

# Pertemuan 5

## Sequence Diagram

# Sequence Diagram

- Sequence diagram mengilustrasikan objek-objek yang berpartisipasi di dalam suatu use case.
- Sequence diagram menunjukkan pesan yang lewat di antara objek untuk use case tertentu dari waktu ke waktu.

# Elemen-Elemen Sequence Diagram



anActor

`<<actor>>`  
anActor

- Actor
  - Adalah orang atau sistem yang memperoleh manfaat dari dan berada di luar sistem.
  - Berpartisipasi dalam suatu urutan dengan mengirim dan / atau menerima pesan.
  - Ditempatkan di bagian atas diagram.

# Elemen-Elemen Sequence Diagram

anObject : aClass

- **Object**

- Berpartisipasi dalam suatu urutan dengan mengirim dan / atau menerima pesan.
- Ditempatkan di bagian atas diagram.

## Penggambaran lain dari object



Boundary Class: Menggambarkan tampilan program



Control Class: Menggambarkan controller



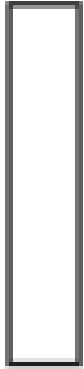
Entity Class: Menggambarkan class

# Elemen-Elemen Sequence Diagram

- Lifeline

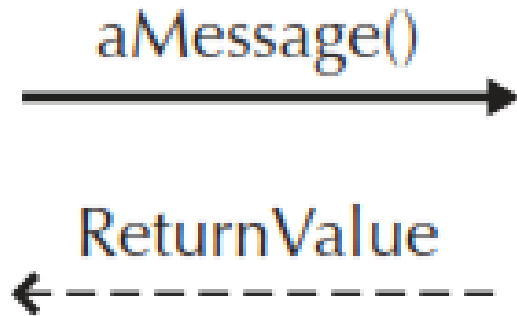
- Menunjukkan kehidupan suatu objek selama suatu urutan.
- Berisi X pada titik di mana kelas tidak lagi berinteraksi.

# Elemen-Elemen Sequence Diagram



- Execution Occurrence (Kejadian eksekusi)
  - Merupakan persegi panjang sempit panjang yang ditempatkan di atas lifeline.
  - Menunjukkan kapan suatu objek mengirim atau menerima pesan.

# Elemen-Elemen Sequence Diagram



- Message
  - Menyampaikan informasi dari satu objek ke objek lainnya.
  - Pemanggilan operasi diberi label dengan pesan yang dikirim dan panah padat, sedangkan pengembalian diberi label dengan nilai yang dikembalikan dan ditampilkan sebagai tanda panah putus-putus.

# Elemen-Elemen Sequence Diagram

[aGuardCondition]:aMessage()

- Guard Condition
  - Merupakan tes yang harus dipenuhi untuk pesan yang akan dikirim.



# Elemen-Elemen Sequence Diagram



- Object Destruction
  - X ditempatkan di ujung lifeline objek untuk menunjukkan bahwa objek tersebut akan keluar dari eksistensi.

# Elemen-Elemen Sequence Diagram



- Frame
  - Menunjukkan konteks sequence diagram

# Metode MVC

- **Model-View-Controller (MVC)** adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data (Model) dari tampilan (View) dan cara bagaimana memprosesnya (Controller).

