Volume IV 2009

SRITI

Proceeding

Seminar Nasional Riset Teknologi Informasi 2009

"Ubiquitos Computing"

Yogyakarta, 08 Agustus 2009

Komputasi

Kecerdasan Buatan

Teknologi Basis Data

(Data Meaning, Data Warehouse)

Pemodelan dan Aplikasi Sistem Informasi

Komunikasi Data dan Jaringan Komputer

Signal Processing

Sistem Kendali Robotika

Pengolahan Citra

Multimedia dan Grafika

Games

Diselenggarakan Oleh :

STMIK AKAKOM YOGYAKARTA

STEERING COMMITTEE

Prof. H. Adhi Susanto, M.Sc., Ph.D (UGM)
Prof. Drs. Suryo Guritno, M.Stat., Ph.D (UGM)
Prof. Dr. Ir. Achmad Djunaedi, MUP (UGM)
Prof. Dr. Ir. Prayoto., M.Sc (STMIK AKAKOM)
Prof. Drs. Setiadji, S.U. (STMIK AKAKOM)
Dr. Ir. Inggriani Liem (ITB)
Dr. Ir. Titon Dutomo, M.Eng (PENS-ITS)
Dr. Ir. Sasongko Pramono Hadi, DEA (Dir. ST Multimedia MMTC)
Ir. Lukito Edi Nugroho, M.Sc., Ph.D (UGM)
Drs. Retantyo Wardoyo, M.Sc., Ph.D (UGM)

PELAKSANA

Pelindung
Ketua STMIK AKAKOM

Penanggung Jawab
Kepala Puslitbang dan PPM

Tim Pengarah
Drs. Berta Bednar, M.T.
Ir. M. Guntara, M.T.
Heru Agus Triyanto, S.E., M.M.
Drs. Tri Prabawa, M.Kom.

Ketua 1
Ir. Totok Suprawoto, M.M., M.T.

Ketua 2
Dra. Syamsu Windarti, M.T. , Apt.

Bendahara
Pulut Suryati, S.Kom.
Dra. Torsinawati

Kesekretariatan/Komunikasi

Deborah Kurniawati, S.Kom.
Cosmas Haryawan, S.Tp., S.Kom.
Sri Redjeki, S.Si., M.Kom.
Sigit Anggoro, S.T., M.T.
H. Sri Widodo
F. Prihantini
Nailus Sa'adah
Rita Darundia
Theo A. Richie Y.

Materi/Acara
L.N. Harnaningrum, S.Si., M.T.
Enny Itje Sela, S.Si., M.Kom.
Agung Budi Prasetyo, S.Kom., M.Kom.
Aries Damayanti, S.Kom.
Al. Agus Subagyo, S.E., M.Si.
Thomas Edison Tarigan, S.Kom.

Perlengkapan\Dokumentasi\Dekorasi\Akomodasi\Konsumsi

Indra Yatini Buryadi, S.Kom., M.Kom.
Ir. Mashudi
F.X. Henry Nugroho, S.T.
Ary Adjidharma A.W., S.Kom., MMSI
Dra. M. Titik Maryanti
Dwi Suwarsono
Teki Astuti
Kuindra Iriyanto

Kata Pengantar

Assalamu'alaikum Wr. Wb.

Puji syukur marilah kita panjatkan ke hadirat Allah SWT yang telah melimpahkan rahmat dan hidayahnya, sehingga dapat terselesaikannya penyusunan *Proceeding* SRITI 2009. Buku ini memuat naskah hasil penelitian dari berbagai bidang kajian yang akan dipresentasikan pada Seminar Riset Teknologi Informasi (SRITI) 2009 ke-4 yang telah menjadi agenda tahunan dari Pusat Penelitian dan Pengembangan STMIK AKAKOM Yogyakarta dan sekaligus sebagai rangkaian dari peringatan 30 tahun STMIK AKAKOM.

Call for paper pada SRITI 2009 berbeda dari 3 periode penyelenggaraan sebelumnya, pada seminar ini naskah yang dikirimkan kepada Panitia sudah dalam bentuk *full paper*, sehingga naskah yang masuk ke panitia merupakan naskah final hasil penelitian yang siap dipublikasikan. Naskah yang masuk ke panitia selanjutnya di-*review* oleh para pakar di bidangnya yang berasal dari ITB, UGM, PENS-ITS, MMTC, dan STMIK AKAKOM. Atas kesediaan, kerjasama dan konsistensinya dalam me-*review* seluruh naskah yang dikirimkan, panitia mengucapkan banyak terima kasih.

