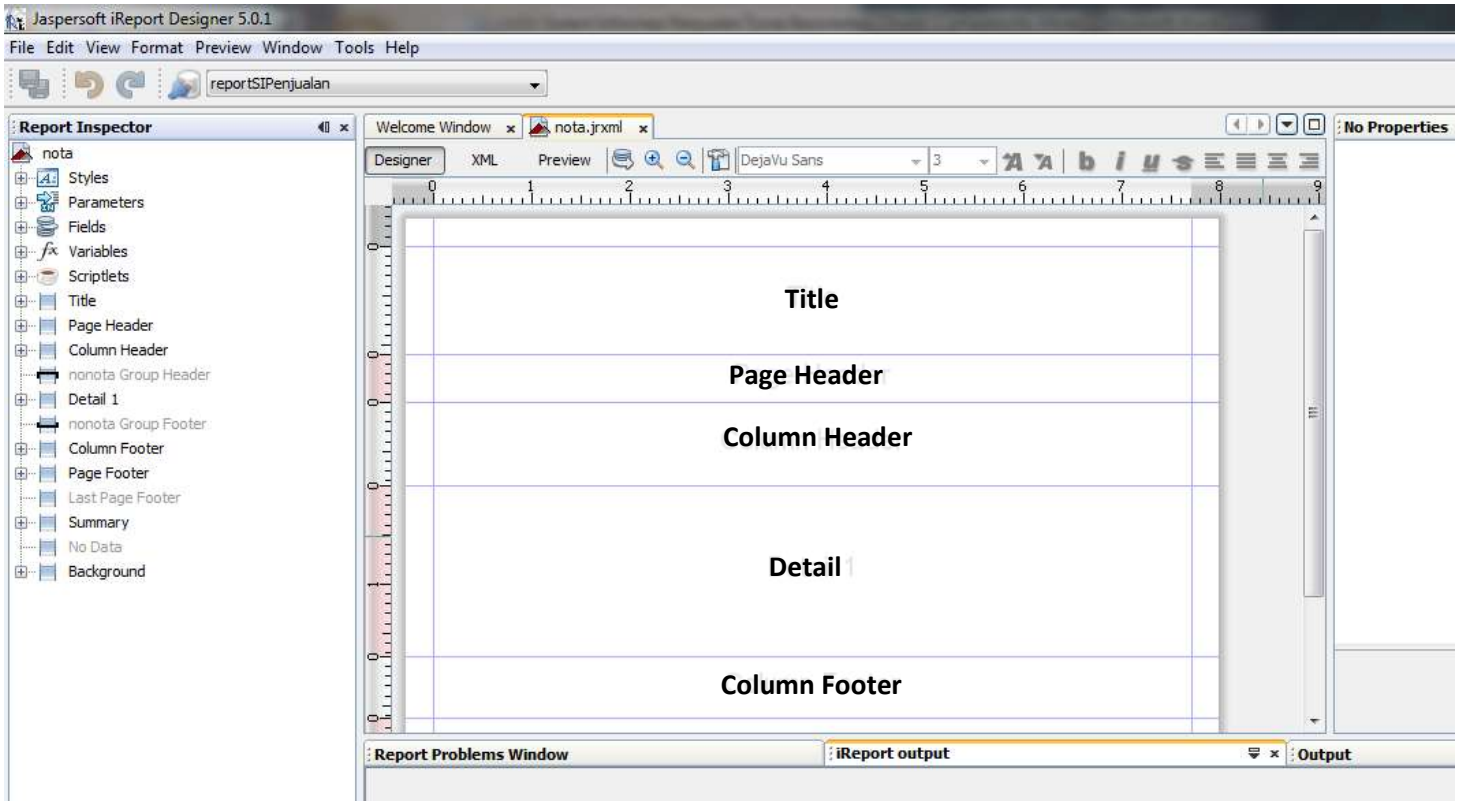
10) Pada pemilihan group, pilih **nonota** pada **group1**. Kemudian klik **next**.



11) Langkah-langkahnya sudah selesai, silahkan klik tombol **Finish**.

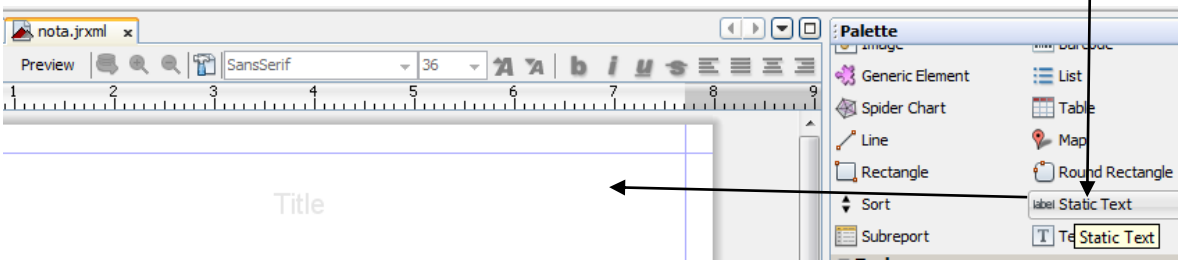


12) Berikut tampilan awal setelah langkah-langkahnya selesai.

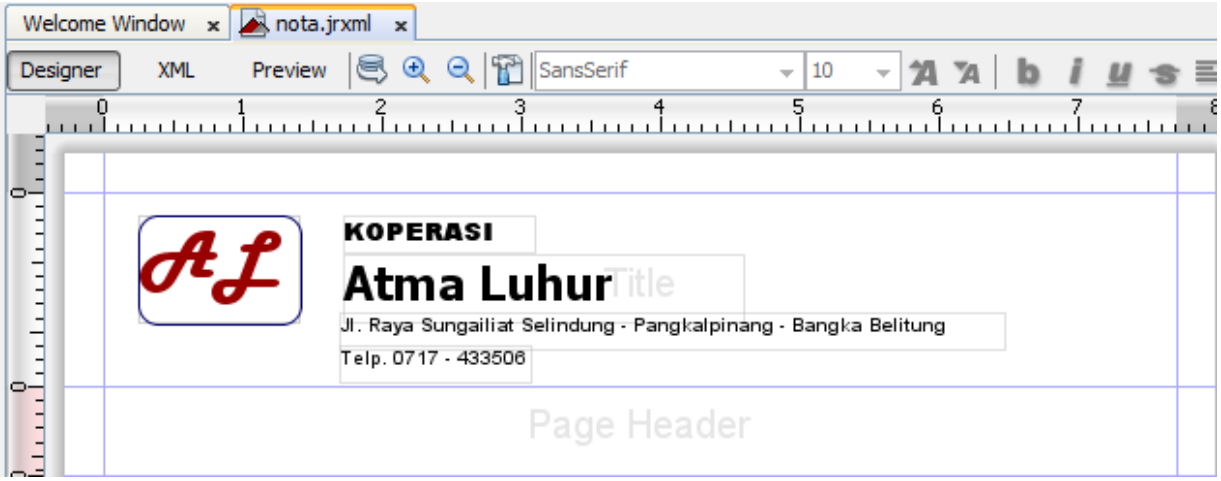


Seperti yang terlihat pada gambar, report punya beberapa bagian yaitu : **Title**, **Page Header**, **Column Header**, **Detail** dan **Column Footer**.

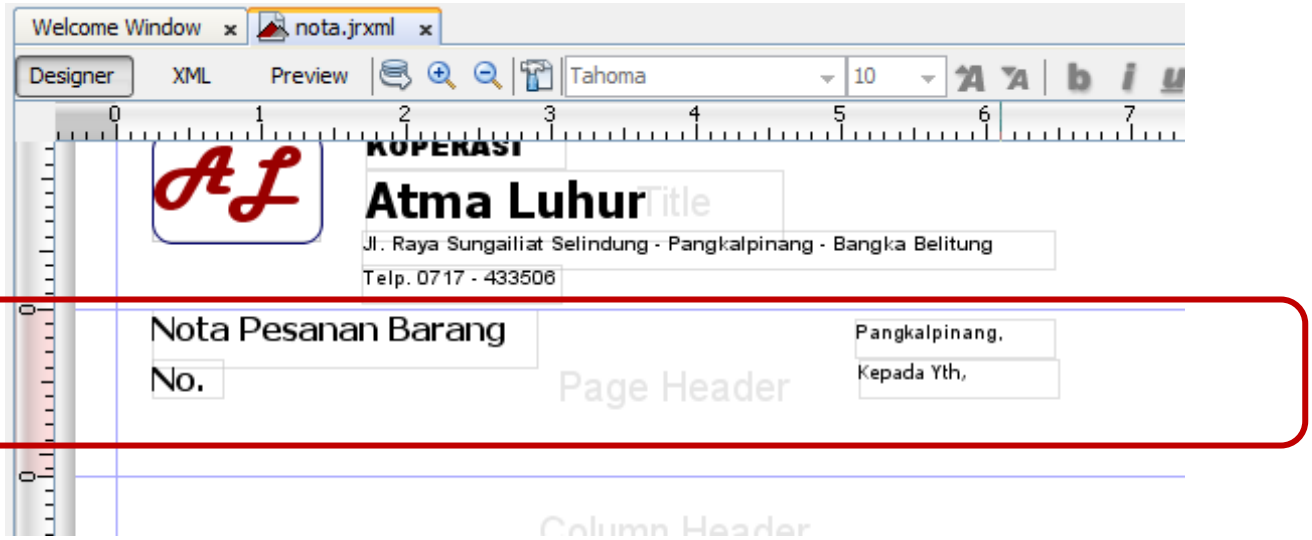
13) Tambahkan Judul pada bagian Title dengan cara memasukkan elemen **Static Text** dari **Palette**. Apabila Palette belum ada, bisa ditampilkan dengan cara mengklik menu **Windows** dan klik sub menu **Palette**.



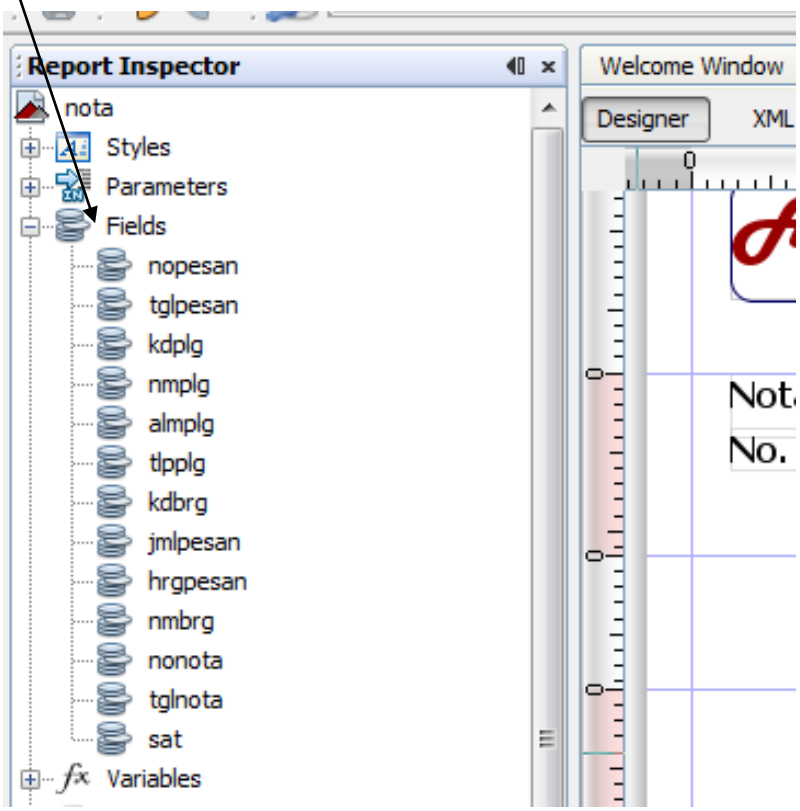
Tampilan akhir sebagai berikut :



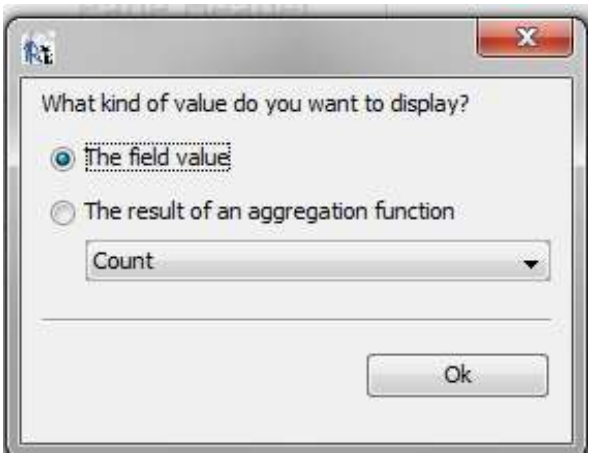
14) Tambahkan tulisan pada bagian **Page Header** dengan contoh sebagai berikut :



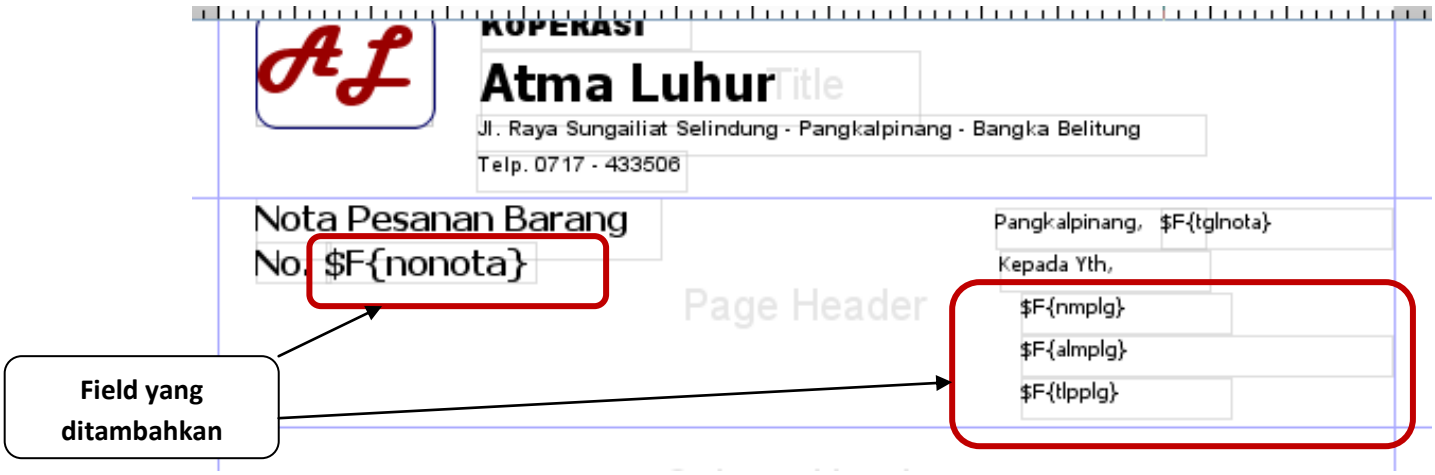
15) Tambahkan **Field** dari **Report Inspector**. Field yang dimasukkan ke yaitu, **nonota**, **tglnota**, **nmplg**, **almplg** dan **tlpplg** dengan cara mendrag field-field tersebut.



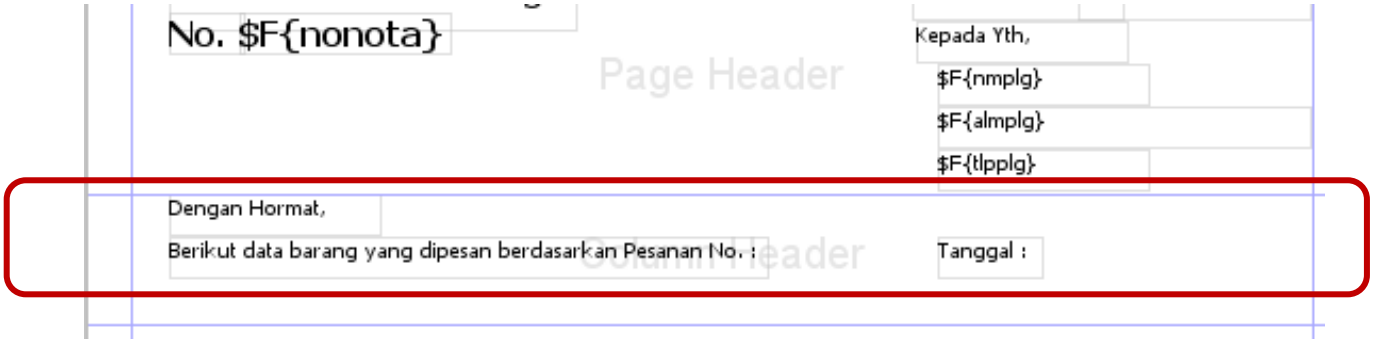
Apabila muncul tampilan sebagai berikut, tekan **tombol Ok** saja :



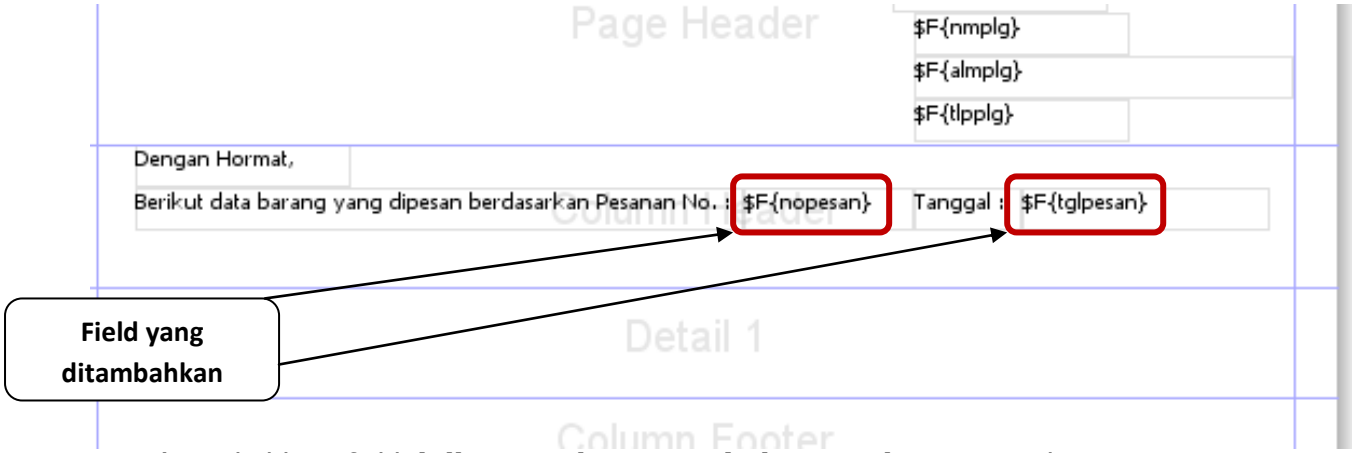
Atur font dan font size dari field yang ditampilkan ke report, sehingga tampilan report sebagai berikut :



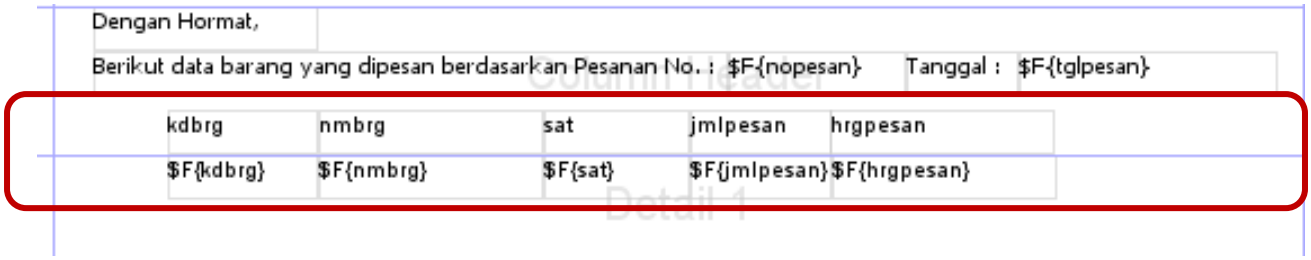
16) Tambahkan tulisan pada **column header** dengan contoh sebagai berikut :



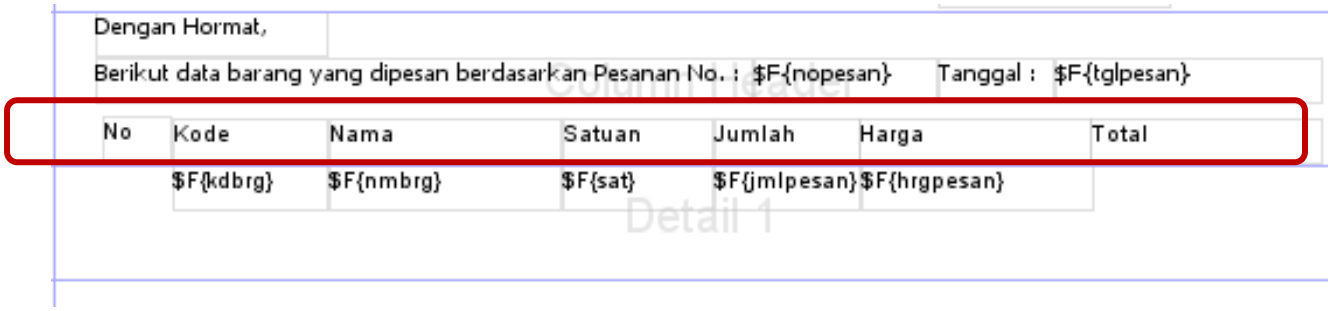
17) Tambahkan field **nopesan** dan **tglpesa**n dari **Report Inspector** sehingga tampilan akhir sebagai berikut :