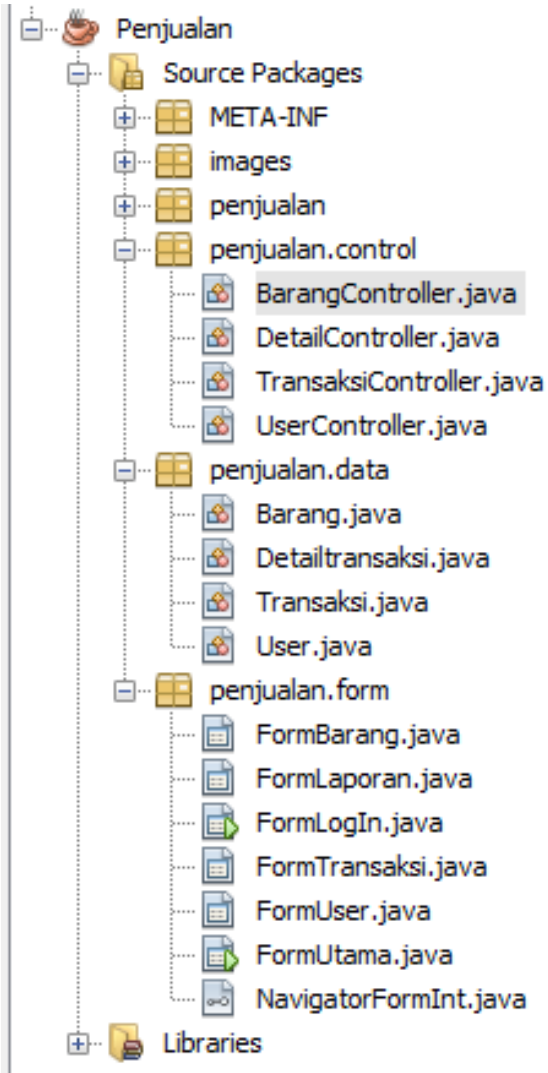
# Metode MVC

- **Model** mewakili struktur data.
- **View** adalah bagian yang mengatur tampilan ke pengguna.
- **Controller** merupakan bagian yang menjembatani model dan view. Controller berisi perintah-perintah yang berfungsi untuk memproses suatu data dan mengirimkannya ke tampilan program.

# Contoh Penerapan MVC

- Sebuah database penjualan memiliki 4 tabel:
  - Barang
  - User
  - Transaksi
  - Detail transaksi
- Masing-masing tabel dibuatkan MVC-nya:
  - Model berisi variabel sesuai nama field dari tabel beserta fungsi.
  - View berupa tampilan form program (kecuali detail transaksi).
  - Controller berisi coding yang berhubungan dengan database.