Kegiatan SRITI 2009 mengambil tema tentang "*Ubiquitous Computing*" direncanakan dapat menyidangkan secara parallel sesuai dengan kelompok kajian ilmu dalam waktu satu hari. Panitia menyadari bahwa, hingga saat ini masih banyak *paper contents* yang belum mengacu pada tema, namun mengingat lingkup bidang kajian teknologi informasi yang sangat luas, maka kedepan diharapkan masih dapat ditingkatkan kesesuaian, kedalaman, maupun spektrum kajiannya.

Meskipun kegiatan seminar dan pendokumentasian naskah dalam *proceeding* ini telah dipersiapkan dengan baik, namun kami menyadari masih terdapat banyak kekurangnnya. Untuk itu, panitia mohon maaf yang sebesar-besarnya dan terima kasih atas kepercayaan serta kerjasamanya dalam kegiatan ini. Kritik dan saran perbaikan sangat diharapkan untuk penyempurnaan dimasa mendatang, yang dapat dikirimkan melalui e-mail sriti@akakom.ac.id.

Kepada semua pihak yang terlibat, baik langsung maupun tidak langsung dalam penyusunan proceeding SRITI 2009, panitia mengucapkan terima kasih.

Wassalamu'alaikum Wr. Wb.

Yogyakarta, 08-08-2009

Panitia SRITI 2009
Ketua Pelaksana,

Ir. Totok Suprawoto, M.M., M.T.

Penerapan Arsitektur *Model-View-Controller* Menggunakan Java Pada E-Commerce

Adi Kusjani

STMIK AKAKOM YOGYAKARTA

ABSTRAK

Dalam penelitian ini membahas tentang program aplikasi *e-commerce* penjualan Batik, dengan arsitektur MVC Model 1 (*page-centric*), menggunakan kode JSP, JavaBean dan database server MySQL dan editornya java Netbeans 5.5. Aplikasi *e-commerce* ini dibuat dengan teknologi MVC (*Model-View-Controller*) yang membagi program aplikasi menjadi 3 bagian. *Model* bertanggungjawab tugas dalam hal melakukan proses pengelolaan data dan menampung data. *View* mempunyai tugas untuk mengelola seluruh detail tampilan *user interface*. *Controller* berfungsi untuk mengelola permintaan dari klien untuk kemudian mengirimkan permintaan tersebut pada fungsi atau *method* yang sesuai untuk pemrosesan selanjutnya. Dalam aplikasi ini komponen *Model* menggunakan JavaBean sedangkan komponen *View* dan *Controller* menggunakan kode JSP, hal ini akan memberikan secara jelas batas antara orang yang bertanggung jawab terhadap kode JSP (*View* dan *Controller*) dan orang yang bertanggung jawab terhadap kode JavaBean (*Model*).

Kata Kunci : e-commerce, JavaBean, JSP, MVC, MySQL, Netbeans, page-centric, user interface

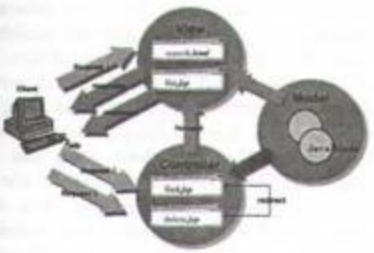
PENDAHULUAN

MVC (*Model-View-Controller*) diciptakan sekitar tahun 1970 oleh tim SmallTalk. MVC adalah sebuah mekanisme yang diciptakan untuk memisahkan objek sesuai dengan fungsinya. MVC memisahkan kode menjadi tiga yaitu :

1. *Model*, merupakan implementasi dari logika bisnis dan data bisnis.
2. *View*, mengandung keseluruhan detail dari implementasi *user interface*.
3. Controller (business logic), pengontrol aliran request juga data.

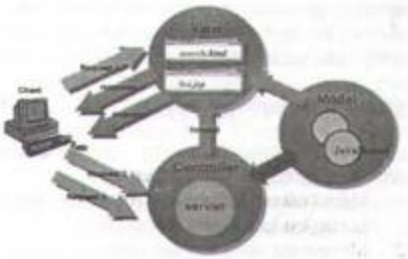
Dalam desain web aplikasi ada dua macam arsitektur MVC sebagai dasar arsitektur.