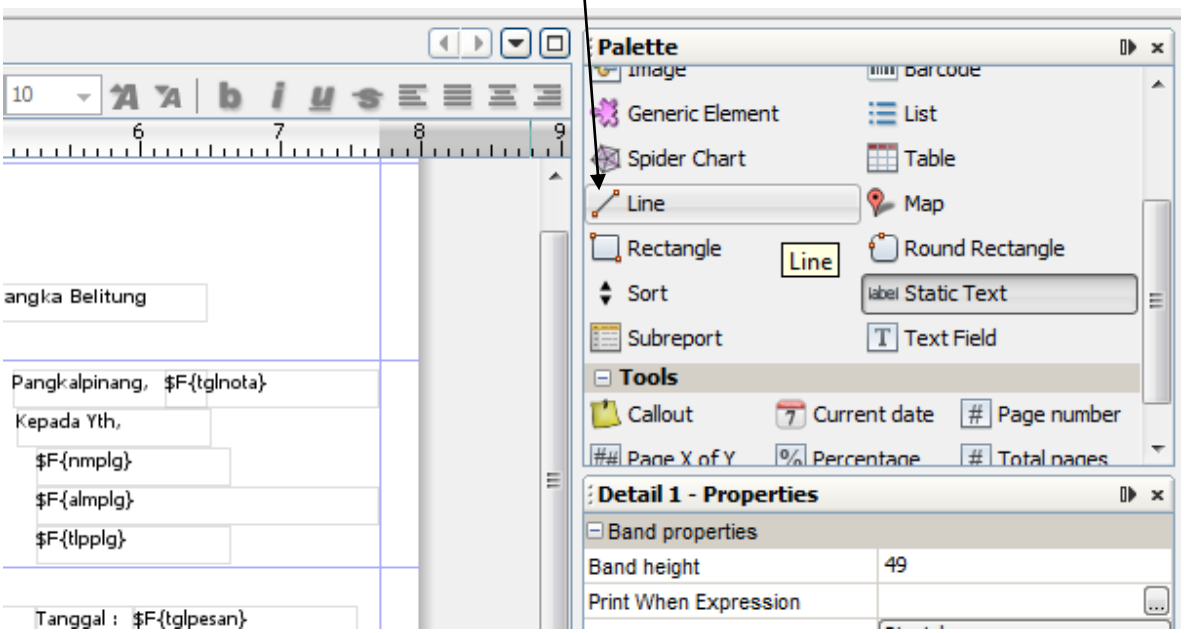
18) Tambahkan field **kdbrg**, **nmbrg**, **sat**, **jmlpesan**, **hrgpesan** dari **Report Inspector** ke bagian **Detail**. Nama kolom dari masing-masing field akan ditampilkan otomatis pada bagian **Column Header**. Atur posisi field-field tersebut sehingga tampilan sebagai berikut :



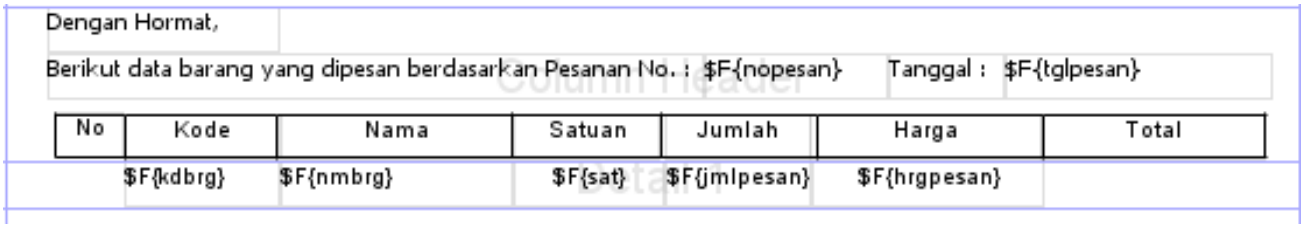
19) Ubah nama kolom dari masing-masing field dan tambahkan No dan Total. Sehingga tampilan sebagai berikut :



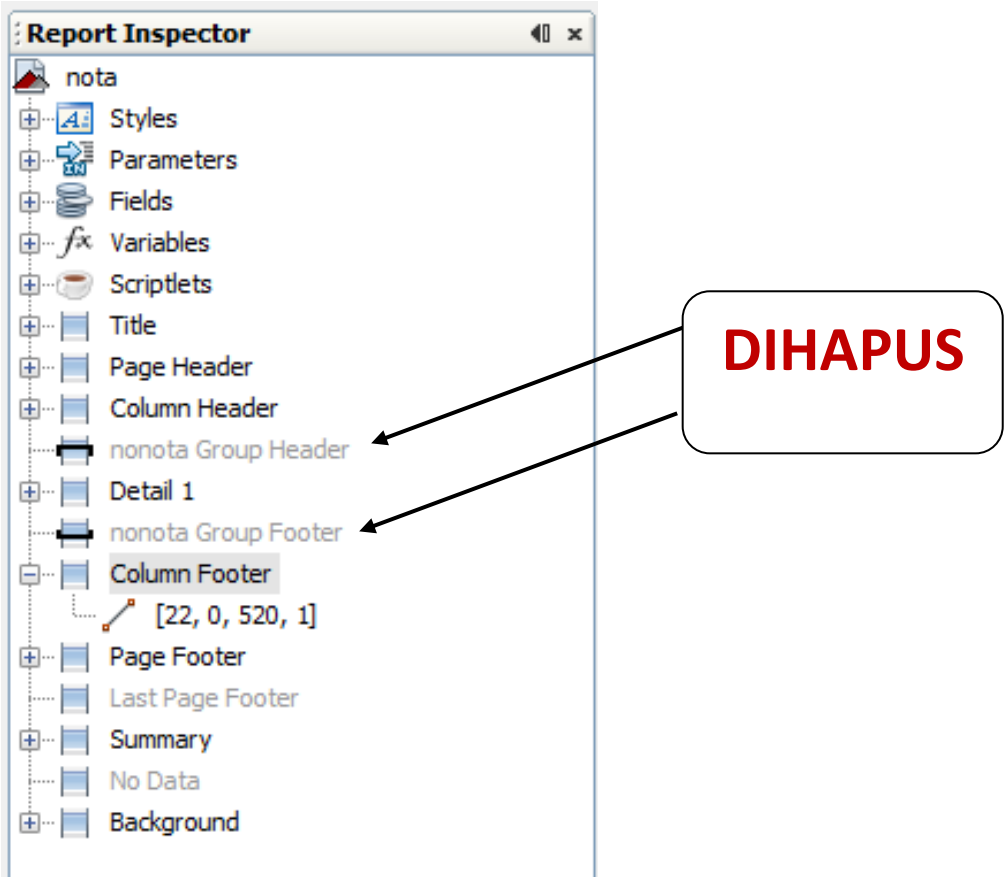
20) Tambahkan garis menggunakan objek **Line** dari **Palette**.



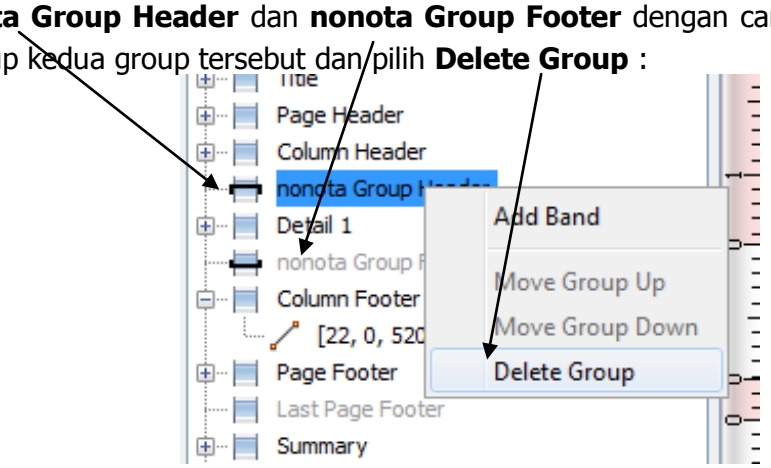
Tampilan akhir sebagai berikut :



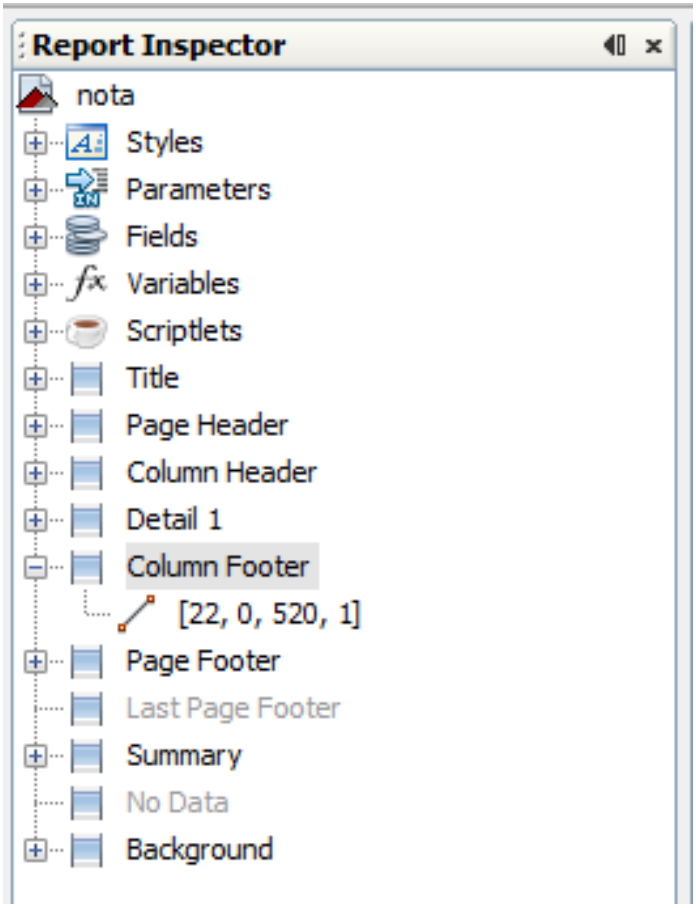
21) Untuk mengatasi space yang kosong antara detail data barang dan total serta data validasi dari bagian penjualan, maka perlu dibuat group sendiri untuk mengatasi hal tersebut. Perhatikan group yang ditandai pada Report Inspector berikut :



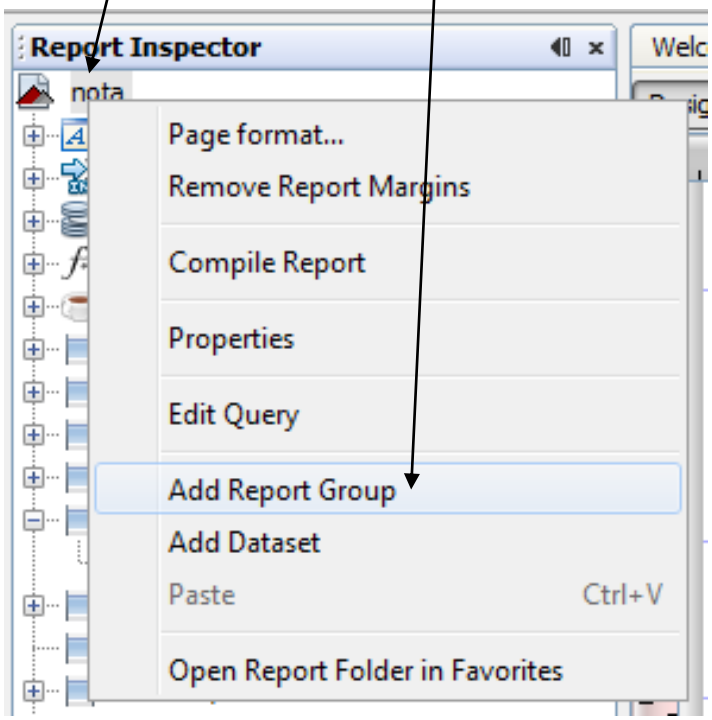
22) Hapus **nonota Group Header** dan **nonota Group Footer** dengan cara klik kanan di group kedua group tersebut dan pilih **Delete Group** :



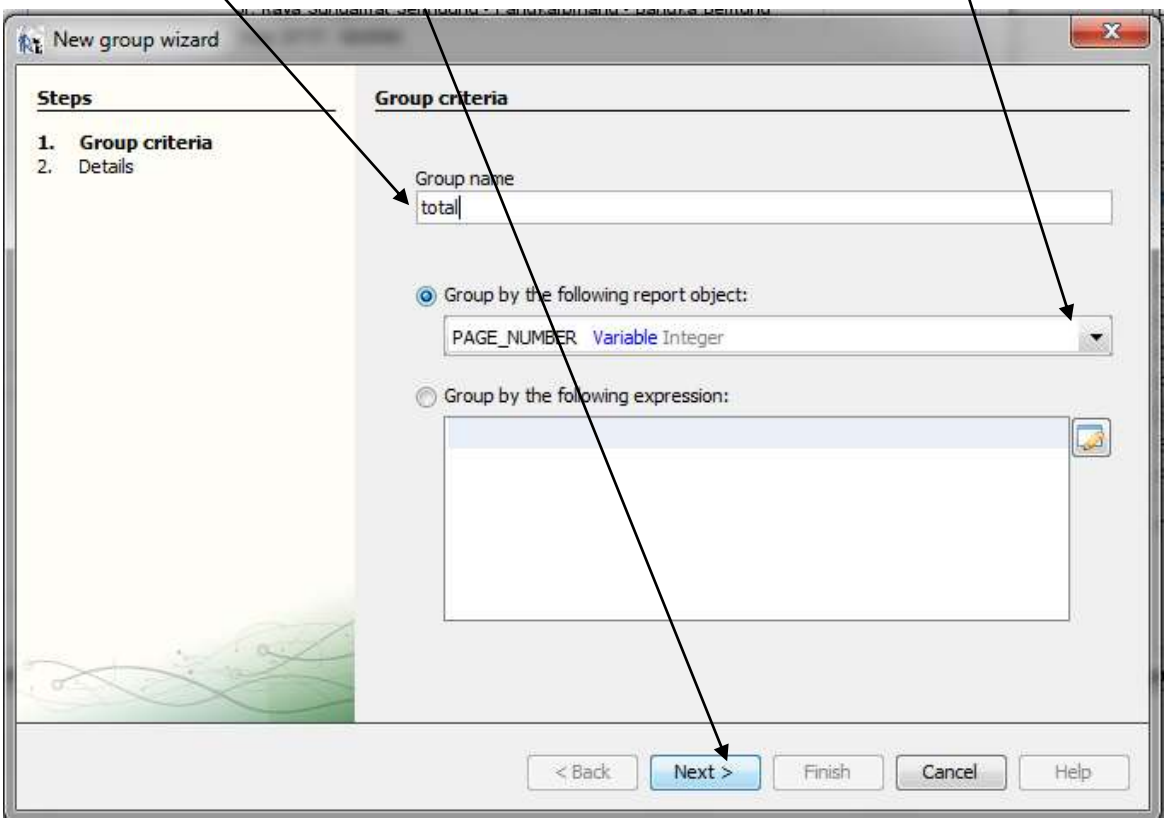
Sehingga tampilan akhir sebagai berikut :



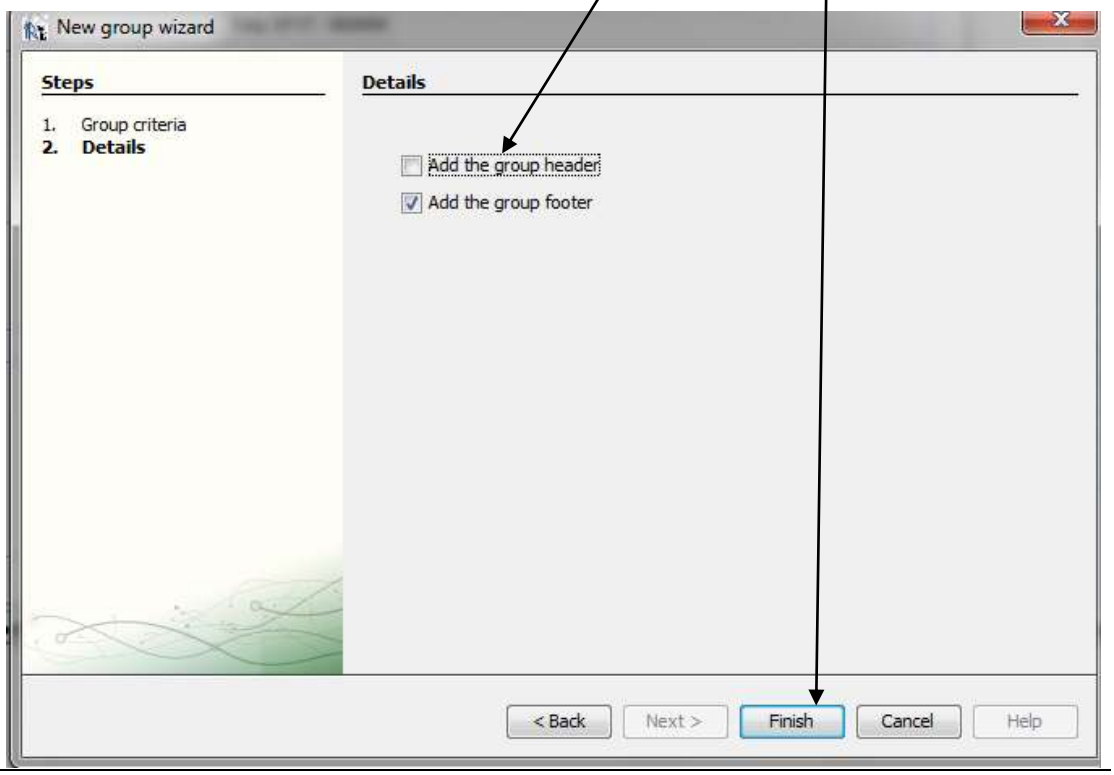
23) Untuk menambah group baru, klik kanan pada project di **Report Inspector** (dalam hal ini **nota**) dan pilih **Add Report Group**.



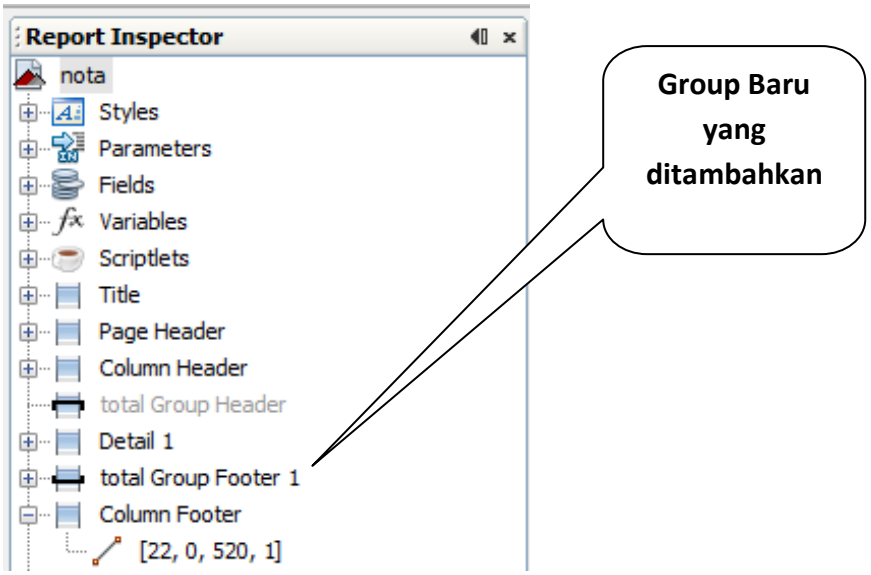
24) Pada kotak dialog **New Group Wizard**, pada bagian **Group Name** isi dengan **total** dan Group by the following report object pilih **PAGE NUMBER**. Kemudian klik tombol **Next**.



25) Pada langkah berikutnya pastikan **Add Group Header TIDAK** dalam keadaan dipilih. Seperti gambar berikut kemudian klik tombol **Finish**.

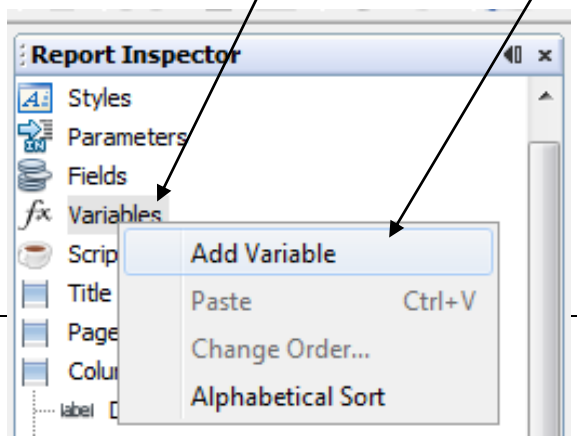


Apabila group sudah berhasil ditambahkan hasil pada Report Inspector sebagai berikut :

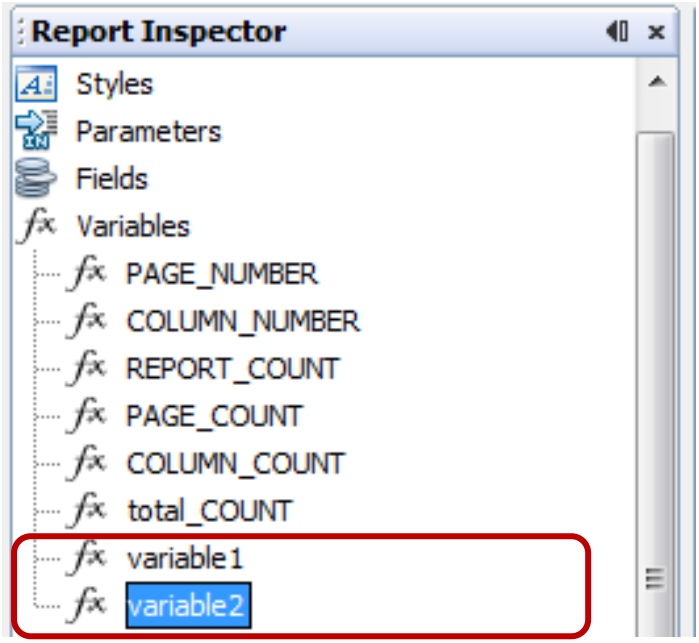


26) Pada **total group Footer** tambahkan tulisan sebagai berikut :

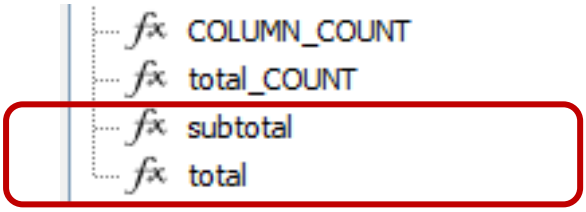
27) Langkah berikutnya, menambahkan dua variabel baru, masing-masing berfungsi untuk menampung hasil perkalian antara jumlah pesan dan harga, dan berfungsi untuk mentotal akhir pembayaran/pembelian. Pada **Report Inspector**, klik kanan pada **Variables** dan pilih **Add Variable**.