# Contoh Penerapan MVC



**Controller**

**Model**

**View**

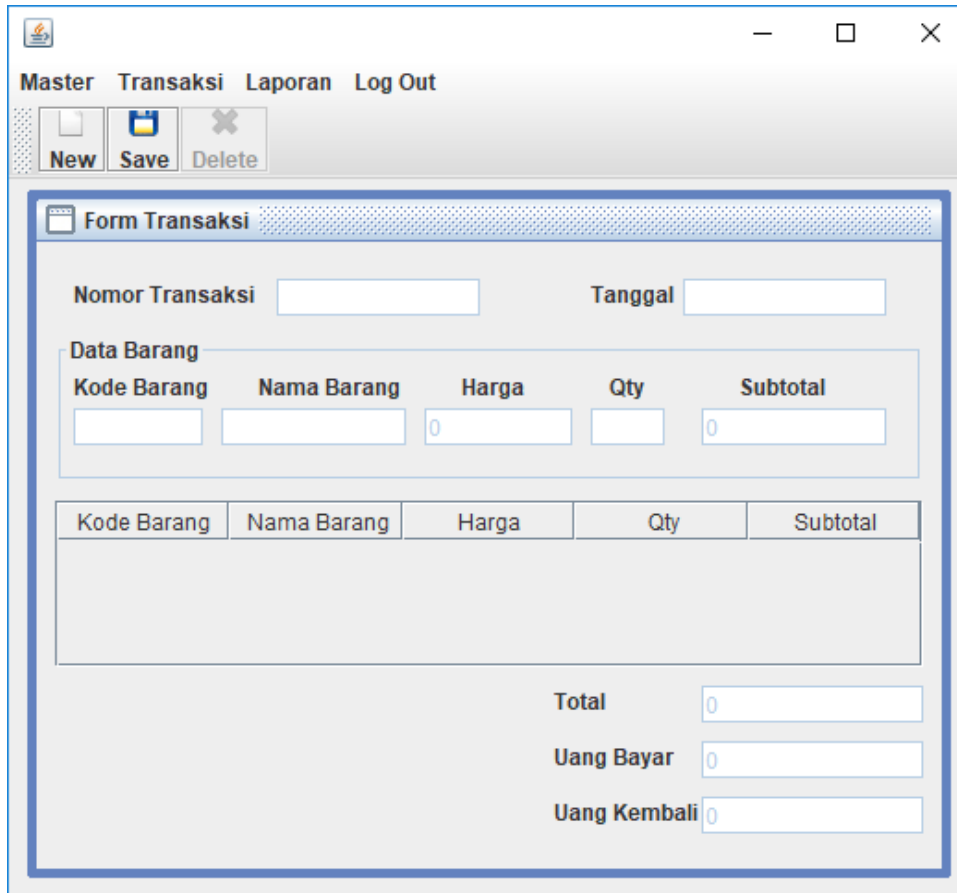
# Contoh Penerapan MVC

- Contoh Model dari tabel Transaksi

```
53 public Transaksi(String noTrans, Date tanggal, double total) {  
54     this.noTrans = noTrans;  
55     this.tanggal = tanggal;  
56     this.total = total;  
57 }  
58  
59 public String getNoTrans() {  
60     return noTrans;  
61 }  
62  
63 public void setNoTrans(String noTrans) {  
64     this.noTrans = noTrans;  
65 }  
66  
67 public Date getTanggal() {  
68     return tanggal;  
69 }  
70  
71 public void setTanggal(Date tanggal) {  
72     this.tanggal = tanggal;  
73 }  
74  
75 public double getTotal() {  
76     return total;  
77 }
```

# Contoh Penerapan MVC

- View dari tabel Transaksi



Master Transaksi Laporan Log Out

New Save Delete

Form Transaksi

Nomor Transaksi  Tanggal

Data Barang

Kode Barang	Nama Barang	Harga	Qty	Subtotal
<input type="text"/>	<input type="text"/>	0 <input type="text"/>	<input type="text"/>	0 <input type="text"/>