1. Arsitektur Model 1 (*Page-Centric*), terdiri : *Model* (JavaBean atau EJB), *View* dan *Controller* (JSP,JSF)



Gambar 1. Diagram MVC Model 1

2. Arsitektur Model 2 (*Servlet-Centric*), terdiri : *Model* (JavaBean atau EJB), *View* (JSP,JSF) dan *Controller* (Servlet).



Gambar 2. Diagram MVC Model 2

Perumusan Masalah

Berkembangnya aplikasi Web pada logika bisnis yang semakin komplek diperlukan suatu arsitektur untuk mengatasi masalah tersebut. Salah satunya adalah arsitektur *Model-View-Controller*, sehingga dengan pembuatan aplikasi E-Commerce berbasiskan arsitektur *Model-View-Controller* pada penelitian ini diharapkan akan dapat mengatasi masalah logika bisnis yang ada pada aplikasi E-Commerce penjualan Batik.

Tujuan Penelitian

Tujuan dari penelitian ini adalah dapat menerapkan, menganalisa dan membuktikan fungsi dari masing-masing komponen dalam arsitektur *Model-View-Controller* menggunakan Java pada suatu aplikasi E-Commerce penjualan Batik.

Konstribusi Penelitian

Menjadikan peneliti memahami tentang keuntungan penggunaan aplikasi web dengan arsitektur *Model-View-Controller* dan juga dapat dijadikan sebagai *prototype* untuk membangun aplikasi web yang sesuai dengan kebutuhan.

TINJAUAN PUSTAKA

Hasil penelitian "Aplikasi E-Ticketing Berbasiskan Model-View-Controller (MVC) Pattern" oleh Faisal Wiryasantika dari Central Library Institute Technology Bandung, tahun 2004, menunjukkan bahwa MVC *pattern* menyederhanakan pemeliharaan dengan tetap menjaga seluruh *logic* agar tidak menjadi rumit (*intertwined*) dan secara alami MVC memberikan batas antara orang yang bertanggung jawab terhadap kode Java dan orang yang bertanggung jawab terhadap presentation layer. Dari penelitiaan diatas muncul ide dari saya untuk menerapkan arsitektur *MVC* pada aplikasi E-Commerce penjualan Batik.

METODE PENELITIAN

Langkah-langkah yang dilakukan pada penilitian ini adalah sebagai berikut :

1. Menentukan kebutuhan perangkat keras dan perangkat lunak yang dibutuhkan.
2. Merancang aplikasi, aplikasi *ecommerce* yang akan dibuat adalah aplikasi *ecommerce* yang menjual pakaian dan kain batik.
3. Membuat program, program dibuat dengan Java Server Page (JSP), Java Bean dan database server MYSQL.
4. Pengujian program, pegujian dilakukan dengan menjalankan program lewat Netbeans yang kemudian hasilnya akan ditampilkan di web browser Firefox.
5. Pengambilan kesimpulan, dilakukan dengan mengevaluasi hasil pengujian program.

HASIL PEMBAHASAN

Penjelasan Aplikasi

Aplikasi *e-commerce* penjualan Batik ini terdiri 5 proses utama, yaitu : Pendaftaran pelanggan, Proses *login* pelanggan, Menampilkan barang berdasarkan kategori, Pencarian barang dan Transaksi pembelian.

Komponen Akses Database

Dalam komponen akses database pada aplikasi ini terdapat 2 fungsi, untuk membuka dan menutup koneksi, yaitu logIn() dan logOff(). Menggunakan *interface* Connection dan Driver Manager, dimana koneksi akan mengarah ke database server MySQL yang terdapat pada *localhost*, dengan *username* = "Adi", *password* = "123" dan nama databasenya batik.db.

```
package Batik.jsf;
import java.sql.*;

public class Koneksi {
    private Connection konek;
    private String driver =
  "com.mysql.jdbc.Driver";
    private String url =
  "jdbc:mysql://localhost/batikdb";
    private String username = "Adi";
    private String password = "123";
    public Connection logIn() {
        try {
        Class.forName( driver
).newInstance();
        konek = DriverManager.getConnection(
        url, username, password ); }
              catch(Exception ex)
        {ex.printStackTrace();          }
         return konek;
    }
    public void logOff()  {
        try {
        konek.close();}
        catch(Exception ex)    {
        ex.printStackTrace(); }
    }
}
```

Halaman Utama Aplikasi

Halaman utama diawali dengan pemanggilan file index.jsp, dimana pada file ini digunakan tag direktif, <%@ page language="java" import="java.util.*, java.sql.*, Batik.jsf.*" %>, yang berarti mengimpor semua kelas yang terdapat pada paket java.util, java.sql, Batik.jsf dan bahasa yang digunakan untuk menyusun kode JSP adalah java, sehingga file index.jsp dapat menggunakan kelas-kelas yang ada dipaket-paket tersebut. Halaman ini berfungsi untuk menampilkan halaman yang berisi batik-batik yang dipromosikan. Dalam melakukan proses menampilkan data batik yang ada dalam database file index.jsp memerlukan 2 file yaitu POBatik.java dan KatalogBean.java. Tag direktif ditujukan untuk memberitahukan pada mesin JSP dan bukan ditujukan untuk membentuk keluaran.