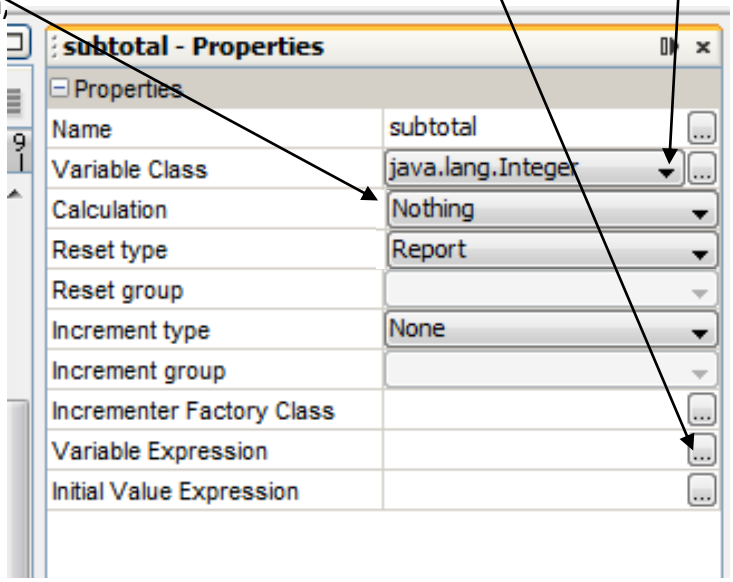
Lakukan sebanyak dua kali sehingga variabel yang terbentuk sebagai berikut :



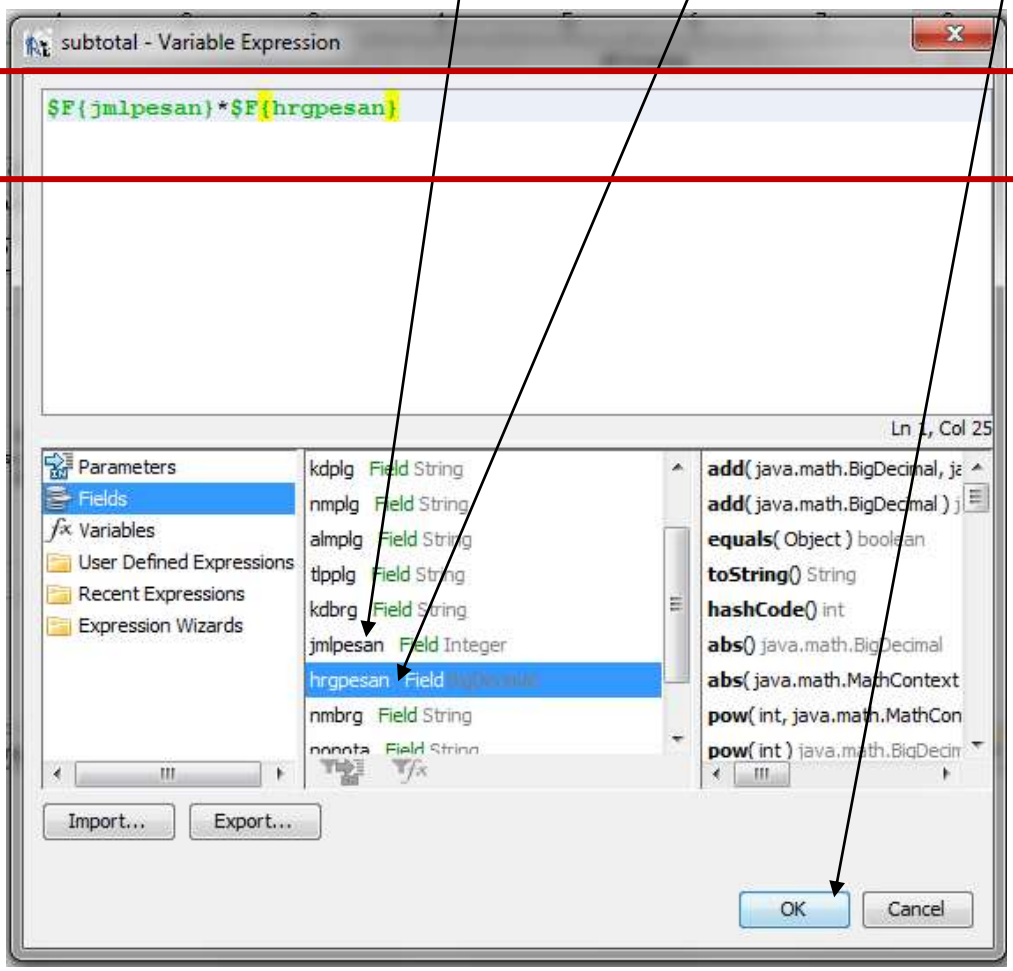
28) Ganti nama variabel tersebut dengan **subtotal** dan **total**.



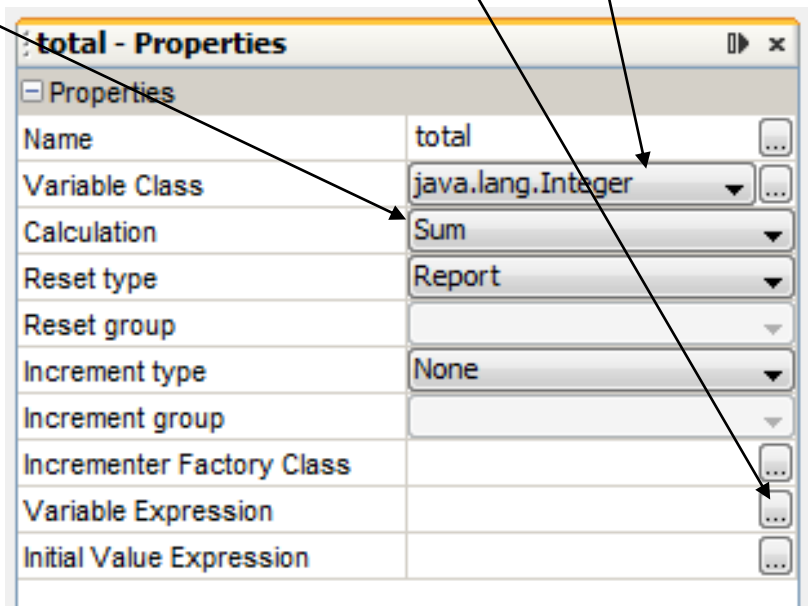
29) Pastikan variabel subtotal dalam keadaan terpilih, lakukan perubahan pada **subtotal-properties**. **Variable class** pilih **java.lang.integer**. **Calculating** pilih **Nothing**, kemudian klik **Variable Expression** pada tombol yang sudah disediakan,



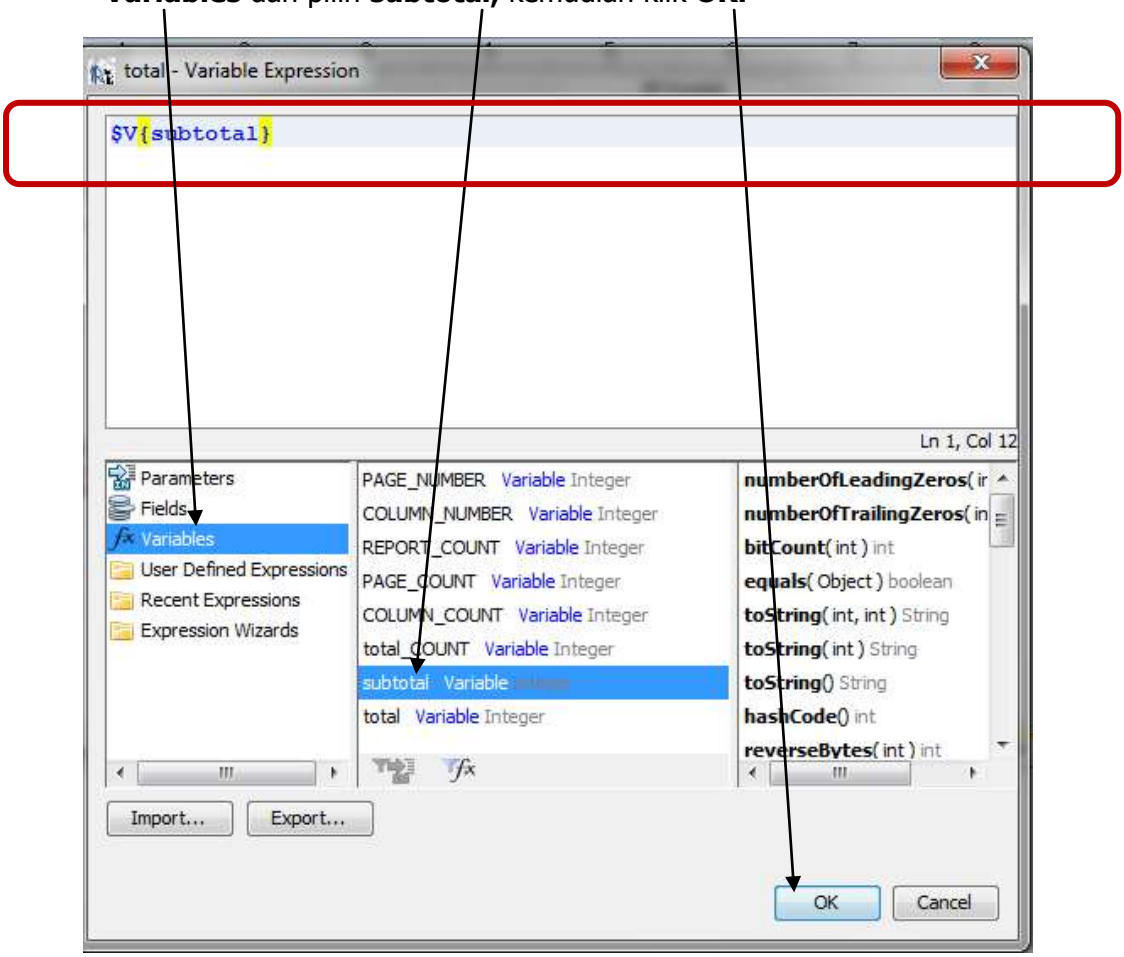
30) Pada kotak dialog subtotal-Variable Expression, tambahkan perintah dengan cara mendouble klik **jmlpesan** dikali (*) **hrgpesan,** kemudian klik **OK**



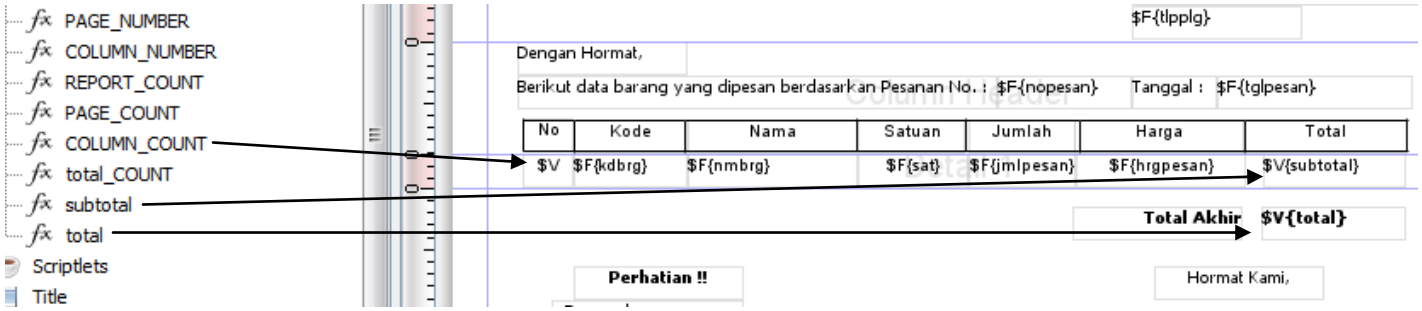
31) Pastikan variabel subtotal dalam keadaan terpilih, lakukan perubahan pada **total-properties**. **Variable class** pilih **java.lang.integer**. **Calculating** pilih **Sum**, kemudian klik **Variable Expression** pada tombol yang sudah disediakan,



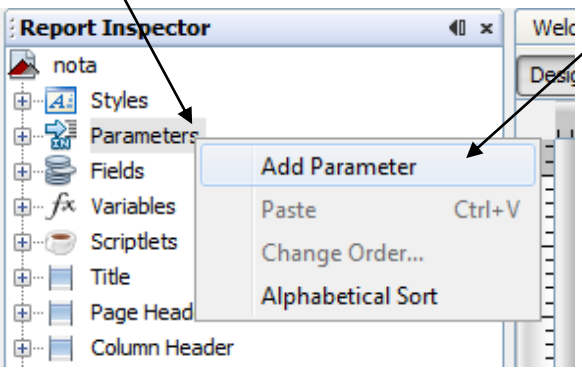
32) Pada kotak dialog **total-Variable Expression**, double klik pada **Variables** dan pilih **subtotal,** kemudian klik **OK.**



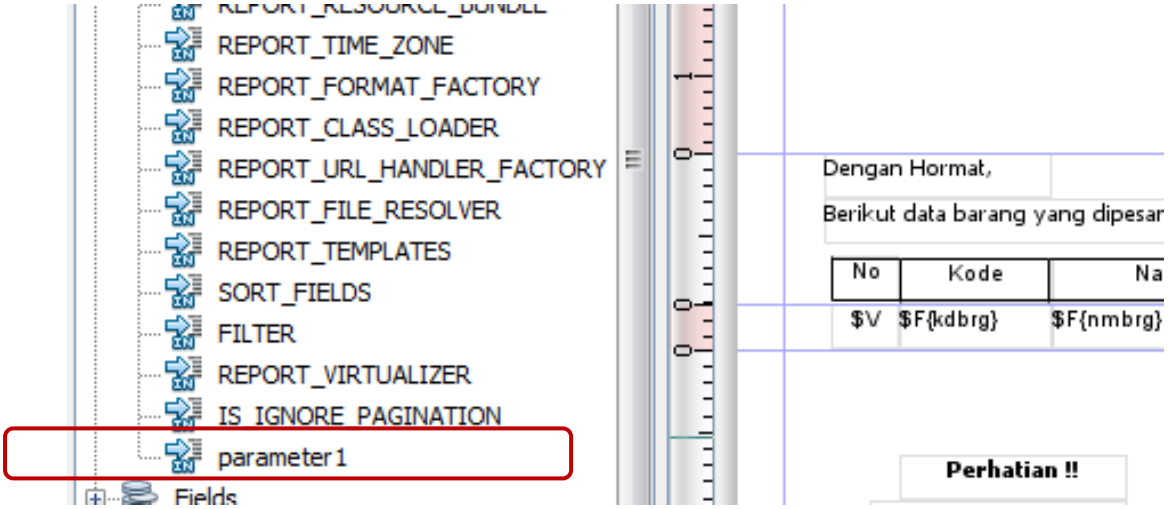
33) Masukkan tiga variabel yang diperlukan kedalam report pada tempat yang sudah disediakan, variabel tersebut **Column_Count**, **subtotal** dan **total** atur posisi variabel tersebut seperti gambar berikut :



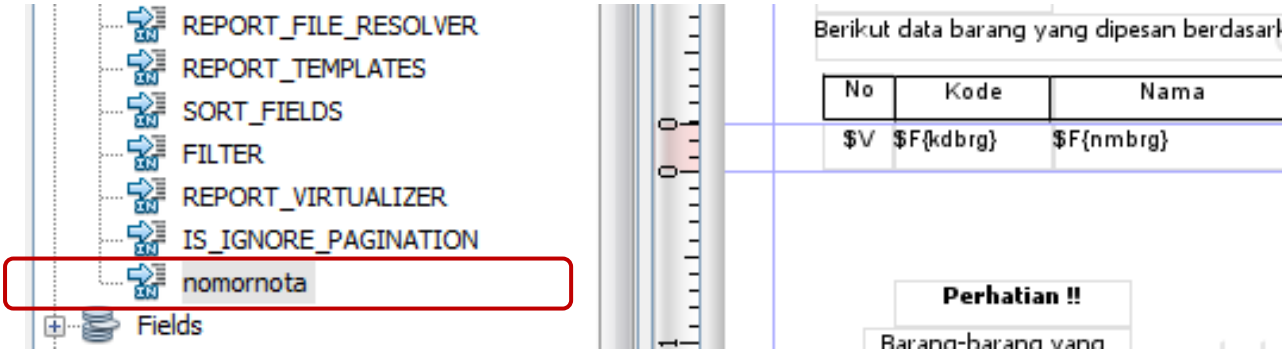
34) Untuk menampilkan data pesanan berdasarkan nomor nota tertentu, maka perlu menambahkan sebuah parameter. Parameter ditambahkan dengan cara mengklik kanan **parameters** pada **Report Inspector**, kemudian pilih **Add Parameter**.



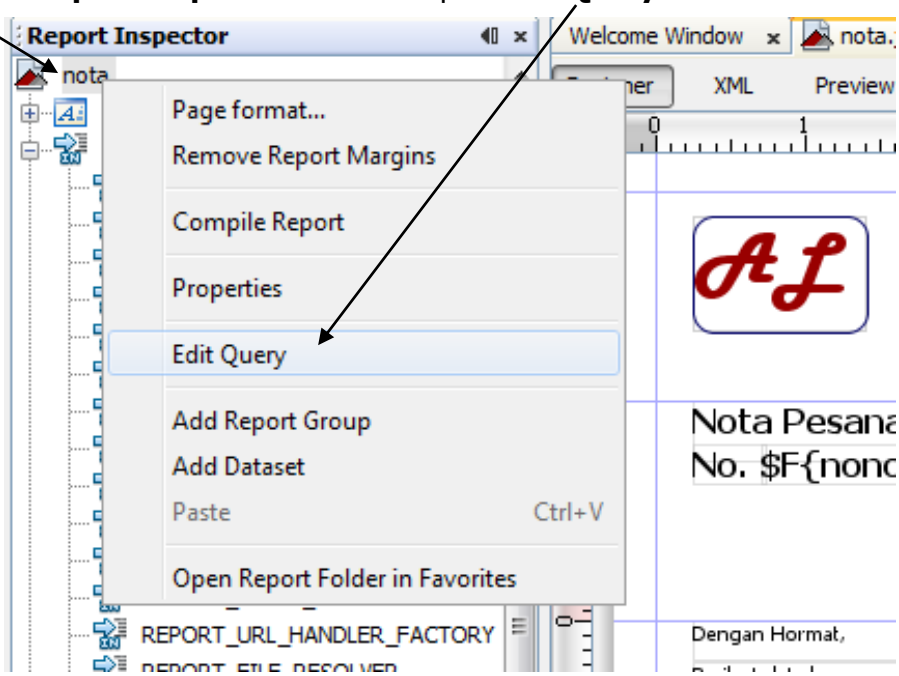
Tampilan pada parameters yang sudah ditambahkan sebagai berikut :



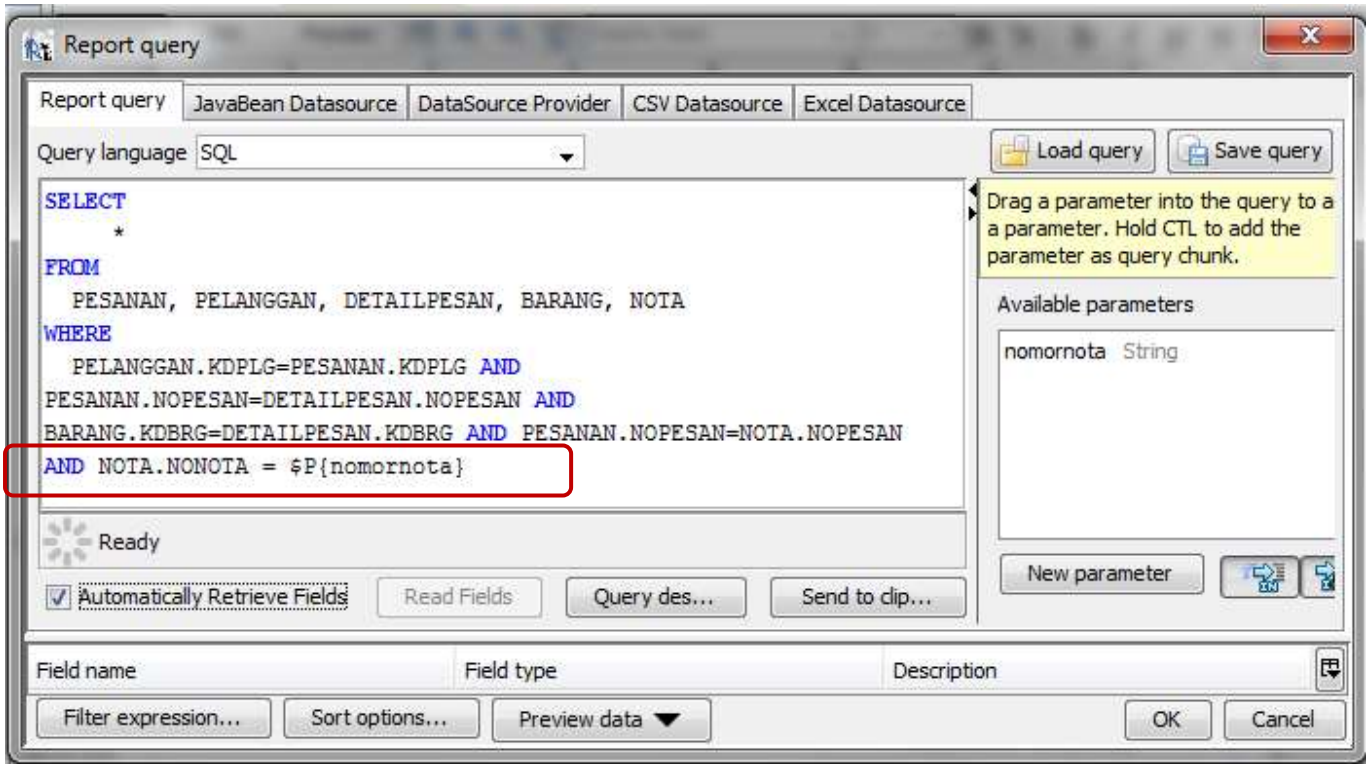
35) Ubah nama **parameter 1** menjadi **nomornota**.