Kode Barang Nama Barang Harga Qty Subtotal

Total

Uang Bayar

Uang Kembali

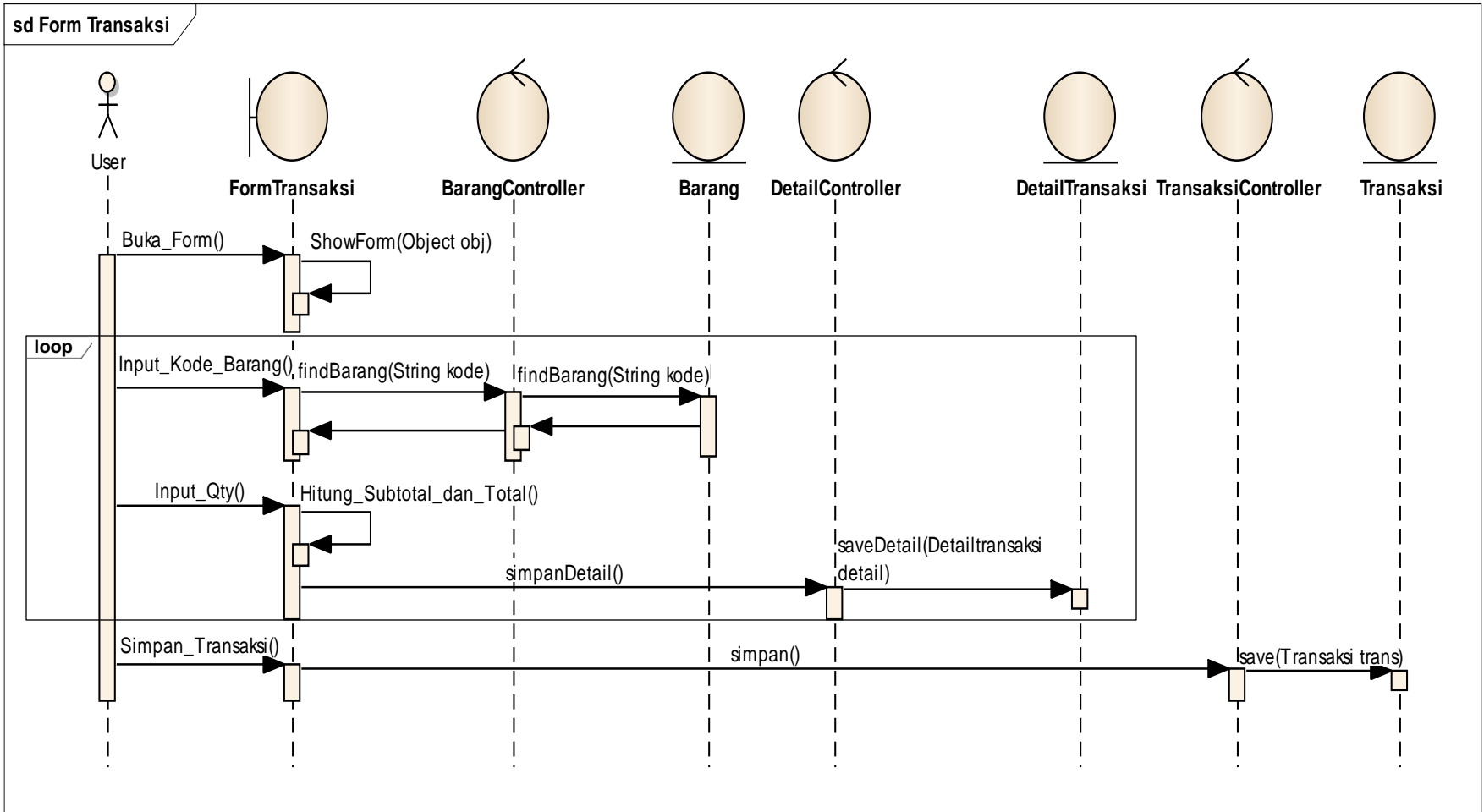


# Contoh Penerapan MVC

- Controller dari tabel Transaksi

```
36 public void save(Transaksi trans) throws Exception{
37     EntityManager em=getEntityManager();
38     try{
39         em.getTransaction().begin();
40         em.persist(trans);
41         em.getTransaction().commit();
42     }catch(Exception ex){}
43 }
44
45 public Transaksi findTransaksi(String kode){
46     EntityManager em=getEntityManager();
47     try{
48         return em.find(Transaksi.class, kode);
49     }finally{}
50 }
51
52 public String nomor(){
53     String kode="TR001";
54     EntityManager em=null;
55     try{
56         em = getEntityManager();
57         Query q=em.createQuery("select count(t.noTrans) from Transaksi t");
58         q.setMaxResults(1);
59         Long hasil=(Long) q.getSingleResult();
```