Tag direktif berikutnya adalah <%@ include file="cpromosi.jsp" %> dan <%@ include file="header.html" %>, tag ini berfungsi untuk

menyisipkan suatu file dalam hal ini adalah file cpromosi.jsp dan header.html. Dimana file cpromosi.jsp digunakan sebagai *Controller* untuk memanggil fungsi getPromosiBatik() yang ada pada file KatalogBean.java, dengan terlebih dahulu membuat aksi <jsp:useBean id="oKatalog" scope="page" class="Batik.jsf.KatalogBean" />, yang mendefinisikan suatu objek dari file *java bean* yaitu KatalogBean.java, dengan nama objeknya oKatalog dan *scope*-nya *page* (objek hanya dikaitkan dengan halaman sekarang). Karena adanya objek oKatalog ini, maka fungsi getPromosiBatik() dapat dipanggil pada file cpromosi.jsp. Fungsi dari getPromosiBatik() sebagai *Model* untuk mendapatkan data pada tabel Batik yang mempunyai *field id_produk*-nya adalah hasil menyeleksian *field id_produk* pada tabel Promo yang mempunya *id_promo* = 1. Pada fungsi getPromosiBatik() juga diciptakan objek oBatikBean dari file POBatik.java dengan perintah POBatik oBatikBean = new POBatik();, objek oBatikBean akan digunakan untuk memanggil fungsi-fungsi yang ada di file POBatik.java (setIDProduk , setNama, setAsal, setDeskripsi, setHarga, setGambar) dengan nilai argumen yang berasal dari hasil perintah eksekusi *query*, dimana pemanggilan fungsi ini akan menghasilkan nilai balik berupa data yang berasal dari hasil penyeleksian pada tabel Batik, untuk kemudian hasil data ini akan dibawa ke file index.jsp untuk kemudian ditampilkan, yang mana file index.jsp difungsikan sebagai *View*.

```
public Vector getPromosiBatik() {
        String tSQL;
        Vector vKatalog = new Vector()
        try {
         tSQL = "SELECT * FROM Batik WHERE
         id_produk IN (SELECT id_produk FROM
         promo WHERE id_promo=1)";
        konek();
        prth = konek.createStatement();
        hasil = prth.executeQuery(tSQL);
        while(hasil.next()) {
                POBatik oBatikBean = new
                POBatik();
                oBatikBean.setIDProduk
                (hasil.getString
                ("id_produk"));
                oBatikBean.setNama
                (hasil.getString ("nama"));
                oBatikBean.setAsal
                (hasil.getString ("asal"));
                oBatikBean.setDeskripsi
                (hasil.getString
                ("deskripsi"));
                oBatikBean.setHarga
                (hasil.getInt ("harga"));
                oBatikBean.setGambar
                (hasil.getString ("gambar"));
                vKatalog.addElement(oBatikBea
                n); }   }
        catch(Exception ex) {
        ex.printStackTrace(); }
        return vKatalog; }
```

Untuk tag direktif <%@ include file="header.html" %> akan memanggil file header.html, yang berfungsi sebagai *header* disetiap halaman web, file header.html difungsikan sebagai *View* dan berisi *link-link* untuk menuju ke halaman web lain.

Gambar 3. Halaman Utama

Berikutnya adalah tag direktif <jsp:include page="navigasi.jsp" flush="true" />, yang berfungsi untuk menyisipkan file navigasi.jsp, dimana file ini difungsikan sebagai *View* untuk menampilkan halaman Login dan halaman Navigasi yang berupa *link* untuk mengarahkan ke file katalog.jsp dengan membawa parameter aksi dan IDkategori.