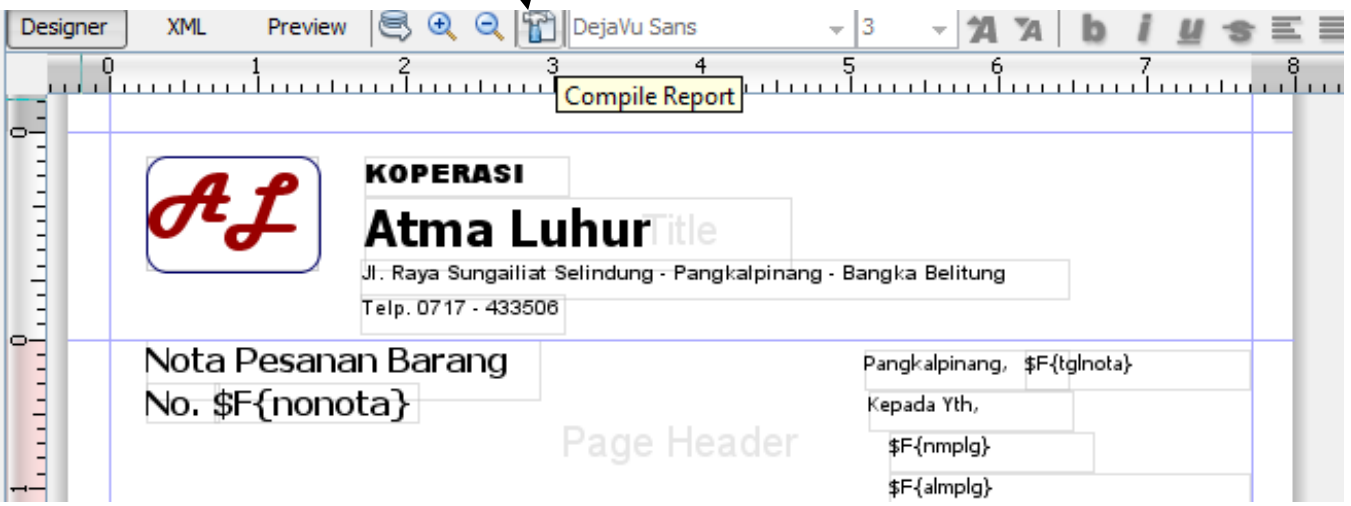
36) Edit query yang sudah dibuat sebelumnya dengan cara mengklik kanan pada **project** di **Report Inspector**. Kemudian pilih **Edit Query**.



37) Tambahkan query seperti yang ditampilkan dalam **kotak segi empat**. Kemudian klik **OK**.



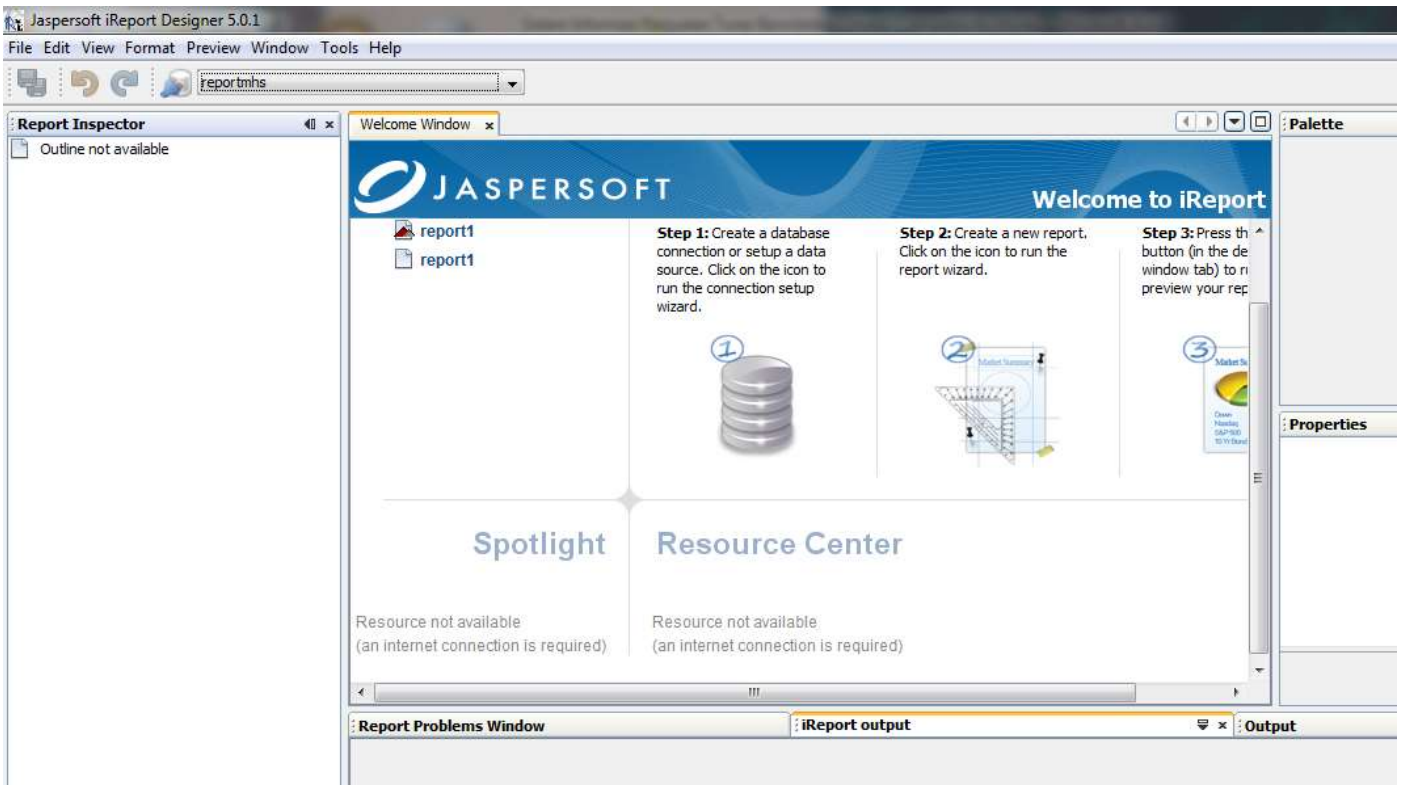
38) Secara keseluruhan nota sudah selesai dibuat. Report Nota silahkan di COMPILE (**Compile Report**) untuk memastikan tidak ada kesalahan dan Report silahkan disimpan/ disave (



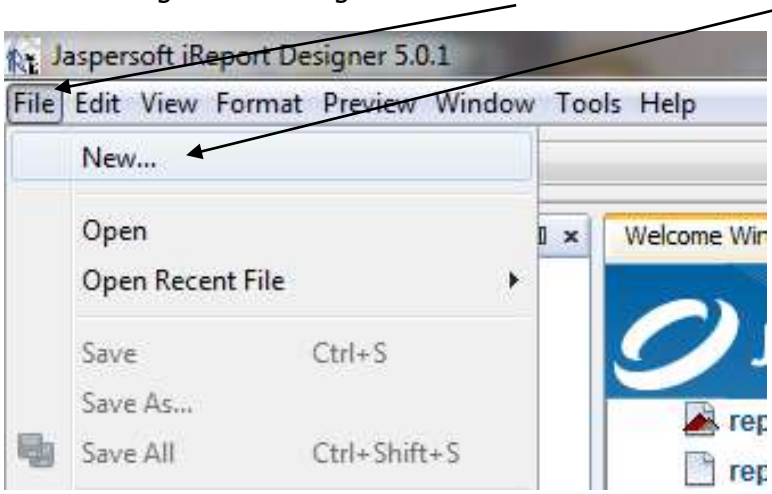
b. Buat LAPORAN PENJUALAN

1) Buka aplikasi iReport (

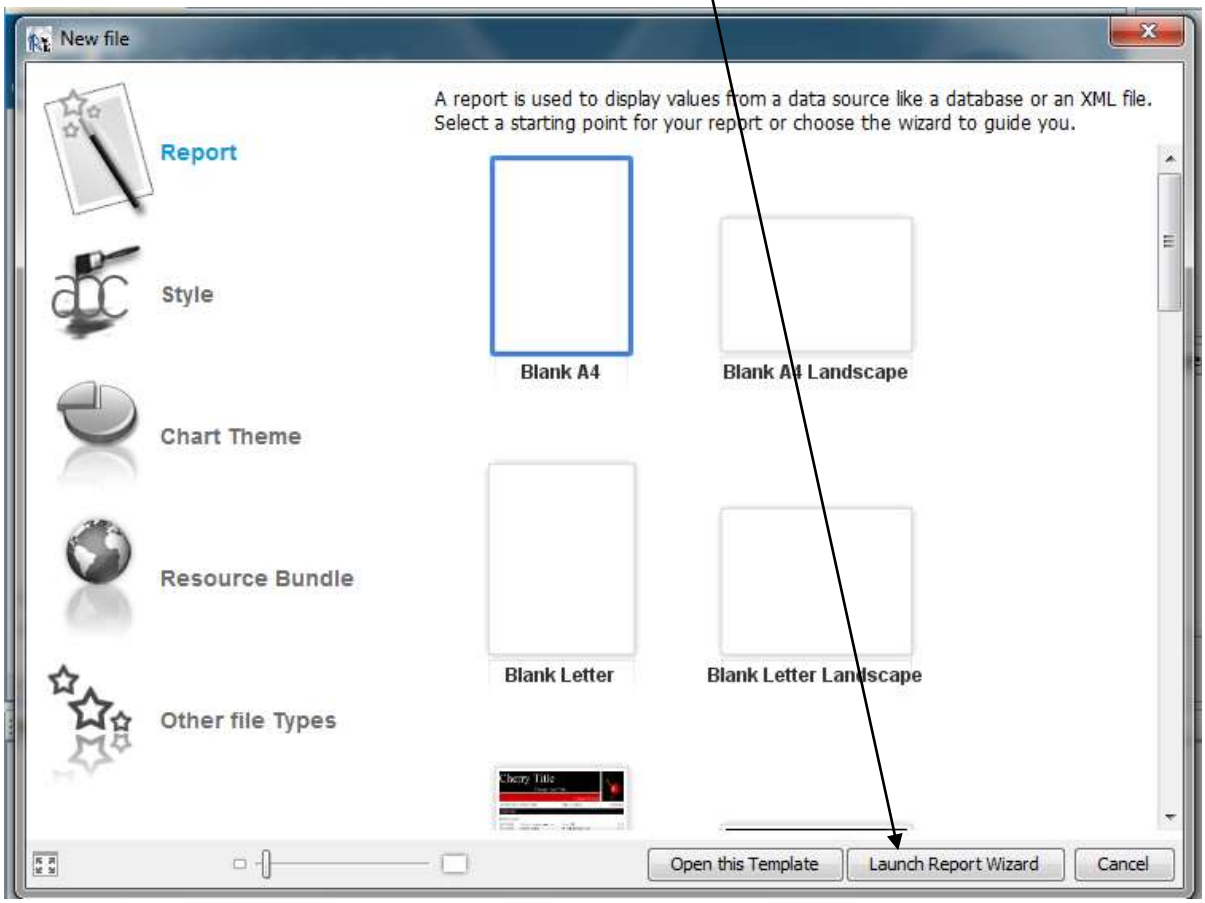
dan tampilan awalnya sebagai berikut :



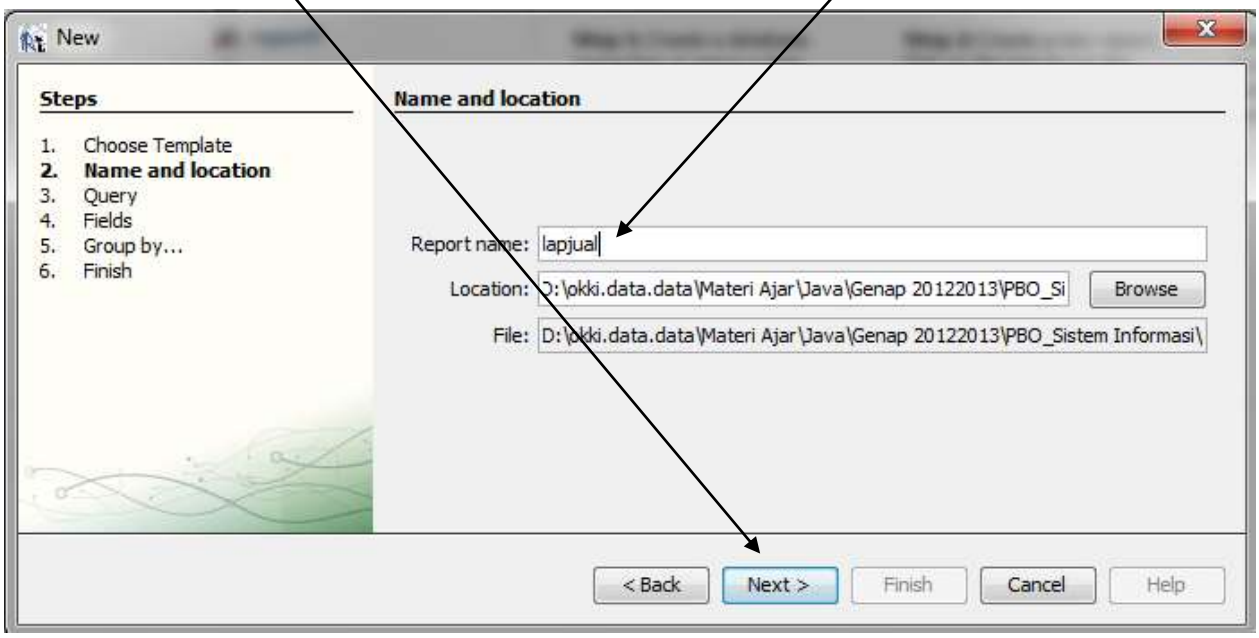
2) Buka report baru dengan cara mengklik menu **File** dan sub menu **New**.



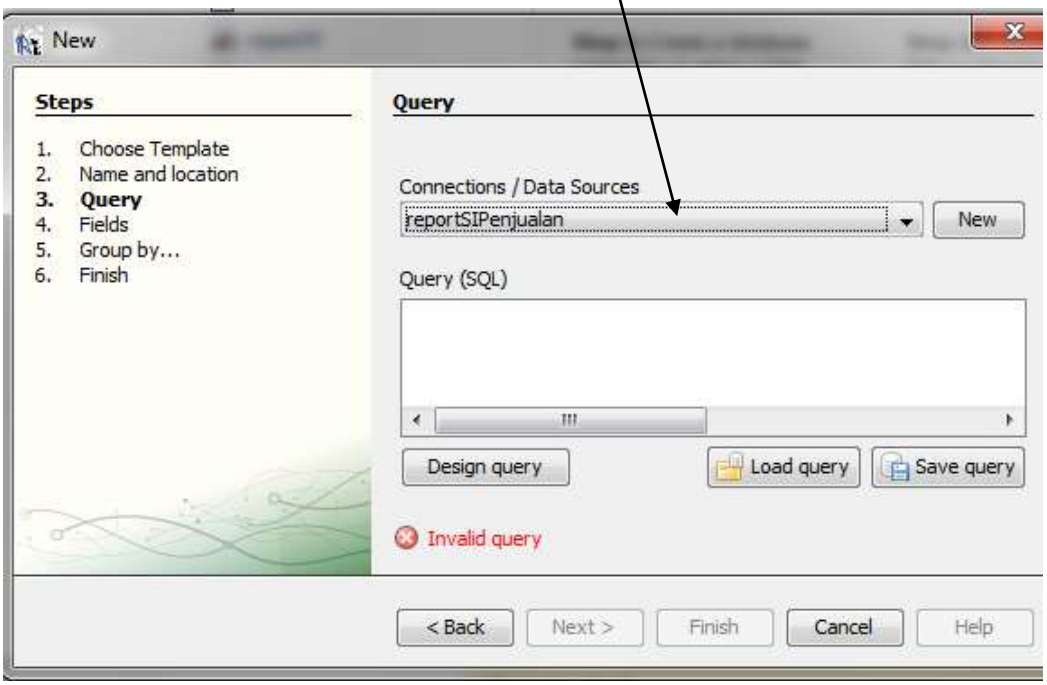
3) Pada kotak dialog **New file**, klik **Launch Report Wizard**.



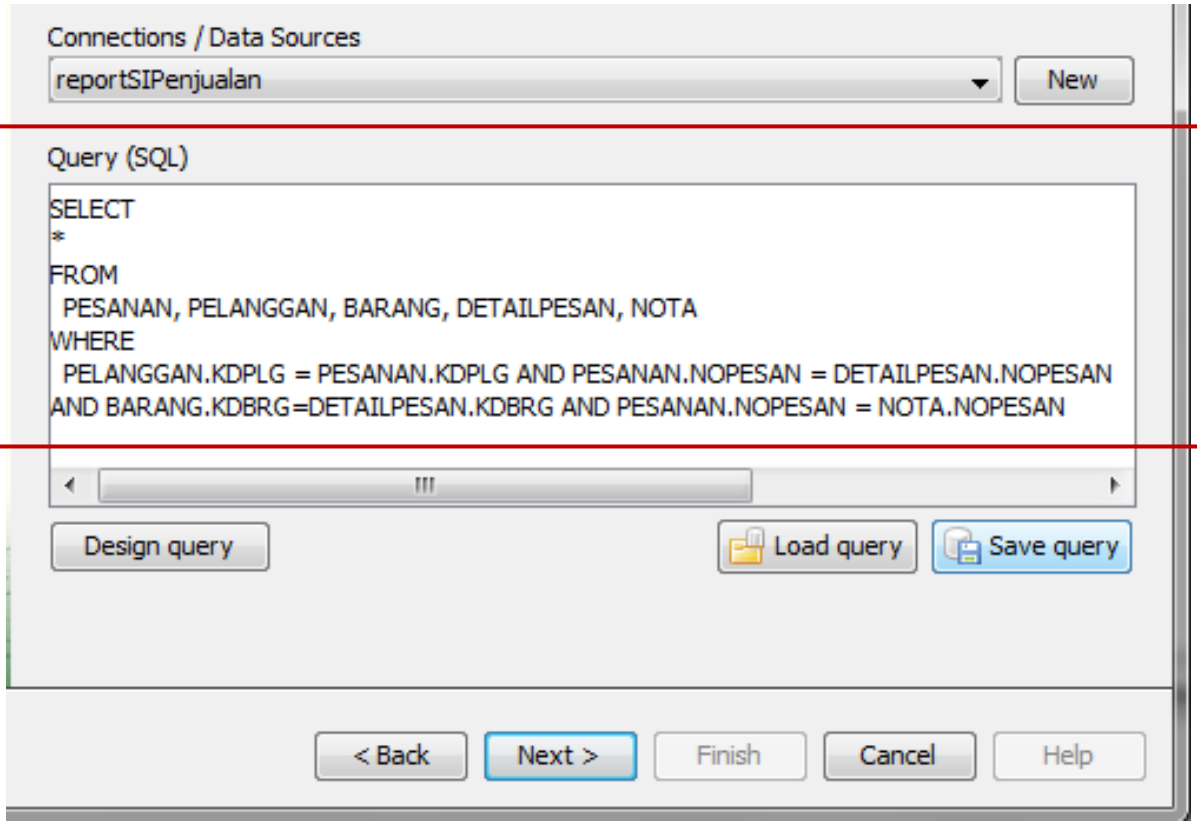
4) Tentukan nama dan lokasi penyimpanan reportnya sesuai dengan keinginan. Pada contoh aplikasi ini, file diberi dengan nama : **lapjual**. Kemudian klik tombol **next**.



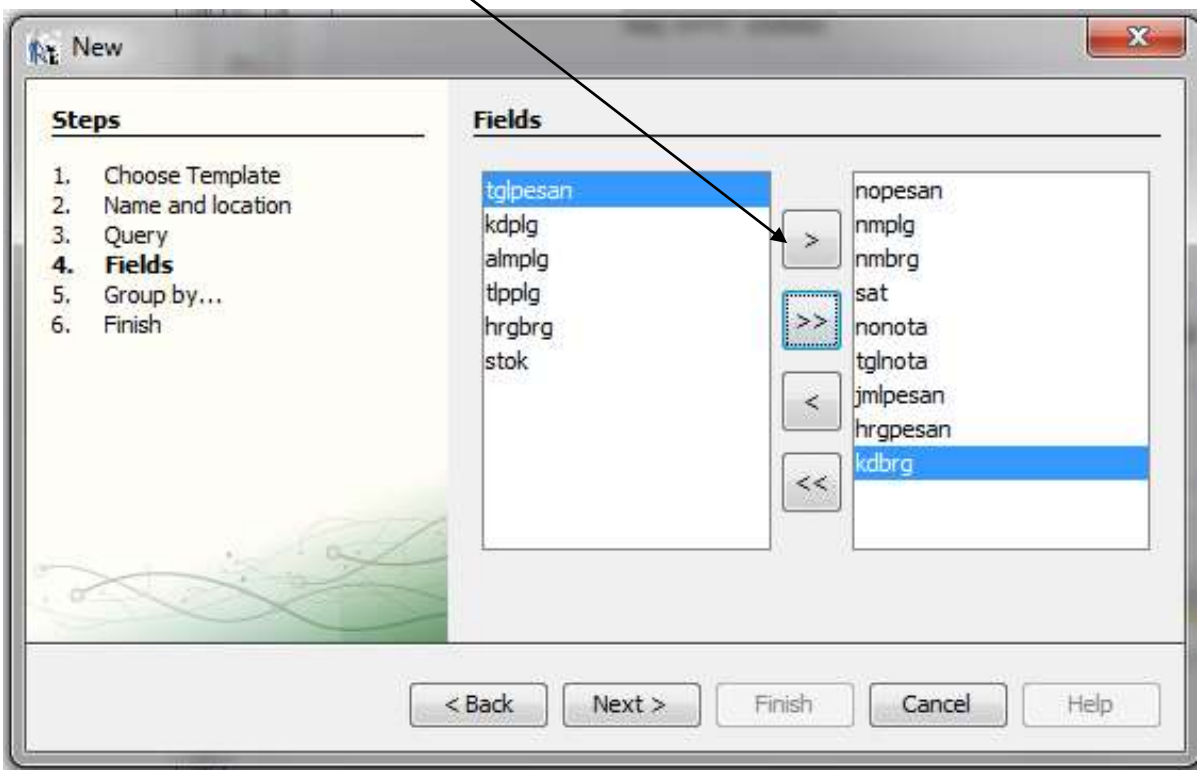
5) Pada kotak dialog New, pilih **reportSIPenjualan** pada Connections / Data Source. Hal ini dilakukan, karena koneksinya sudah dibuat sebelumnya pada saat membuat nota. Database yang digunakan sama dengan nota. Apabila koneksinya belum ada, maka ikuti langkah-langkahnya pada bagian membuat koneksi di nota.