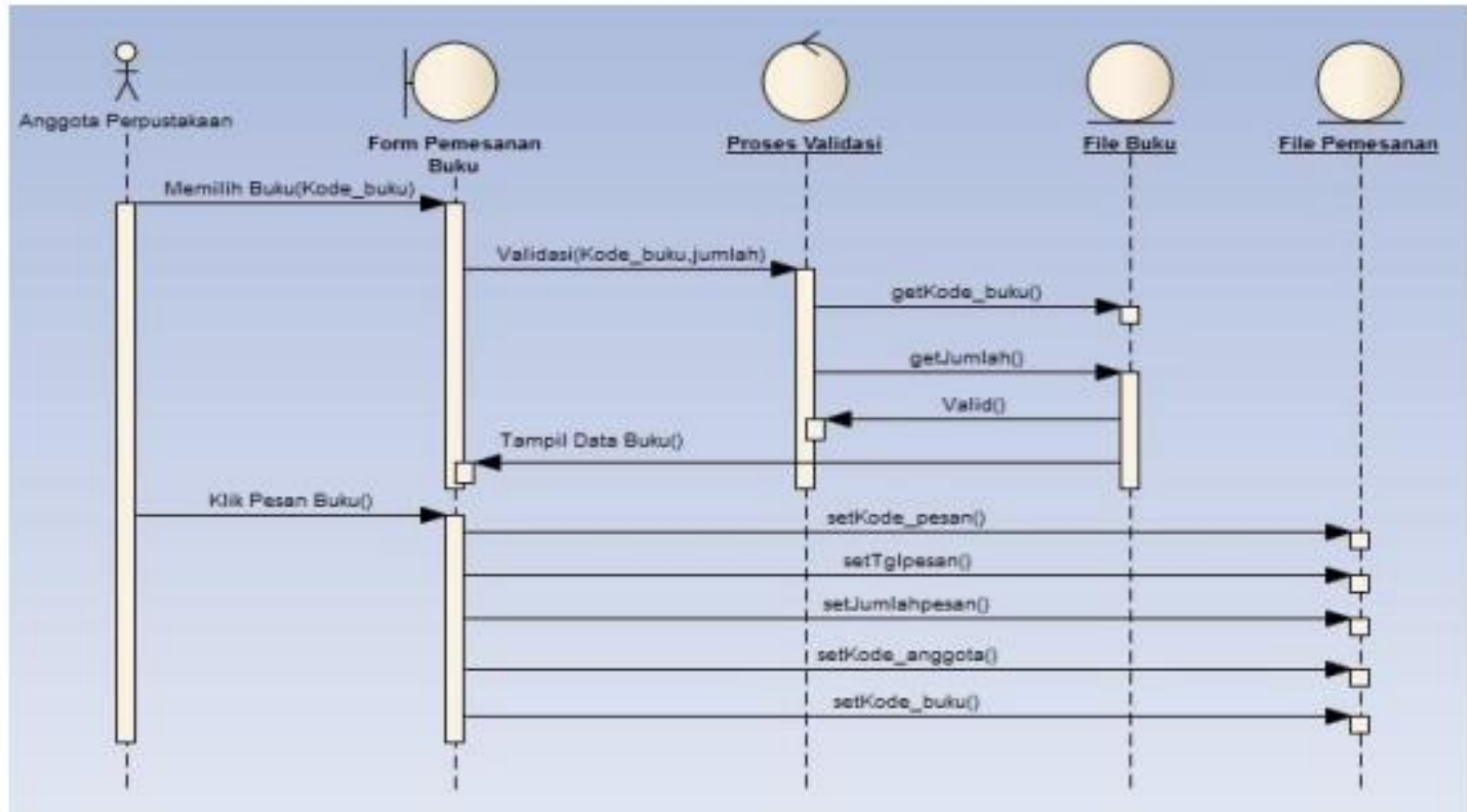
# Contoh Sequence Diagram



# Penjelasan Diagram

- User berinteraksi langsung dengan tampilan layar, pada contoh di atas, User berinteraksi dengan form transaksi. Hal yang pertama dilakukan adalah membuka form transaksi, lalu input kode barang, proses pencarian kode barang melalui controller barang menuju ke entitas Barang, kemudian nama dan harga barang ditampilkan pada form.
- Selanjutnya User menginput jumlah beli (qty) dan dilakukan perhitungan subtotal dan total. Setelah itu data tersebut masuk disimpan dalam entitas DetailTransaksi melalui controller detail.
- Setelah semua data diinput, maka semua data transaksi disimpan dalam entitas Transaksi melalui controller transaksi.