Pada file navigasi.jsp juga terdapat tag direktif <%@ include file="cnavigasi.jsp" %>, yang berfungsi untuk menyisipkan file cnavigasi.jsp, dimana file cnavigasi.jsp difungsikan sebagai *Controller*. Pada file cnavigasi.jsp ini terdapat deklarasi objek oStatus dimana objek ini akan diisi data sesi, hasil dari pemanggilan fungsi getAttribute oleh variabel *session*, dengan perintah Object oStatus = session.getAttribute("Status"), kemudian isi dari objek oStatus akan diseleksi dengan perintah penyeleksian dimana jika objek oStatus kosong maka akan mendaftarkan data *session* Status dengan isi "Tidak Login" dan juga variabel tUserName akan diisi teks kosong.

```
if (oStatus == null ) {

    session.setAttribute("Status",      "Tidak
    Login");    }
      String tUserName = "";
```

Jika objek oStatus tidak kosong maka akan mengubah objek oStatus menjadi bertipe *String* dan kemudian menyimpannya pada variabel sStatus. Kemudian ada penyeleksian lagi dimana jika isi variabel sStatus sama dengan "Login" maka akan menjadikan isi variabel tUserName adalah hasil dari pemanggilan fungsi getAttribute oleh variabel *session* dengan parameter data sesi "User" yang

dirubah ke tipe *String*. Dan juga mengisi variabel sLogout dengan "Tpl".

```
if (oStatus != null) {
      String sStatus = oStatus.toString();
     if (sStatus.equals("Login")) {
      tUserName                              =
   session.getAttribute("User").toString();
      sLogout = "Tpl";  }
    }
```

Pada file navigasi.jsp, terdapat metode "Post" yang mengarah ke file login.jsp dengan membawa isi variabel username dan password yang telah diisikan pada waktu login. File login.jsp difungsikan sebagai *Controller*. Pada file login.jsp ini terdapat deklarasi objek oInfoPelanggan. Objek oInfoPelanggan akan digunakan untuk memanggil fungsi cekLogin yang ada di file InfoPelangganBean.java.

```
String tStatusLoginPelanggan =
oInfoPelanggan.cekLogin(tUserName, tUser
Password);
```

File InfoPelangganBean.java difungsikan sebagai *Model*, untuk mendapatkan data pada tabel Pelanggan yang mempunyai *username* sama dengan isi variabel *username* yang berasal dari file navigasi.jsp, dimana jika hasil perintah *query*-nya tidak ada maka akan mengisi variabel tLoginStatus dengan "Username Tidak Valid", jika ada akan mengecek kecocokan passwordnya jika sama akan mengisi variabel tLoginStatus dengan "Login Sukses" dan jika tidak cocok akan mengisi variabel tLoginStatus dengan "Password Salah". Pemanggilan fungsi cekLogin akan menghasilkan nilai balik berupa status *login* pelanggan yang kemudian akan disimpan pada variabel tStatusLoginPelanggan, yang ada di file login.jsp.

```
public String cekLogin(String _tUserName,
String _tPassword) {
        String tLoginStatus;
        String tUserPassword;
        try {
        String sql = "SELECT user_name,
        password " + "FROM pelanggan
         WHERE user_name = '" + _tUserName
         + "'";
         System.out.println(sql);
          konek = konek();
          prth = konek.createStatement();
          hasil = prth.executeQuery(sql);
          if(hasil.next()) {
          tUserPassword =
          hasil.getString("password");
         if(tUserPassword.equals(_tPassword)
         ) {
              tLoginStatus = "Login Sukses";
}
              else
              tLoginStatus = "Password
Salah";  }
  }
   else
{   tLoginStatus = "Username Tidak
Valid";  }
}
 catch(Exception ex) {
       ex.printStackTrace();
 tLoginStatus = "Login Gagal";  }
System.out.println("STATUS LOGIN :
"     + tLoginStatus);
  return tLoginStatus;
}
```

Kemudian isi variabel tStatusLogin-Pelanggan akan diseleksi dimana jika isinya sama dengan "Username Tidak Valid" maka data sesi Status akan berisi "invalidusr", jika isinya sama dengan "Password Salah" maka data sesi Status akan berisi "invalidpass", jika isinya sama dengan "Login Sukses" maka data sesi Status akan berisi "Login" dan data sesi "User" akan berisi "tUserName", sedangkan *default*-nya akan mengisi data sesi Status akan berisi "Tidak Login". Tag direktif berikutnya adalah <%@ include file="cdata.jsp" %>, tag ini berfungsi untuk menyisipkan suatu file dalam hal ini adalah file cdata.jsp. Dimana file cdata.jsp digunakan sebagai *Controller* untuk memanggil fungsi-fungsi yang ada pada file POBatik.java, dengan terlebih dahulu membuat aksi, yaitu :

```
<jsp:useBean id="oBatik" scope="page"
class="Batik.jsf.POBatik" />
```

Aksi diatas untuk mendefinisikan suatu objek dari file *java bean* yaitu POBatik.java, dengan nama objeknya oBatik. Karena adanya objek oBatik ini, maka fungsi-fungsi yang ada pada file POBatik.java, dapat dipanggil pada file cdata.jsp. File POBatik.java digunakan sebagai *Model* untuk mendapatkan data IDProduk ,Nama, Asal, Deskripsi, Harga, dan Gambar dari tabel Batik.