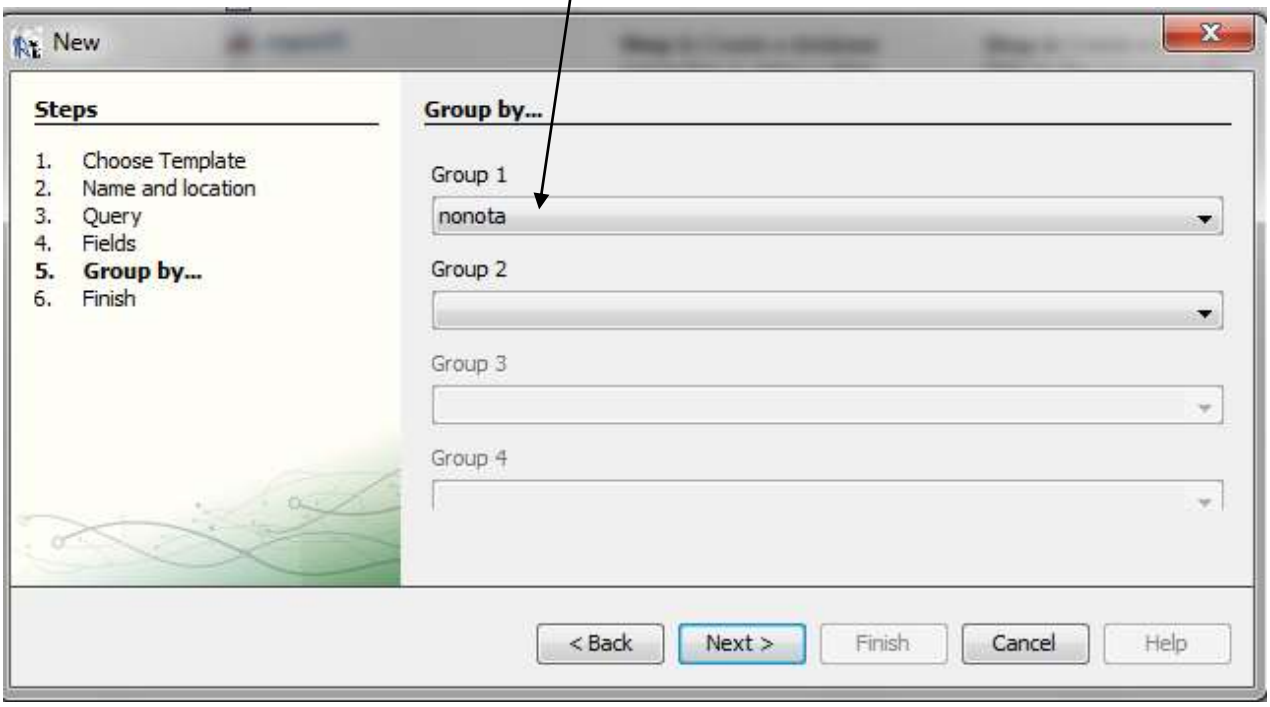
6) Ketik perintah berikut pada **Query (SQL)**. Untuk melanjutkan ke langkah-langkah berikutnya klik **Next**.



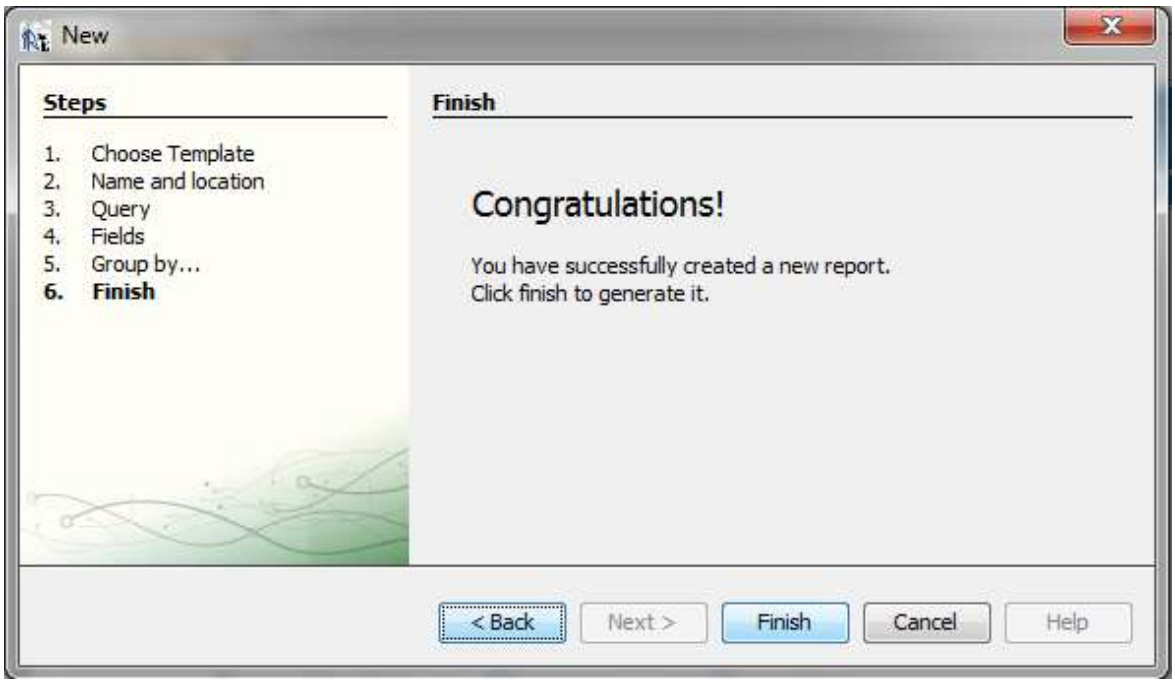
7) Pilih field-field yang diperlukan untuk ditampilkan direport dengan cara mengklik **field** dan **tombol** yang sudah disediakan. Field-field yang diperlukan seperti yang ditampilkan. Kemudian klik **Next.**



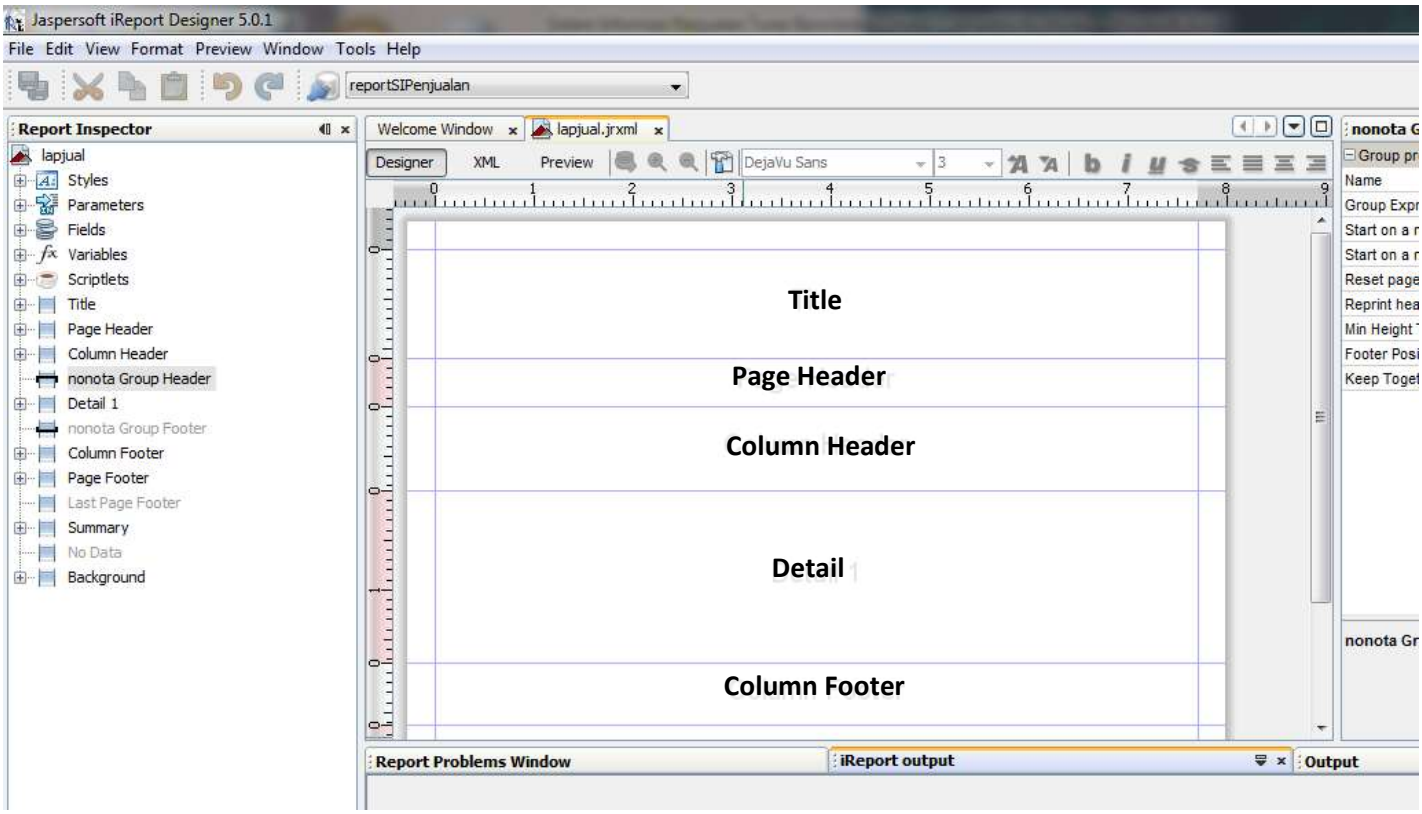
8) Pada pemilihan group, pilih **nonota** pada **group1**. Kemudian klik **next.**



9) Langkah-langkahnya sudah selesai, silahkan klik tombol **Finish**.

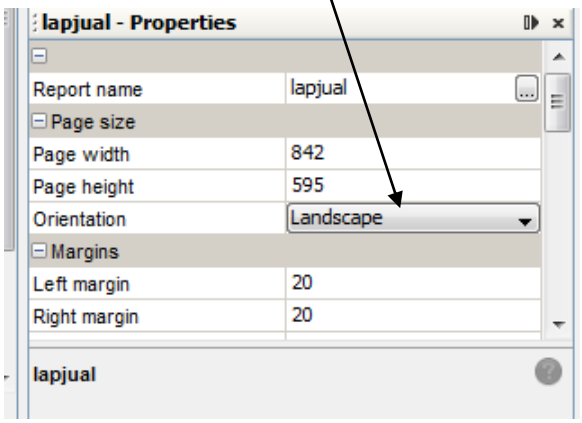


10) Berikut tampilan awal setelah langkah-langkahnya selesai.

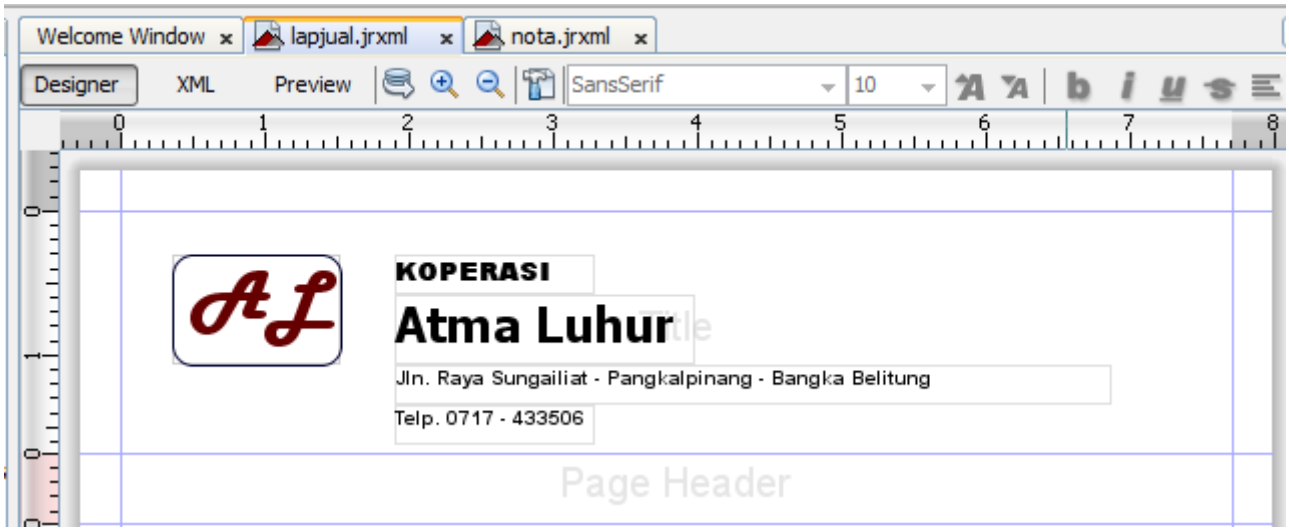


Seperti yang terlihat pada gambar, report punya beberapa bagian yaitu : **Title**, **Page Header**, **Column Header**, **Detail** dan **Column Footer**

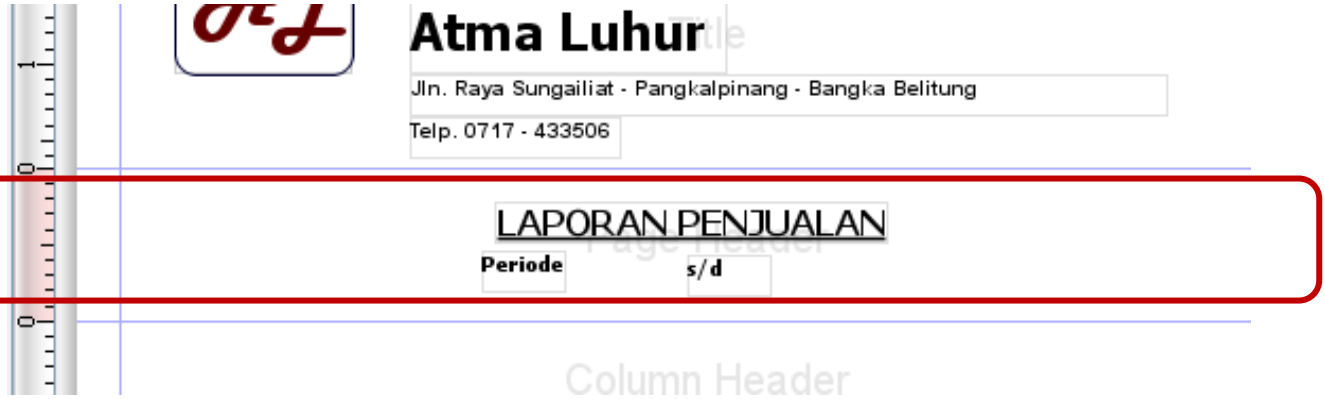
11) Ubah orientasi kertas menjadi **landscape**. Pada saat mengubah orientasi pastikan project **lapjual** pada Report Inspector dalam keadaan terpilih.



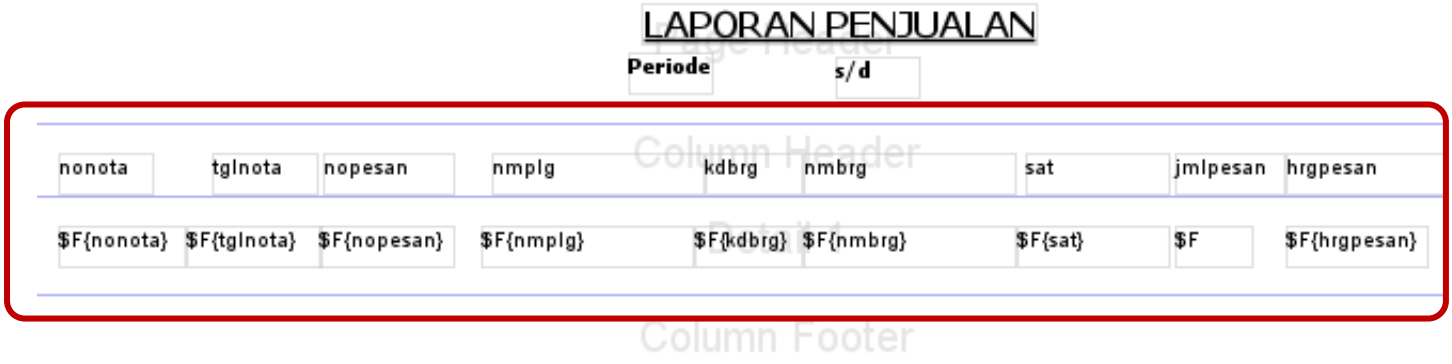
12) Tambahkan Judul pada bagian Title dengan cara memasukkan elemen **Static Text** dari **Palette**. Apabila Palette belum ada, bisa ditampilkan dengan cara mengklik menu **Windows** dan klik sub menu **Palette.**



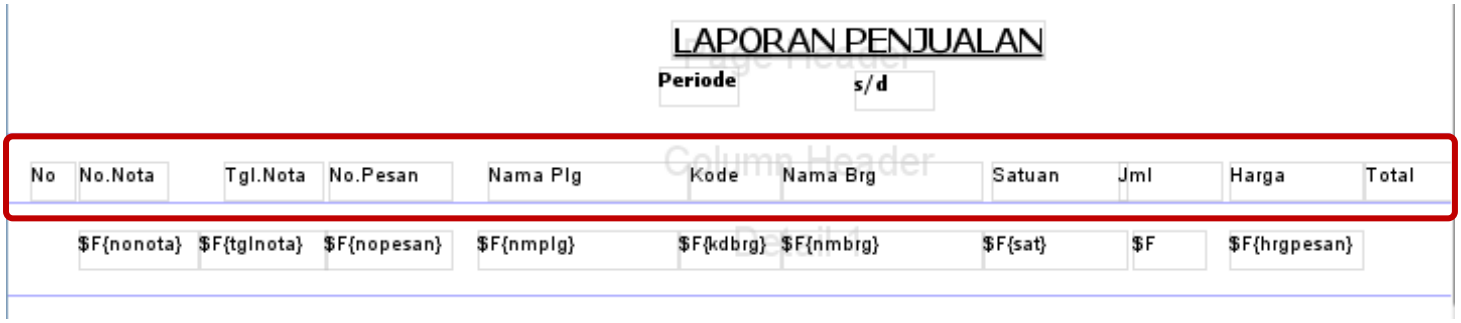
13) Tambahkan tulisan pada bagian **Page Header** dengan contoh sebagai berikut :



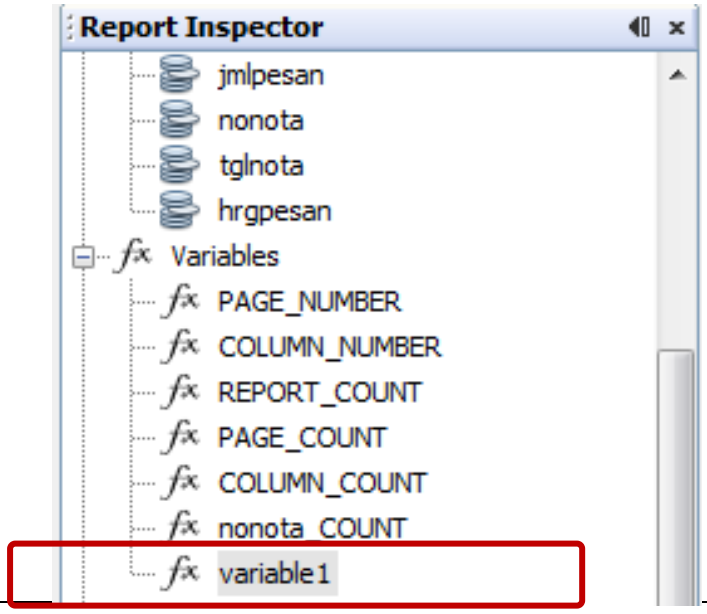
14) Tambahkan seluruh **field** dari Report Inspector ke bagian **Detail.** Posisi field seperti yang dicontohkan. Nama kolom dari masing-masing field akan ditampilkan otomatis ke bagian **Column Header**, dengan contoh sebagai berikut :



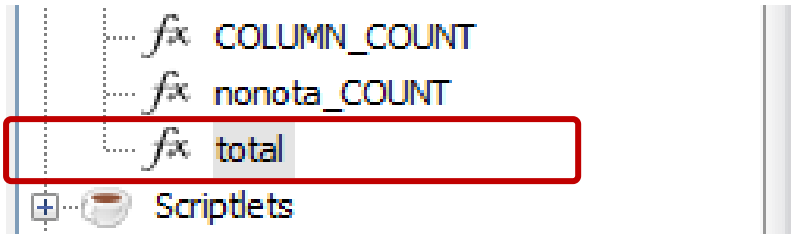
15) Ubah nama kolom dari masing-masing field dan tambahkan No dan Total. Sehingga tampilan sebagai berikut :