# Contoh Sequence Diagram

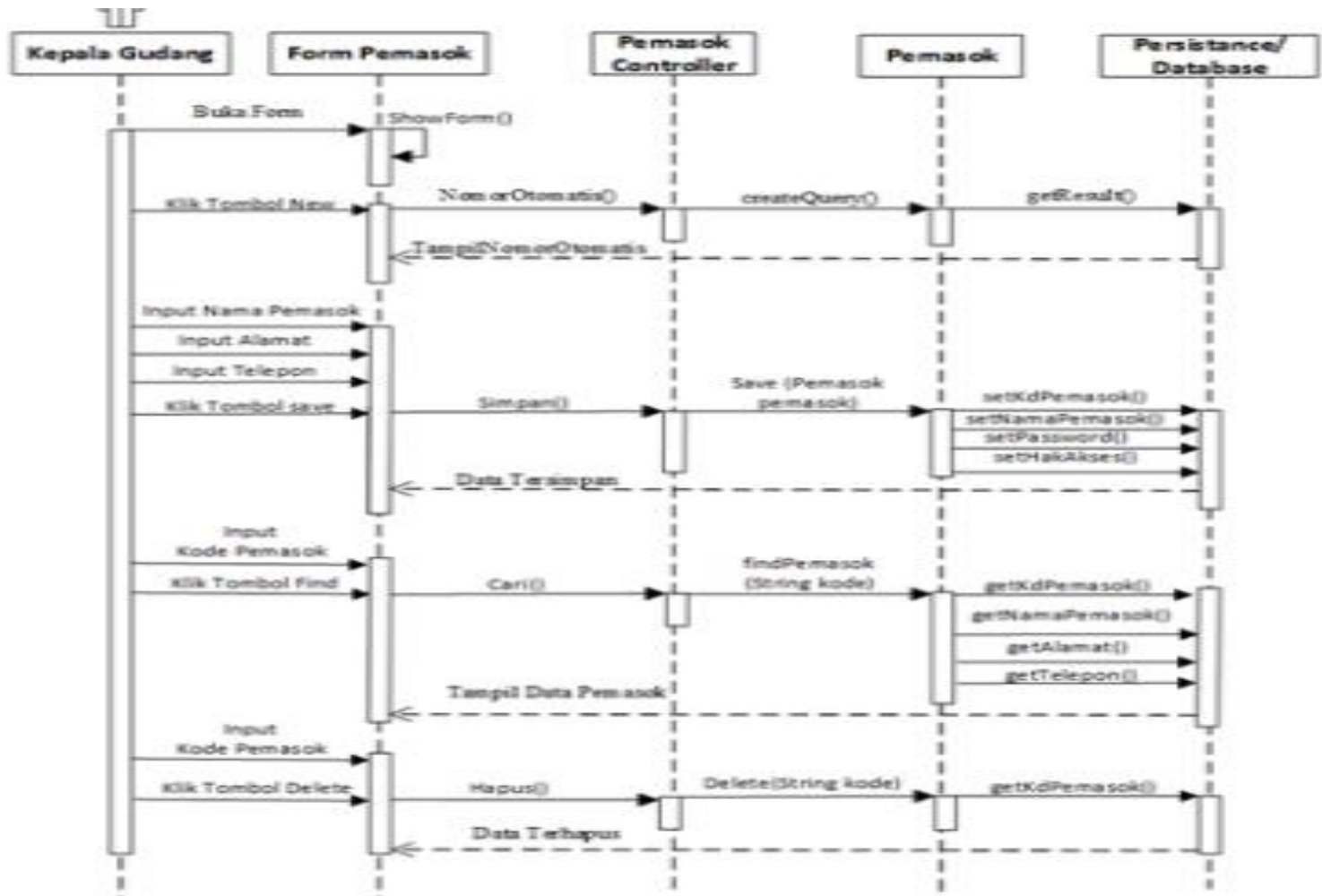


# Penjelasan Diagram

- Sequence diatas merupakan sequence diagram dari proses pemesana buku pada perpustakaan. User berinteraksi melalui form pemesanan dengan memilih buku, kemudian dilakukan validasi ketersediaan buku oleh controler, dilakukan pengecekan ke tabel buku. Tabel buku mengirim data buku ke form pemesanan. User melakukan pemesanan buku data buku dikirim ke tabel pemesanan

Studi kasus diambil dari artikel ilmiah Yoyok Maryono & Ida Darwati dengan judul Perancangan Web Perpustakaan Pada SMP Taruna Bhakti

# Contoh Sequence Diagram



# Penjelasan Diagram

- Sequence diatas adalah sequence dari use case mengelola data pemasok dimana kegiatan yang bisa dilakukan adalah menyimpan, mencari, menghapus data pemasok. Nomor otomatis ditampilkan oleh sistem, data yang diinput adalah nama pemasok, alamat, No tlp. Tabel yang terkoneksi adalah tabel pemasok.