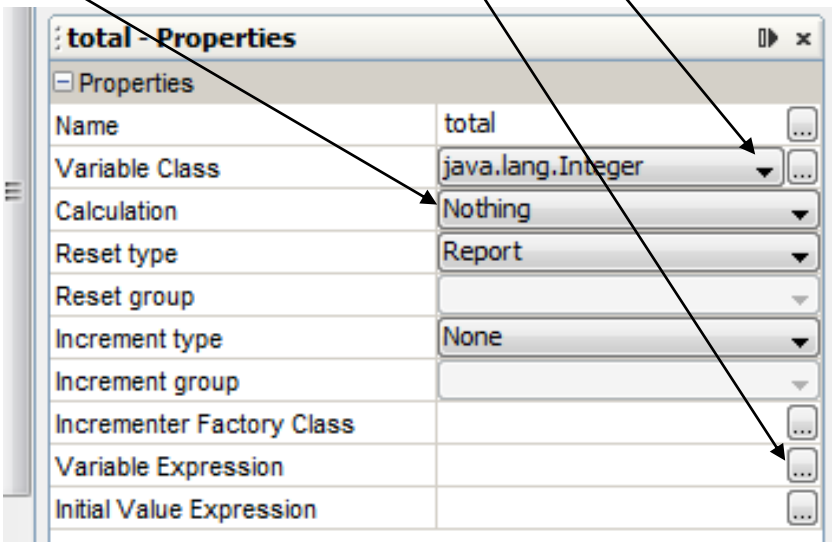
16) Langkah berikutnya, tambahkan sebuah variabel baru untuk menampung hasil perkalian antara jumlah pesan dan harga. Pada Report Inspector, klik kanan pada **Variables** dan pilih **Add Variable**. Sehingga Tampilan Variables sebagai berikut :



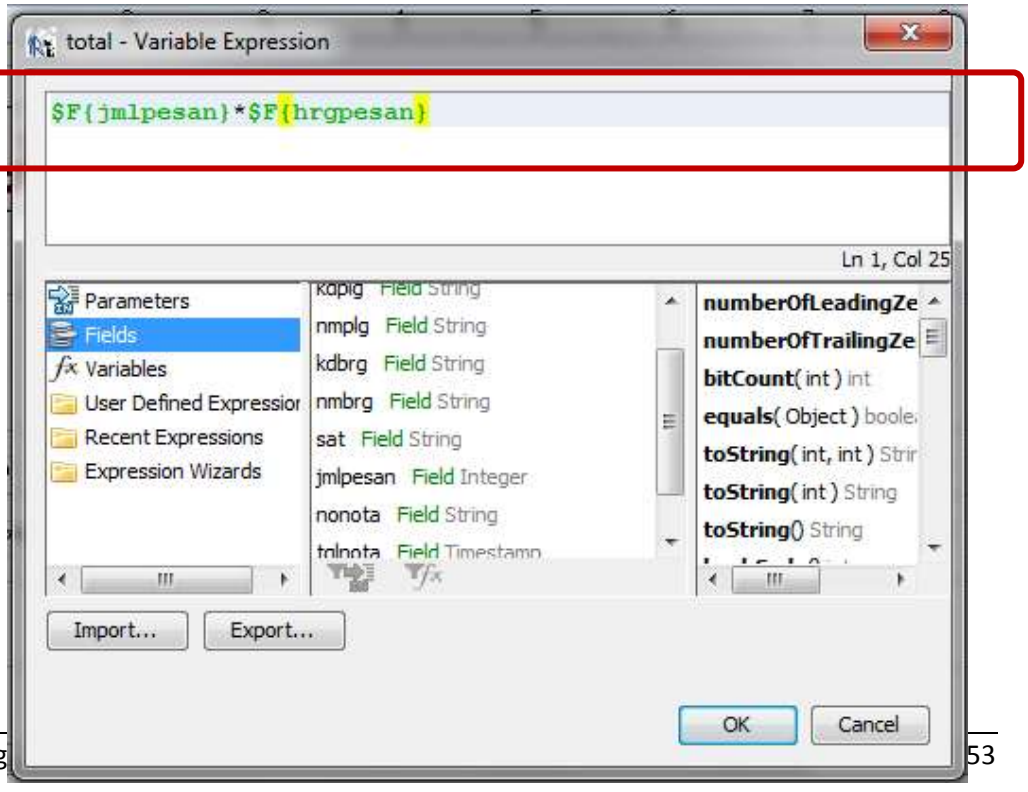
17) Ganti nama variabel tersebut menjadi total.



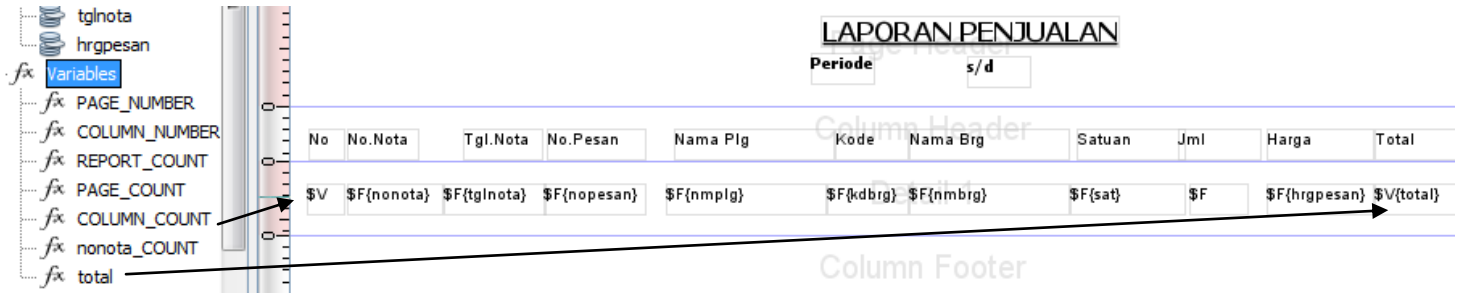
18) Pastikan variabel total dalam keadaan terpilih, lakukan perubahan pada **total-properties**. **Variabel class** pilih **java.lang.integer**. **Calculating** pilih **Nothing**, kemudian klik **Variable Expression** pada tombol yang sudah disediakan.



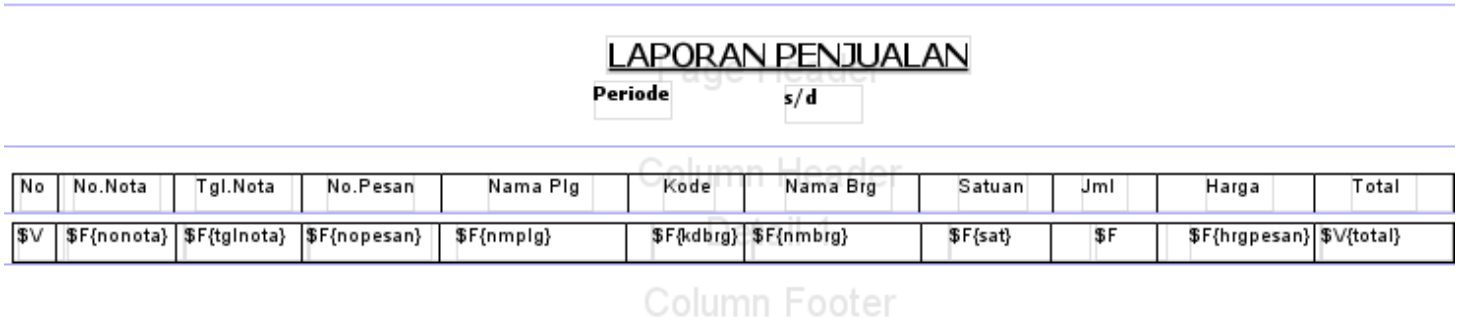
19) Pada kotak dialog **total-variable expression**, tambahkan perintah dengan cara mendouble klik **jmlpesan** dikali (*) **hrgpesan**, kemudian klik **OK**.



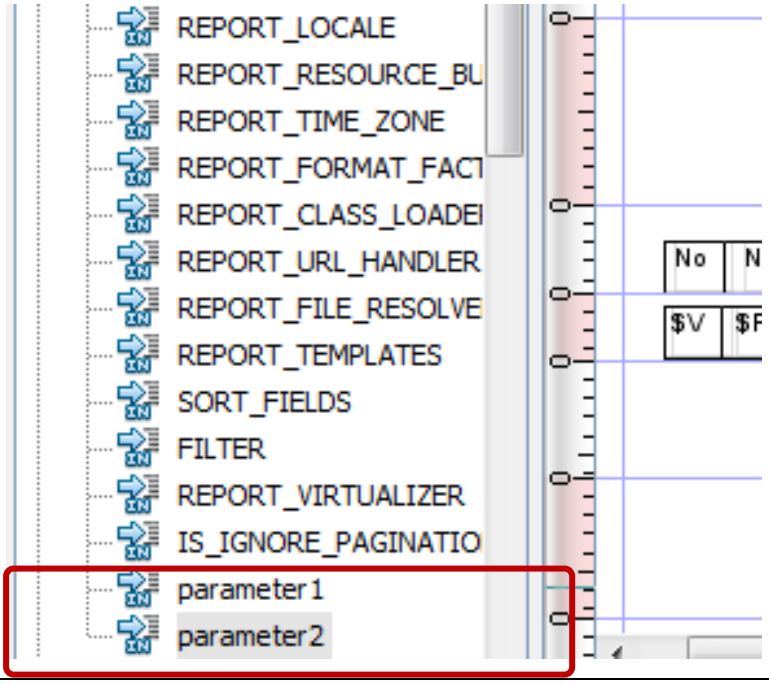
20) Masukkan dua variabel yang diperlukan kedalam report pada tempat yang sudah disediakan, variabel tersebut **Column-Count** dan **total**, atur posisi variabel tersebut seperti gambar berikut :



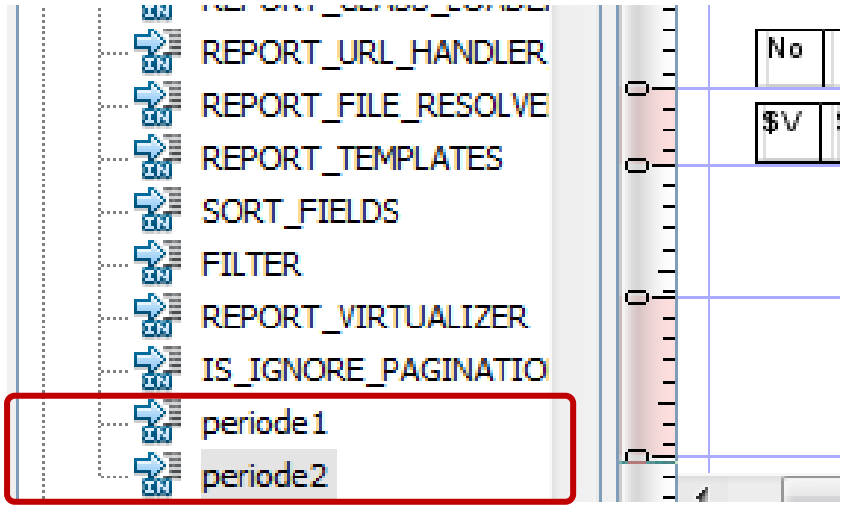
21) Tambahkan garis menggunakan objek **Line** dan **Rectangle** dari **Palette**, tampilan report sebagai berikut :



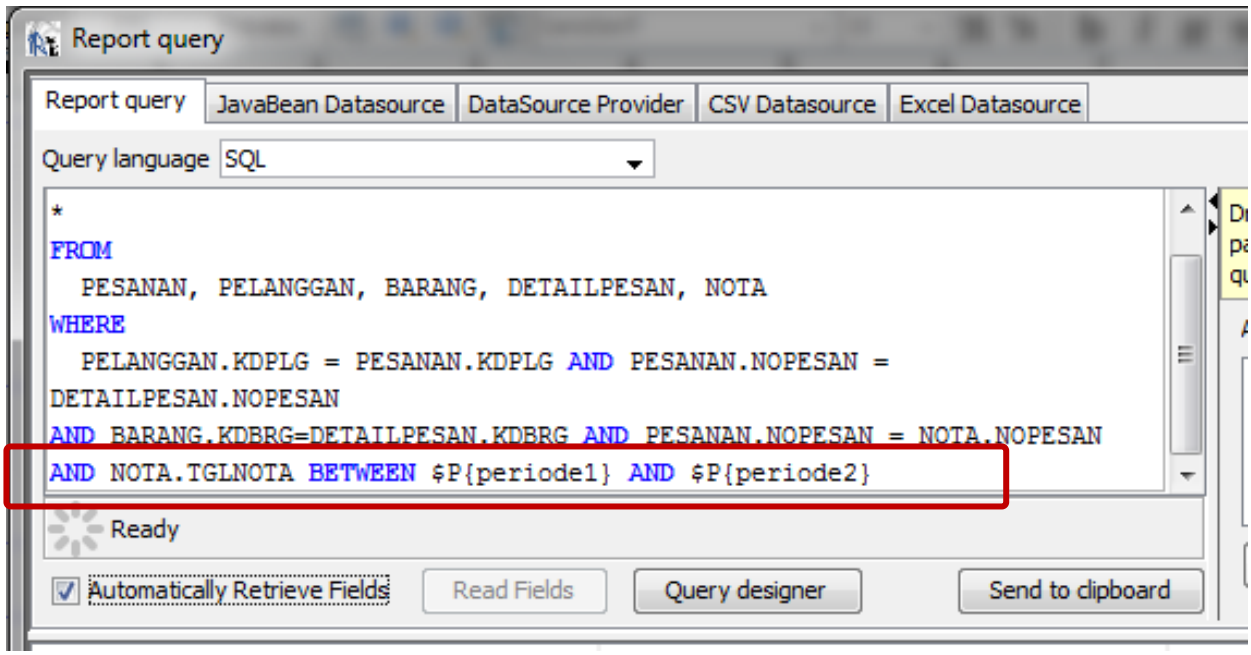
22) Untuk menampilkan data penjualan berdasarkan periode tertentu, maka perlu ditambahkan parameter kedalam report. Parameter ditambahkan dengan cara mengklik kanan **parameters** pada **Report Inspector**, kemudian pilih **Add Parameter**. Dalam laporan penjualan ini ada **dua parameter** yang ditambahkan sehingga tampilan parameternya sebagai berikut :



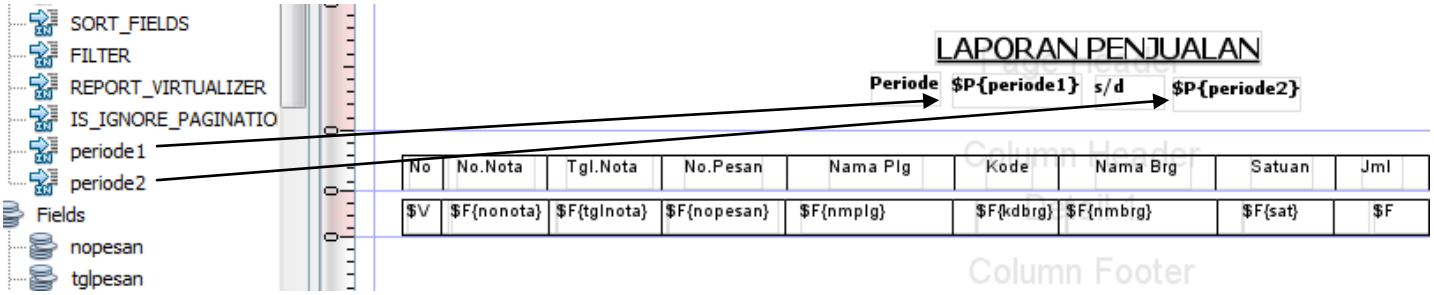
23) Ubah nama paremeter menjadi **periode1** dan **periode2**. Tampilan parameter sebagai berikut :



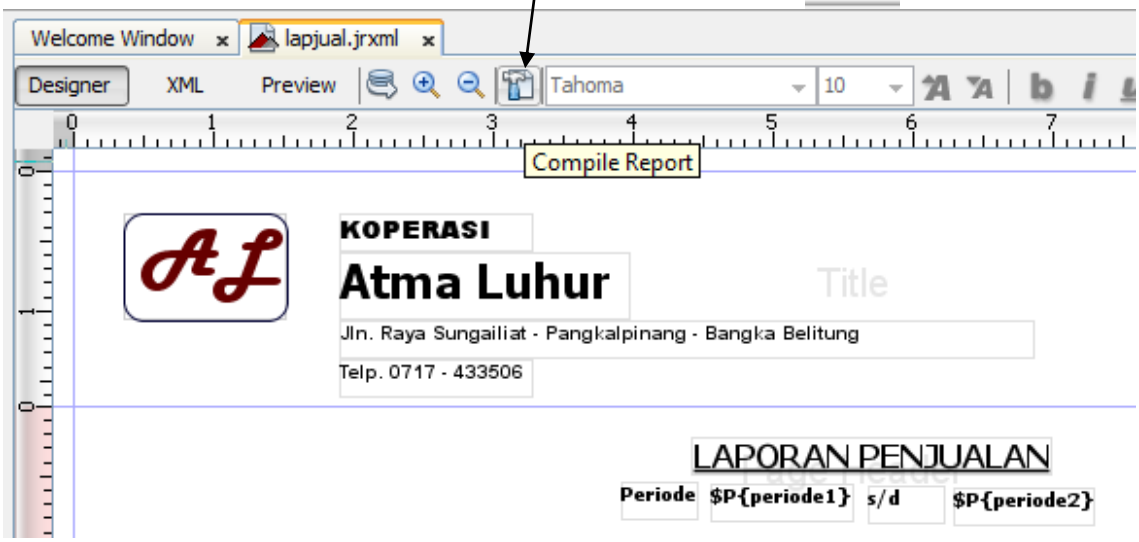
24) Edit query yang sudah dibuat sebelumnya dengan cara mengklik kanan pada **project** (**lapjual**) di **Report Inspector**. Kemudian pilih **Edit Query**. Tambahkan query seperti yang ditampilkan dalam **kotak segi empat**. Kemudian klik **OK**.



25) Tambahkan parameter **periode1** dan **periode2** ke report pada tempat yang sudah disediakan sebagai berikut :



26) Secara keseluruhan laporan penjualan sudah selesai dibuat. Repot laporan penjualan silahkan di COMPILE (**Compile Report**) untuk memastikan tidak ada kesalahan dan report silahkan disimpan/disave (



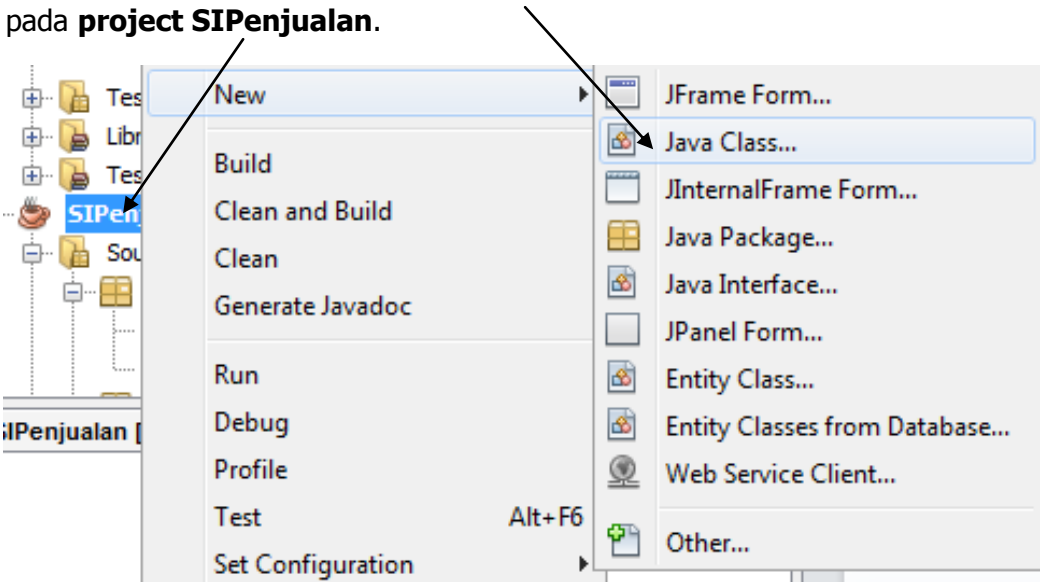
5. Integrasi Report di ireport dengan form/aplikasi di NetBeans.

Setelah selesai membuat dua report yaitu nota dan laporan penjualan, maka proses berikutnya membuat class dan form di netbeans. Hal ini dilakukan untuk menghubungkan report tersebut dengan form sehingga report dapat dibuka melalui menu utama/project utamanya. Form dan class yang akan dibuat masing-masing akan menghubungkan report nota dan laporan penjualan.

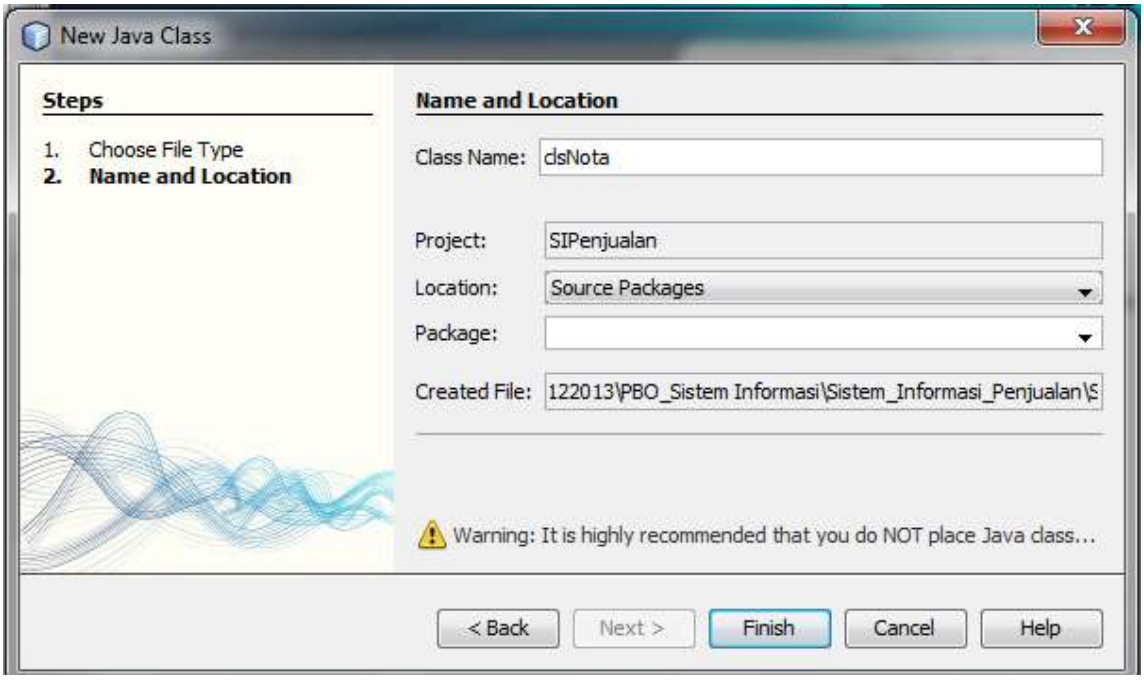
Langkah pertama yang dilakukan adalah buka project yang sebelumnya sudah dibuat di netbeans, yaitu project **SIPenjualan**.

a. Buat class nota

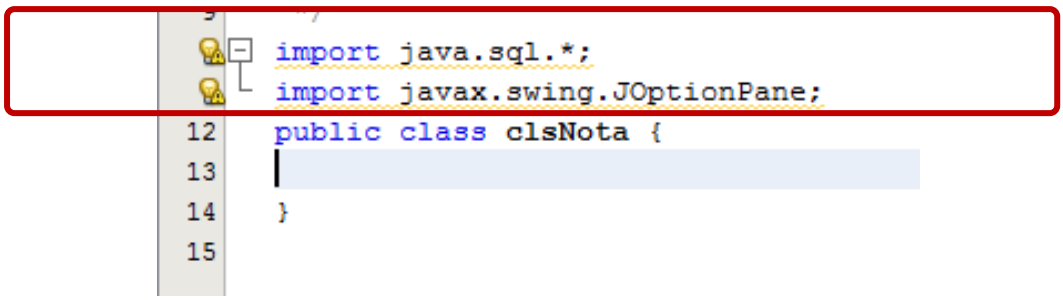
1) Tambahkan sebuah class baru (**Java Class**) dengan cara mengklik kanan pada **project SIPenjualan**.



2) Class tersebut beri dengan nama **clsNota**, kemudian klik **finish**.



3) Pada class tersebut, tambahkan sintaks untuk mengimport beberapa library yang diperlukan (**Listing program yang ditambahkan terdapat pada kotak segi empat**).



4) Didalam class **clsNota** tersebut, tambahkan perintah untuk membuat variabel/atribut dan methode yang diperlukan untuk menentukan atau mengirimkan nilai dari atau ke variabel tersebut. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;
public class clsNota {
  protected String nnota, npesan;
  protected Date tgnota;
  protected int flag;

  public void setNoNota(String n)
  { nnota = n;}
  public void setNoPesan(String np)
  { npesan = np;}
  public void setTglNota (Date t)
  { tgnota = t;}
  public String getNoNota()
  { return(nnota);}
  public String getNoPesan()
  { return(npesan);}
  public Date getTglNota()
  { return(tgnota);}

  public void setFlag(int f)
  { flag = f;}
  public int getFlag()
  { return(flag);}
}
```

5) Didalam class **clsNota** tersebut, tambahkan metode **simpanNota()** yang berfungsi untuk menyimpan data ke database. Metode ditambahkan dibawah metode **getFlag()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
public int getFlag()
{ return(flag);}
```

```
public void simpanNota()
{
    try
    {
        Koneksi k=new Koneksi();
        Connection cn=k.openKoneksi();
        Statement st=cn.createStatement();
        String sql="insert into nota values(";
               sql+="'"+getNoNota()+"','"+getTglNota()+"',";
               sql+="'"+getNoPesan()+"')";

          st.executeUpdate(sql);
          setFlag(1);
          st.close();
          cn.close();
          JOptionPane.showMessageDialog(null,
                  "Data sudah disimpan dan "
                  + "Nota akan Dicetak","SIMPAN",
                  JOptionPane.INFORMATION_MESSAGE);

    }
    catch(SQLException sqe)
    {
        JOptionPane.showMessageDialog(null, "Gagal Simpan",
                "Gagal Simpan",JOptionPane.ERROR_MESSAGE);
    }
}
}
```

6) Didalam class **clsNota** tersebut, tambahkan metode **autoNoNota()** yang berfungsi untuk menampilkan nomor nota secara otomatis. Metode ditambahkan dibawah metode **simpanNota()**. (**Listing program yang ditambahkan terdapat pada kotak segi empat**).

```
            JOptionPane.showMessageDialog(null, "Gagal Simpan",
                    "Gagal Simpan",JOptionPane.ERROR_MESSAGE);
        }
    }
```

```
    public void autoNoNota()
    {
        try
        {
            int hit=0;

            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select count(nonota) from nota";
            ResultSet rs=st.executeQuery(sql);

            if(rs.next())
            {
                if(Integer.parseInt(rs.getString(1))==0)
                {
                    setNoNota("NT001");
                    st.close();
                    cn.close();
                }
                else
                {
                    sql="select Max(mid(nonota,3,3)) from nota";
                    rs = st.executeQuery(sql);
                    rs.next();
                    hit = (Integer.parseInt(rs.getString(1)))+1;
                    if(hit<10)
                    { setNoNota("NT00"+hit);}
                    else if(hit<100)
                    { setNoNota("NT0"+hit);}
                    else
                    { setNoNota("NT"+hit);}
                    st.close();
                    cn.close();
                }
            }
        }
        catch(SQLException sqe)
        {}
    }
```

Bah

7) Secara keseluruhan class clsNota sudah selesai dibuat. Project dapat disimpan ulang/disave ( ).

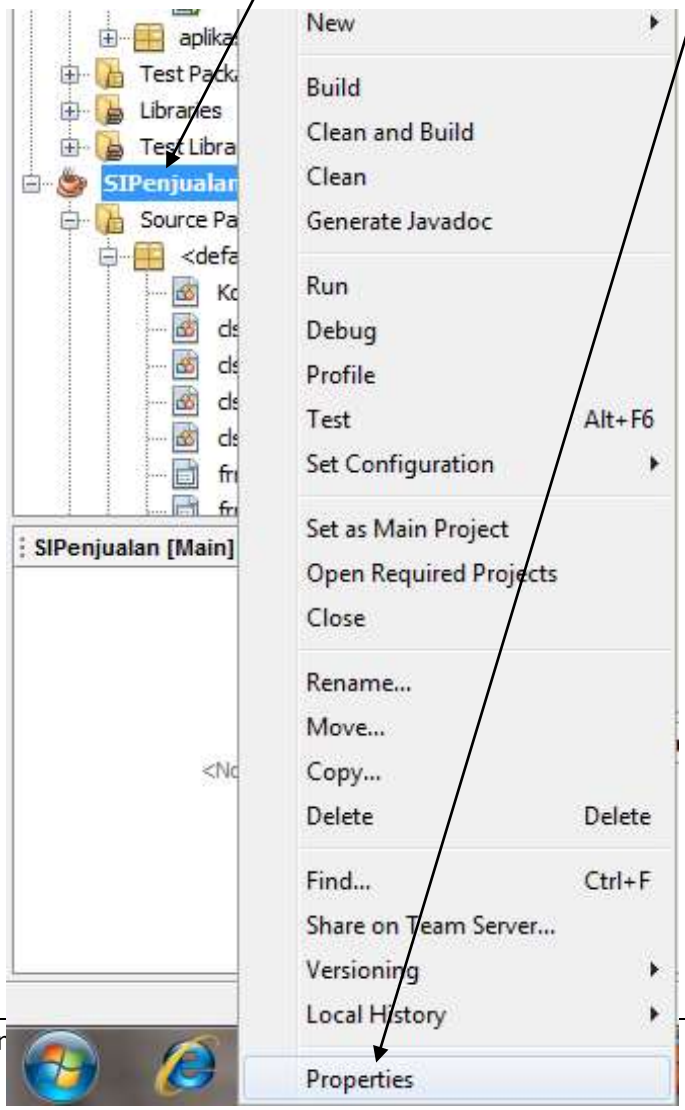
b. Buat Form Nota

Form nota berfungsi untuk memanggil /mencetak report (**nota**) yang telah dibuat di ireport, sehingga perlu untuk mengimport beberapa library yang diperlukan kedalam project (SIPenjualan). Library yang diperlukan (disesuaikan dengan versi ireportnya) adalah :

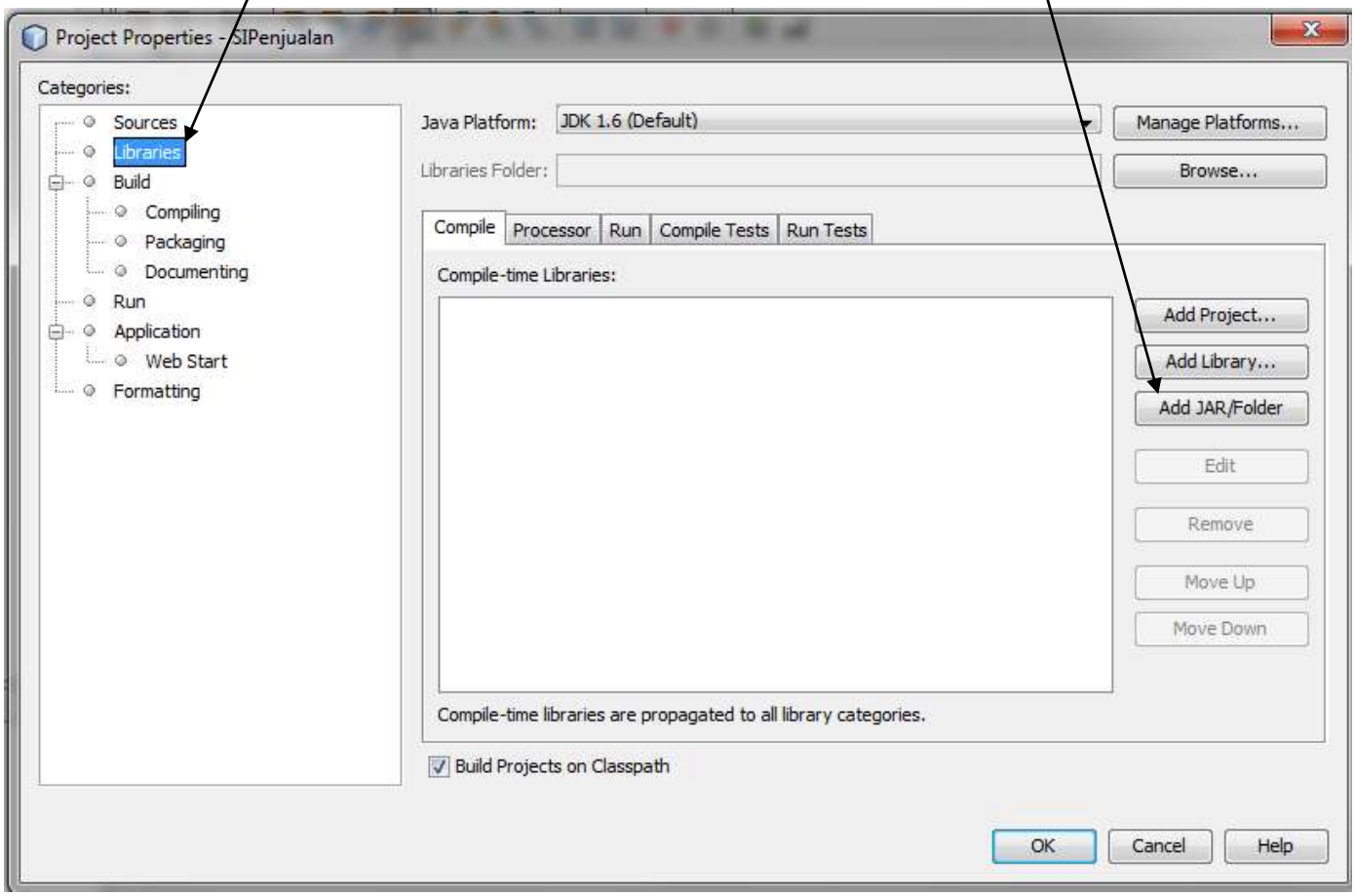
1) commons-beanutils-1.8.2.jar
2) commons-collections-3.2.1.jar
3) commons-digester-2.1.jar
4) commons-logging-1.1.jar
5) jasperreports-5.0.1.jar
6) groovy-all-1.7.5.jar
7) jdt-compiler-3.1.1.jar

Adapun beberapa library tersebut sudah disediakan pada direktori khusus tempat lokasi ireport diinstall. Library tersebut biasanya terdapat pada **direktori ireport\modules\ext**. Berikut langkah-langkah mengimport library tersebut dilanjutkan membuat form nota.

1) Klik kanan pada project **SIPenjualan** kemudian pilih **Properties**.



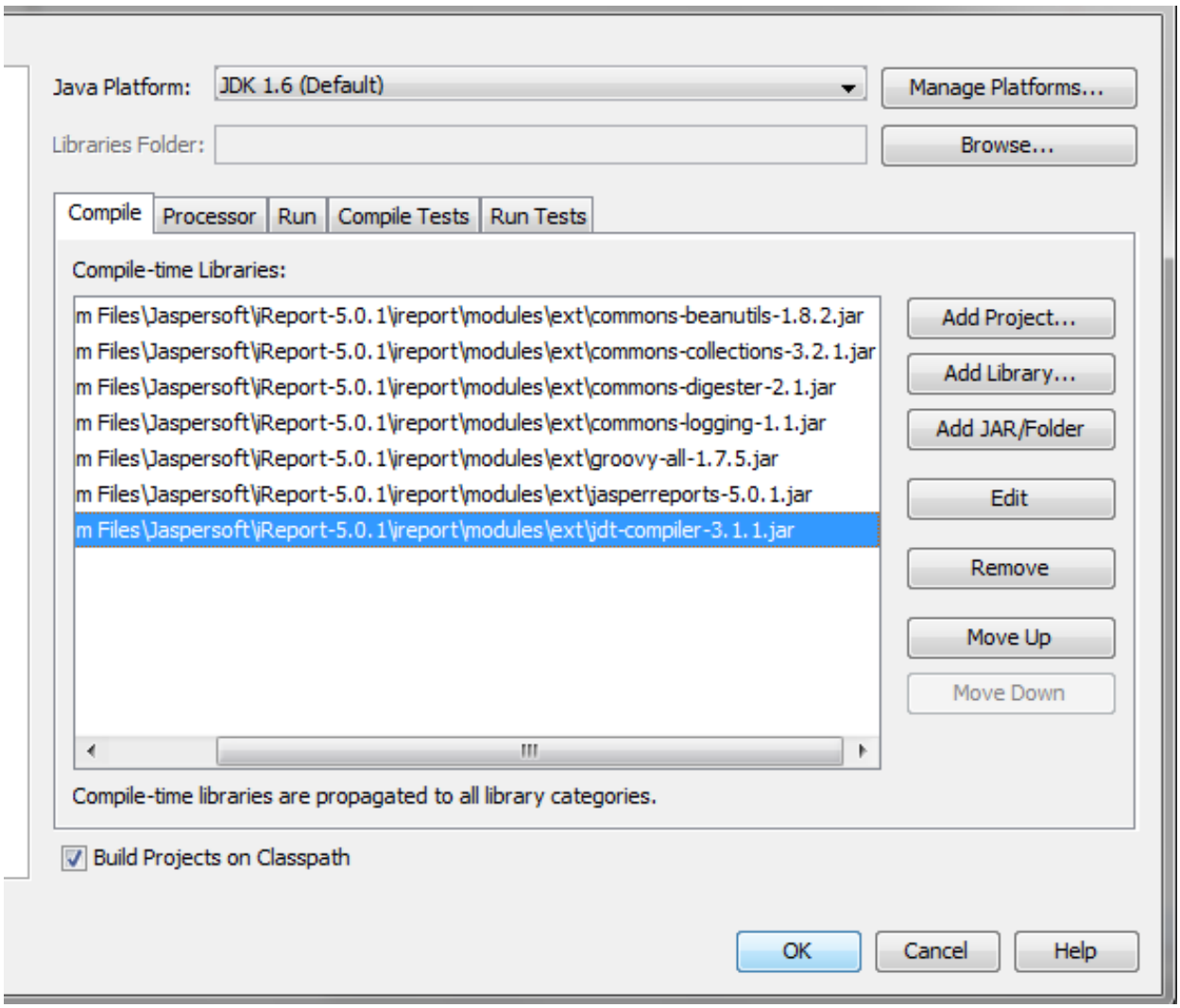
2) Pada kotak dialog **Project Properties**, pada bagian **Categories**, pastikan **Libraries** dalam keadaan terpilih kemudian klik tombol **Add JAR/Folder**.



3) Cari lokasi ireport yang sudah diinstal. Pada contoh dimodul ini ireport diinstal di **C:\Program Files\Jaspersoft\iReport-5.0.1**. Apabila sudah lokasinya sudah diketahui, diteruskan dengan membuka folder : **ireport\modules\ext**. Sehingga alamat lengkap untuk mencari librarynya adalah :

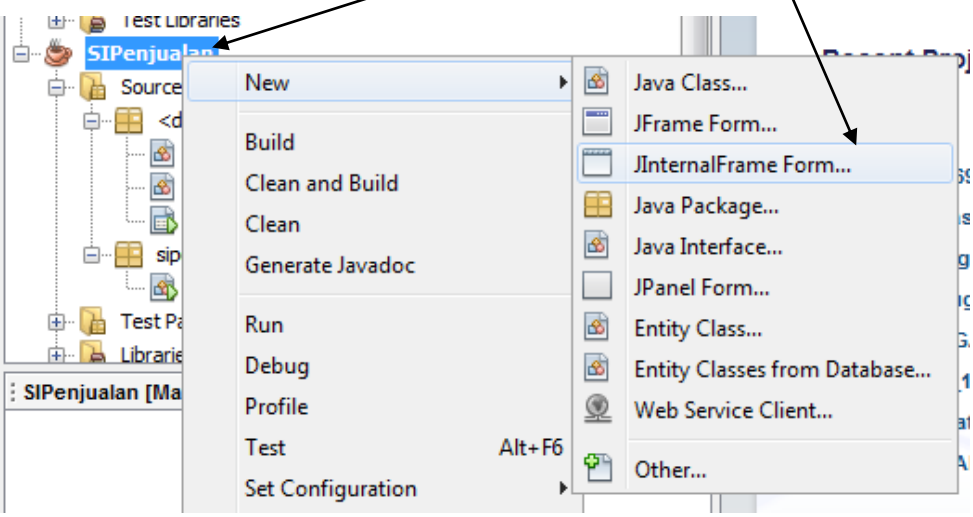
   C:\Program Files\Jaspersoft\iReport-5.0.1 ireport\modules\ext.

4) Pilih library yang diperlukan seperti yang disebutkan diatas. Apabila sudah silahkan klik OK. Sehingga tampilan **project properties** sebagai berikut :

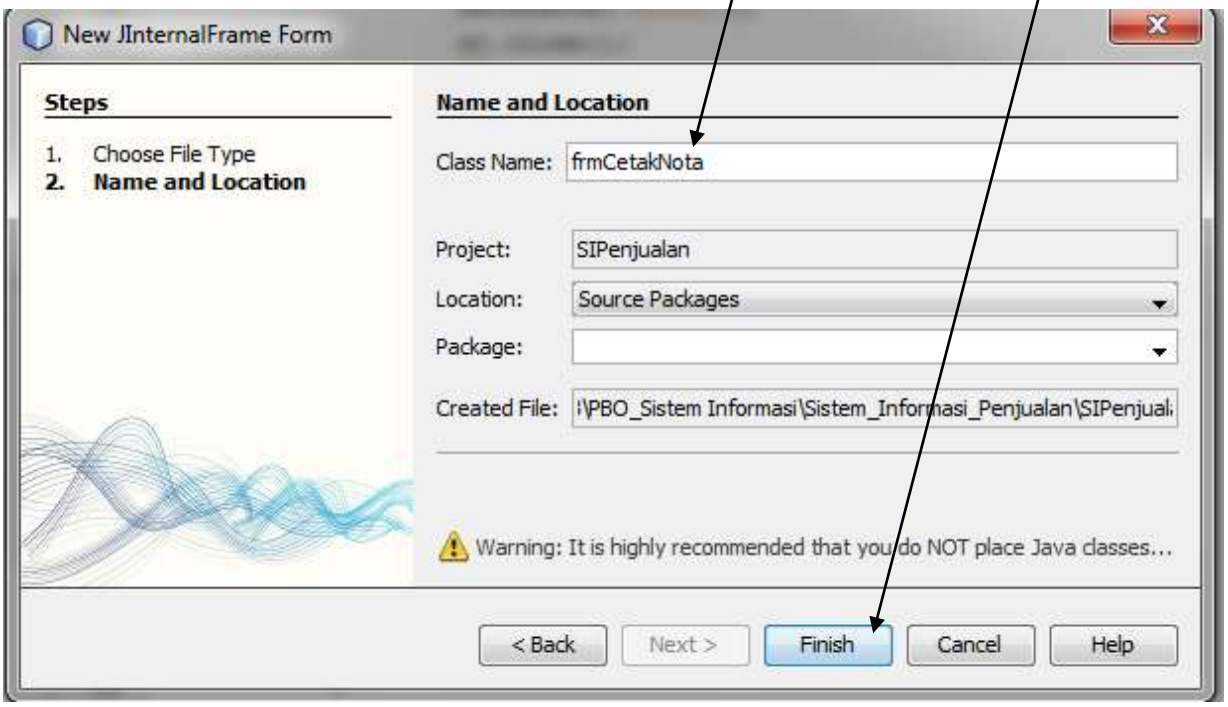


Klik tombol **OK** untuk menutup kotak dialog tersebut.

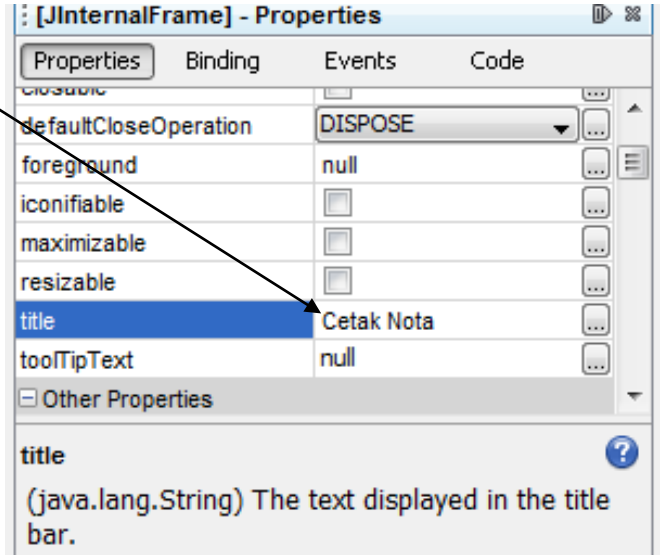
5) Tambahkan sebuah form baru (**JInternalFrame Form**), dengan cara mengklik kanan pada project **SIPenjualan**.