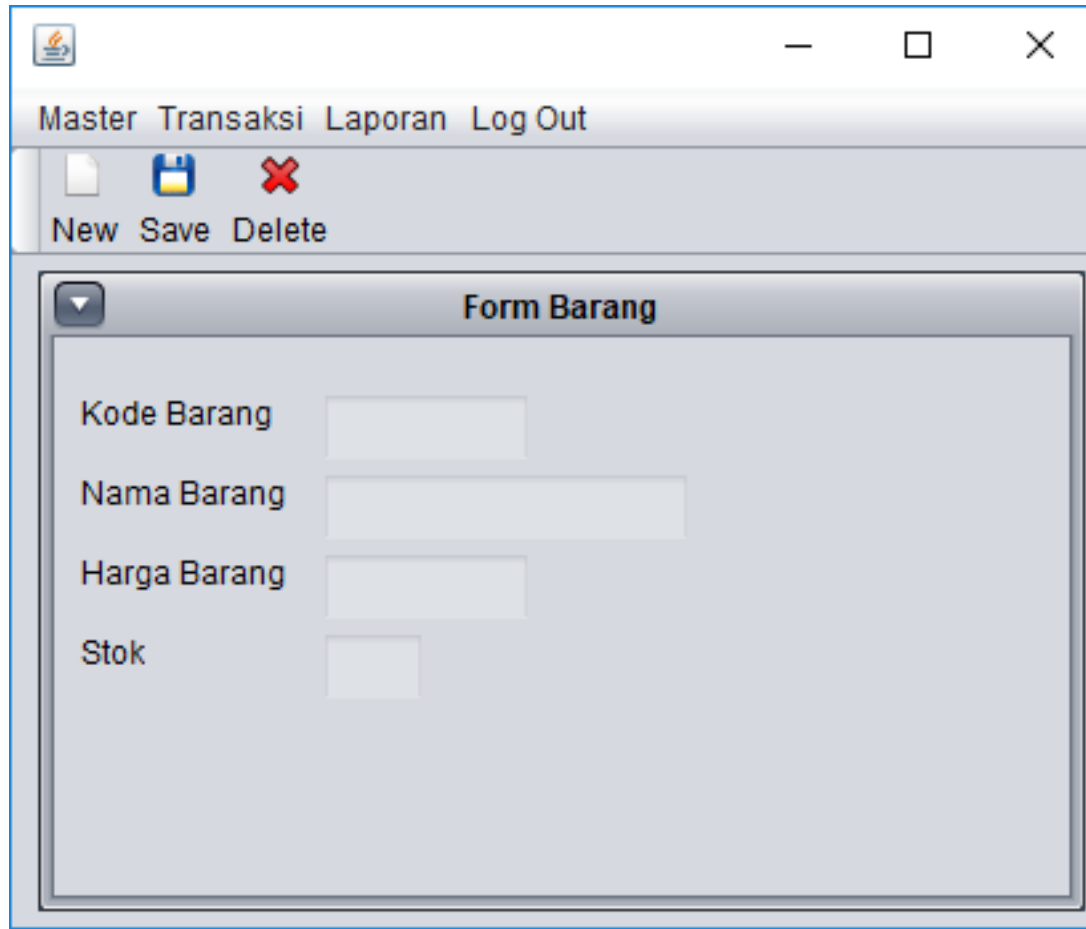
Studi kasus diambil dari artikel ilmiah Bibit Sudarsono & Erniyati dengan judul Perancangan Program Sistem Informasi Persediaan dan Penjualan Barang pada Toko Sparepart Motor

# Tugas

- Buatlah kelompok terdiri dari maksimal 5 orang.
- Tugas dikerjakan di kelas.
- Waktu 1 jam.
- Buatlah sequence diagram sesuai dengan petunjuk di slide berikut ini.



# Tugas



The screenshot shows a web application window with a title bar containing a small icon and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: Master, Transaksi, Laporan, and Log Out. Underneath the menu bar is a toolbar with three icons: a document (New), a floppy disk (Save), and a red X (Delete), with the labels 'New', 'Save', and 'Delete' respectively. The main content area is titled 'Form Barang' and contains four input fields arranged vertically, each with a label to its left: 'Kode Barang', 'Nama Barang', 'Harga Barang', and 'Stok'.

# Tugas

- User membuka form barang, klik tombol New, lalu muncul kode barang otomatis melalui BarangController.
- User menginput nama barang, harga, dan stok, lalu klik tombol Save untuk menyimpan data ke dalam entitas/class Barang melalui BarangController.