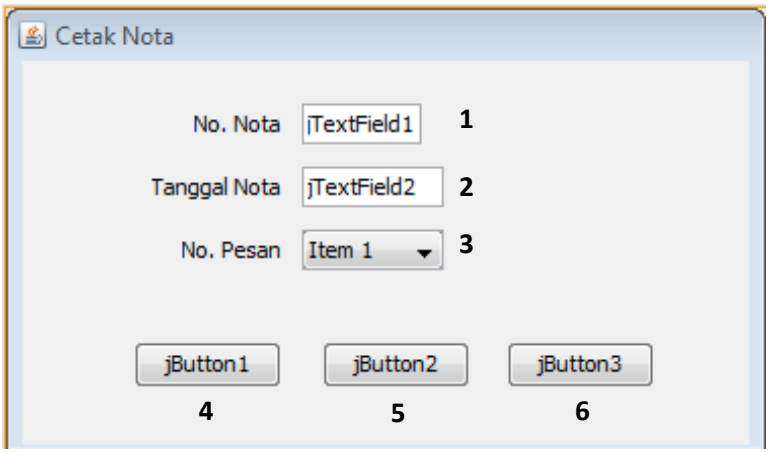
6) Form baru tersebut beri dengan nama **frmCetakNota**, kemudian klik **Finish**.



7) Beri judul form tersebut pada kolom yang sudah disediakan dengan nama : **Cetak Nota**.

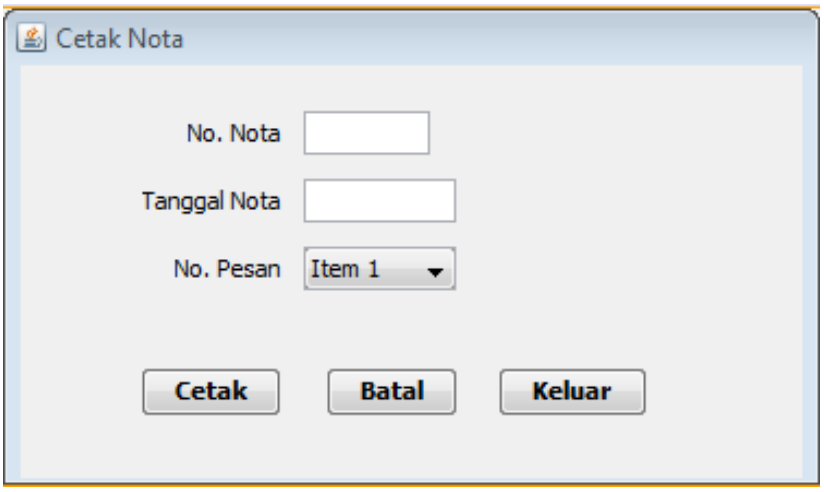


8) Atur ukuran form dan tambahkan objek seperti gambar berikut :



9) Berdasarkan form tersebut, ubah nama text dan nama variabelnya sesuai dengan penomorannya sebagai berikut :
   - **1. jTextField1**, Text dihapus, change variable name = **txtNoNota**.
   - **2. jTextField2**, Text dihapus, change variable name = **txtTglNota**.
   - **3. jComboBox1**, change variable name = **cmbNoPesan**.
   - **4. jButton1**, Text = **Cetak**, change variable name = **btnCetak.**
   - **5. jButton2**, Text = **Batal**, change variable name = **btnBatal.**
   - **6. jButton3**, Text = **Keluar**, change variable name = **btnKeluar.**

   Sehingga tampilan akhir form sebagai berikut :



10) Klik tabulasi **source** dan tambahkan perintah berikut untuk mengimport beberapa library yang diperlukan dari packagenya. (**Sintaks program yang ditambahkan terdapat pada kotak segi empat**)

```
import java.sql.*;
import javax.swing.JOptionPane;

//#Import tambahan untuk Report
import net.sf.jasperreports.engine.*;
import net.sf.jasperreports.engine.util.*;
import net.sf.jasperreports.view.*;

import java.util.*;
import java.io.*;

public class frmCetakNota extends javax.swing.JInternalFrame {

    /** Creates new form frmCetakNota */
    public frmCetakNota() {
        initComponents();
```

11) Tambahkan methode **kosong()** dibawah konstruktor class tersebut. Methode tersebut berfungsi untuk membersihkan komponen dari data yang telah dinput/diolah sebelumnya sekaligus sebagai langkah awal untuk proses berikutnya. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
    /** This method is called from within the constructc
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code
    public void kosong()
    {
        clsNota nt = new clsNota();

        txtNoNota.setText("");
        txtTglNota.setText("");
        cmbNoPesan.removeAllItems();
        btnCetak.setEnabled(false);
        nt.autoNoNota();
        txtNoNota.setText(nt.getNoNota());
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnBatal;
    private javax.swing.JButton btnCetak;
    private javax.swing.JButton btnKeluar;
```

12) Tambahkan methode **comboPesanan()** yang berfungsi untuk menampilkan data ke objek yang sudah disiapkan di form. Methode ini ditambahkan dibawah methode **kosong()**. (**Baris perintah/methode yang ditambahkan terdapat pada kotak segi empat**).

```
        nt.autoNoNota();
        txtNoNota.setText(nt.getNoNota());
    }

    public void comboPesanan()
    {
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            Statement st=cn.createStatement();
            String sql="select nopesan from pesanan ";
                   sql+="where nopesan not in (";
                   sql+="select nopesan from nota)";
            ResultSet rs = st.executeQuery(sql);
           while(rs.next())
           {
               cmbNoPesan.addItem(rs.getString("nopesan"));
           }
           st.close();
           cn.close();
        }
        catch(SQLException sqe)
        {}
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnBatal;
    private javax.swing.JButton btnCetak;
    private javax.swing.JButton btnKeluar;
```

Perhatikan : antara pesanan dan tanda petik ganda (“) dikasih jarak / di spasi !!!!!. contoh : pesanan” (SALAH), pesanan ” (BETUL).

13) Kembali ke konstruktor **frmCetakNota**, tambahkan perintah berikut untuk memanggil methode **kosong()**, **comboPesanan()**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).

```
public class frmCetakNota extends javax.swing.JInternalFrame {

    /** Creates new form frmCetakNota */
    public frmCetakNota() {
        initComponents();

        kosong();
        comboPesanan();
    }
```

14) Tambahkan perintah melalui event **actionPerformed** pada object **btnCetak** (Tombol **Cetak**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat)**.

```
    private void btnCetakActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        clsNota nt=new clsNota();
        nt.setNoNota(txtNoNota.getText());
        nt.setTglNota(java.sql.Date.valueOf(txtTglNota.getText()));
        nt.setNoPesan(""+cmbNoPesan.getSelectedItem());
        nt.simpanNota();
        try
        {
            Koneksi k=new Koneksi();
            Connection cn=k.openKoneksi();
            String nmFile="D:/okki.data.data/Materi Ajar/Java/Genap 20122013/";
                   nmFile+="PBO_Sistem Informasi/Sistem_Informasi_Penjualan/";
                   nmFile+="Cetak/nota.jasper";
            String nont = txtNoNota.getText();
            HashMap map=new HashMap();
            map.put("nomornota", nont);

            File reportFile = new File(nmFile);
JasperReport jReport = (JasperReport)JRLoader.loadObject(reportFile.getPath());
            JasperPrint jPrint = JasperFillManager.fillReport(jReport, map,cn);
            JasperViewer.viewReport(jPrint,false);
            JasperViewer.setDefaultLookAndFeelDecorated(true);

        }
        catch(Exception sqe)
        {
            JOptionPane.showMessageDialog(null, "Nota tidak dapat dicetak"+
                    sqe.getMessage(),"Cetak Nota",
                    JOptionPane.ERROR_MESSAGE);
        }
        kosong();
        comboPesanan();
    }
```

Perlu diperhatikan **nmFile** berisi lokasi tempat menyimpan report Nota. Apabila berbeda, sesuaikan dengan lokasi tempat penyimpanan yang ditentukan sendiri.

nomornota merupakan parameter yang dibuat pada saat membuat report Nota. Apabila berbeda, sesuaikan dengan nama parameter yang dibuat sendiri.

15) Tambahkan perintah melalui event **actionPerformed** pada object **btnBatal** (Tombol Batal) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat).**

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    kosong();
    comboPesanan();
}
```

16) Tambahkan perintah melalui event **mouseClicked** pada object **cmbNoPesan** (combo box No Pesan) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat).**

```
private void cmbNoPesanMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    btnCetak.setEnabled(true);
    btnCetak.requestFocus();
}
```

17) Tambahkan perintah melalui event **actionPerformed** pada object **btnKeluar** (Tombol Keluar) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat).**

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

18) Secara keseluruhan form Cetak Nota sudah selesai dibuat, project silahkan disimpan ulang/disave ().

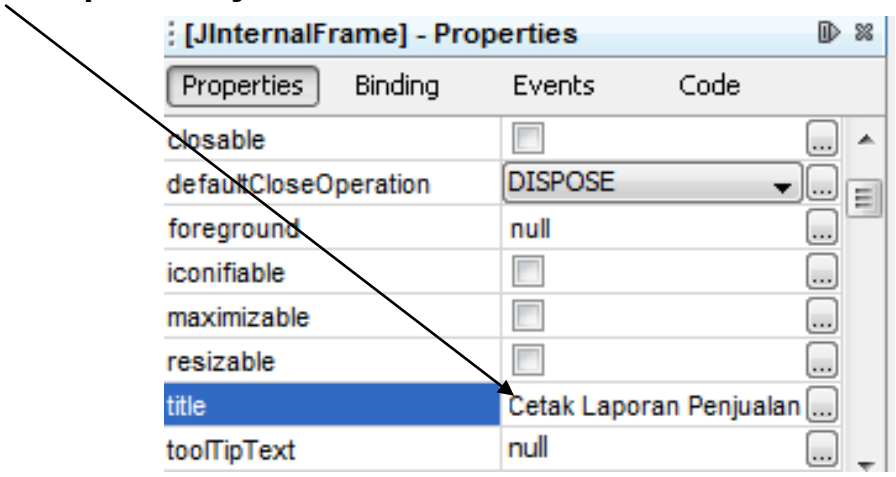
c. Buat Form Laporan Penjualan

Apabila library yang diperlukan untuk menghubungkan report ke form belum diimport ke project, maka silahkan ikuti langkah-langkah untuk mengimport library ireport. Langkah-langkahnya dapat dilihat pada proses pembuatan form nota diatas.

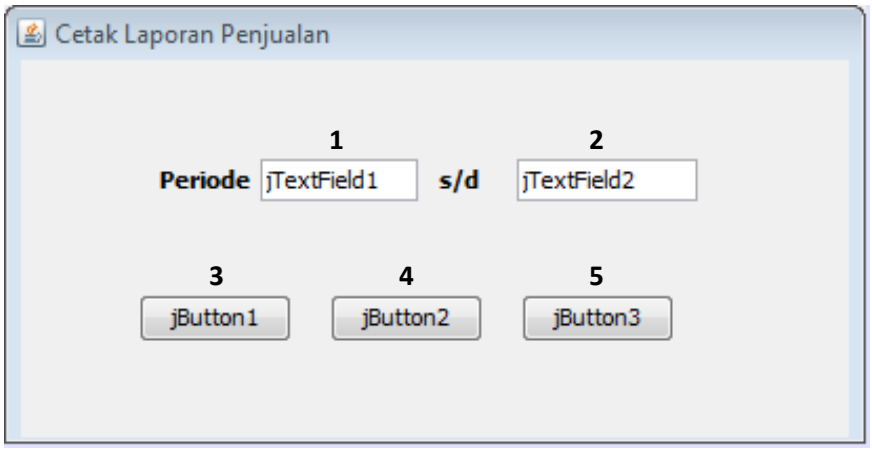
Berikut langkah-langkah membuat form laporan penjualan :

1) Tambahkan sebuah form baru (**JInternalFrame Form**), dengan cara mengklik kanan pada project **SIPenjualan.**
2) Form baru tersebut beri dengan nama **frmCetakLapJual**, kemudian klik **Finish**.

3) Beri judul form tersebut pada kolom yang sudah disediakan dengan nama : **Cetak Laporan Penjualan**.

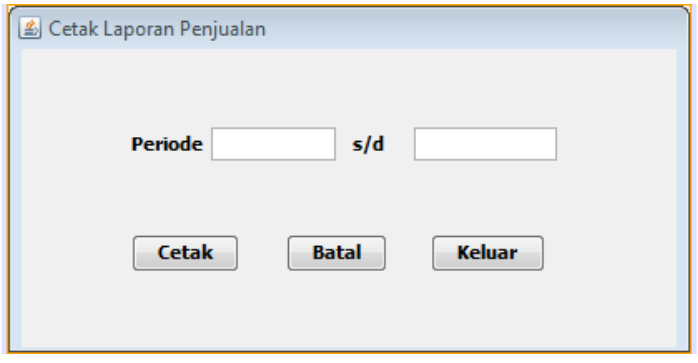


4) Atur ukuran form dan tambahkan objek seperti gambar berikut :



5) Berdasarkan form tersebut, ubah nama text dan nama variabelnya sesuai dengan penomorannya, sebagai berikut :
   - **1. jTextField1**, Text dihapus, change variable name = **txtPeriode1**.
   - **2. jTextField2**, Text dihapus, change variable name = **txtPeriode2**.
   - **3. jButton1**, Text = **Cetak**, change variable name = **btnCetak.**
   - **4. jButton2**, Text = **Batal**, change variable name = **btnBatal.**
   - **5. jButton3**, Text = **Keluar**, change variable name = **btnCetak.**

   Sehingga tampilan akhir form sebagai berikut :



6) Klik tabulasi **source** dan tambahkan perintah berikut untuk mengimport beberapa library yang diperlukan dari packagenya. (**Sintaks program yang ditambahkan terdapat pada kotak segi empat**).

```
import java.sql.*;
import javax.swing.JOptionPane;

//#import tambahan untuk report
import net.sf.jasperreports.engine.*;
import net.sf.jasperreports.engine.util.*;
import net.sf.jasperreports.view.*;

import java.util.*;
import java.io.*;

public class frmCetakLapJual extends javax.swing.JInternalFrame {

    /** Creates new form frmCetakLapJual */
```

7) Tambahkan perintah melalui event **actionPerformed** pada object **btnCetak** (Tombol **Cetak**) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat).**

```
private void btnCetakActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try
    {
        Koneksi k=new Koneksi();
        Connection cn=k.openKoneksi();
        String nmFile="D:/okki.data.data/Materi Ajar/Java/Genap 20122013/";
               nmFile+="PBO_Sistem Informasi/Sistem_Informasi_Penjualan/";
               nmFile+="Cetak/lapjual.jasper";
        String tperiode1 = txtPeriode1.getText();
        String tperiode2 = txtPeriode2.getText();

        HashMap map=new HashMap();
        map.put("periode1", tperiode1);
        map.put("periode2", tperiode2);

        File reportFile = new File(nmFile);
JasperReport jReport = (JasperReport)JRLoader.loadObject(reportFile.getPath());
        JasperPrint jPrint = JasperFillManager.fillReport(jReport, map,cn);
        JasperViewer.viewReport(jPrint,false);
        JasperViewer.setDefaultLookAndFeelDecorated(true);
    }
    catch(Exception sqe)
    {
        JOptionPane.showMessageDialog(null, "Laporan tidak dapat dicetak"+
                sqe.getMessage(),"Cetak Laporan",
                JOptionPane.ERROR_MESSAGE);
    }
}
```

9) Tambahkan perintah melalui event **actionPerformed** pada object **btnBatal** (Tombol Batal) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat).**

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    txtPeriode1.setText("");
    txtPeriode2.setText("");
    txtPeriode1.requestFocus();
}
```

10) Tambahkan perintah melalui event **actionPerformed** pada object **btnKeluar** (Tombol Keluar) dengan cara klik kanan pada object tersebut. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat).**

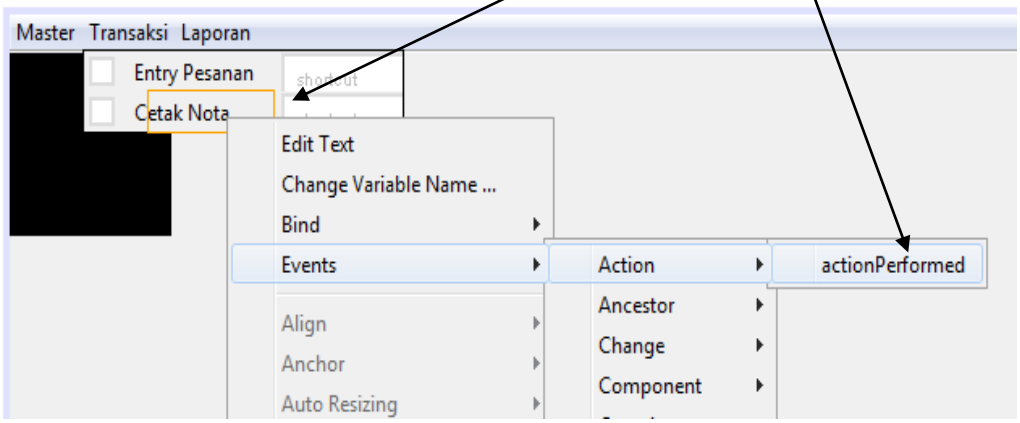
```
private void btnKeluarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    setVisible(false);
}
```

11) Secara keseluruhan form Cetak Laporan sudah selesai dibuat, project silahkan disimpan ulang/disave ( .

d. Integrasi Form Nota Dan Laporan Penjualan Ke Menu Utama

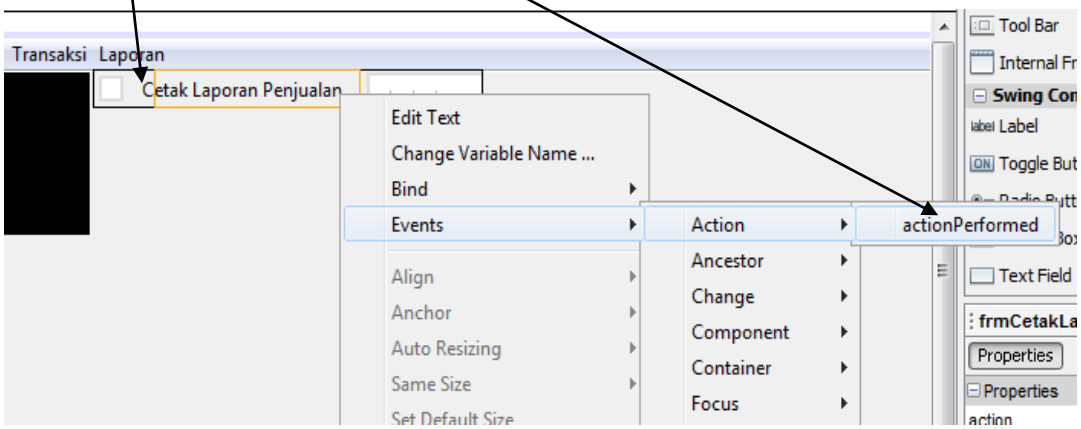
Langkah-langkahnya sebagai berikut :

1) Buka form **menu utama**. Form dapat dibuka dengan cara mendouble klik form tersebut pada jendela **project**. Pastikan tabulasi **design** dalam keadaan terpilih.
2) Klik tabulasi **design**. Tambahkan perintah pada sub menu **Cetak Nota** untuk menghubungkan ke class/form Cetak Nota. Perintah ditambahkan dengan cara klik kanan pada sub menu tersebut dan melalui event **actionPerformed**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).



```
private void frmNotaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    frmCetakNota frmnota=new frmCetakNota();

    dskPane.add(frmnota);
    frmnota.setVisible(true);
}
```

3) Klik tabulasi **design**. Tambahkan perintah pada sub menu **Cetak Laporan Penjualan** untuk menghubungkan ke class/form Cetak Nota. Perintah ditambahkan dengan cara klik kanan pada sub menu tersebut dan melalui event **actionPerformed**. (**Baris perintah yang ditambahkan terdapat pada kotak segi empat**).




```
private void frmCetakLaporanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    frmCetakLapJual frmlaporan = new frmCetakLapJual();

    dskPane.add(frmlaporan);
    frmlaporan.setVisible(true);
}
```

4) Integrasi form cetak nota dan laporan penjualan sudah selesai. Project dapat disimpan ulang/save ().

Setelah selesai mengintegrasikan form cetak nota dan laporan, maka seluruh form dan class sudah lengkap dan secara keseluruhan project SIPenjualan sudah selesai dibuat. Selamat Mencoba.

DAFTAR PUSTAKA

cariprogram.blogspot.com/2012/08/menampilkan-jasperreport-ireport-dengan-java-netbeans.html, diakses tanggal 13 pebruari 2013

community.jaspersoft.com/project/ireport-designer
diakses tanggal 11 pebruari 2013

Diktat Mata Kuliah Pemrograman Berorientasi Objek, M. Anif – Jati Lestari ,2009

hendrowijaya.com/menampilkan-data-jtable-ke-jtextfield.html

macintel.net/2012/06/mengisi-data-jcombobox-netbeans-dari-database-mysql

mdsaputra.wordpress.com/, diakses tanggal 11 maret 2013

Membuat aplikasi database dengan Java, MySQL dan Netbeans, Miftakhul Huda dan Bunafit Nugroho, Elex Media Komputindo, 2010

Membangun GUI dengan Java Netbeans 6.5, ANDI – Wahana Komputer, 2010.

Menghubungkan Database Access Menggunakan Netbeans, Yuliana Setiowati, eepis-its.edu, diakses agustus 2012

mrhandsblog.blogspot.com/2011/07/membuat-report-di-java-dengan-ireport.html
diakses tanggal 11 pebruari 2013

Sistem Informasi Penjualan dengan JAVA, Ir. Yuniar Supardi, Elex Media Komputindo, 2008

sourceforge.net/projects/ireport/

www.nagasakti.or.id/roller/ifnu, Materi pelatihan java swing, ifnu bima, diakses agustus 2012