Gambar 4.Login Sukses

Pada file index.jsp juga terdapat *form* pencarian data Batik, dengan metode *post* dan

aksinya akan mengarah ke file katalog.jsp dengan membawa parameter aksi = "cari".

```
<form name="form1" method="post"
action="katalog.jsp?aksi=cari">
<font color="red">Masukkan kata kunci :
<input type="text" name="katakunci"><br><br>
<b><font color="red"><input type="submit"
name="Submit2" value="Cari"> </form>
```

Ganbar 5. Form Pencarian Batik

Pada file katalog.jsp terdapat pemanggilan file ckatalog.jsp, dimana pada file ini didefinisikan objek dari file KatalogBean.java dengan nama oKatalog. Karena adanya objek oKatalog ini, maka fungsi-fungsi yang ada pada file KatalogBean.java dapat digunakan oleh objek oKatalog. Kedudukan file ckatalog.jsp sebagai *Controller*, didasarkan pada isi parameter aksi dimana jika isinya sama dengan "cari" maka objek oKatalog akan menjalankan fungsi getCariKatalogBatik yang ada pada file KatalogBean.java.

```
if (tAksi.equals("cari"))   {
  Stringt Katakunci=request.getParameter
  ("katakunci");
  System.out.println(tKatakunci);
  vKatalog = (Vector)
  oKatalog.getCariKatalogBatik(tKatakunci);
    tJudul = "Hasil Pencarian"; }
```

Gambar 6. Hasil Pencarian dengan Katakunci "Energic"

Fungsi getCariKatalogBatik pada file KatalogBean.java akan melakukan proses pencarian data *field* nama pada tabel Batik. Pencarian didasarkan pada kata kunci yang dimasukkan pada *form* pencarian yang bernama "katakunci", kemudian akan dialihkan ke parameter _tNama, disini file KatalogBean.java berkedudukan sebagai *Model*. Pada fungsi getCariKatalogBatik diciptakan objek oBatikBean dari file POBatik.java dengan perintah POBatik oBatikBean = new POBatik();, objek ini digunakan untuk memanggil fungsi-fungsi di file POBatik.java, dimana pemanggilan fungsi ini akan menghasilkan nilai balik berupa data yang berasal dari hasil proses pencarian data *field* nama di tabel Batik, kemudian hasilnya akan dibawa ke file katalog.jsp untuk kemudian ditampilkan.

```
public Vector
getCariKatalogBatik(String_tNama) {
        String tSQL;
        Vector vKatalog = new Vector();
        try {
        tSQL = "SELECT * FROM Batik WHERE
        nama LIKE '%" + _tNama + "%'";
            konek();
            prth = konek.createStatement();
            hasil = prth.executeQuery(tSQL);
            while(hasil.next()) {
                POBatik oBatikBean = new
                POBatik();
                oBatikBean.setIDProduk
                (hasil.getString
                ("id_produk"));
                oBatikBean.setNama
                (hasil.getString ("nama"));
                oBatikBean.setAsal
                (hasil.getString ("asal"));
                oBatikBean.setDeskripsi
                (hasil.getString
                ("deskripsi"));
                oBatikBean.setHarga
                (hasil.getInt ("harga"));
                oBatikBean.setGambar
                (hasil.getString ("gambar"));
                vKatalog.addElement(oBatikBea
                n); }   }
catch(Exception ex) {
ex.printStackTrace(); }
return vKatalog;
}
```

Proses Pesan Barang

Untuk pemesanan batik diawali dengan mengklik *link* Tambahkan ke Keranjang yang ada disetiap pilihan, dimana jika di-klik akan mengarah ke file keranjang.jsp.

```
<a href="keranjang.jsp?aksi=tambah&id=
<%=dIDProduk%>"><b>Tambahkan ke
Keranjang</a>
```

Perintah diatas berarti mengarah ke file keranjang.jsp dengan membawa parameter aksi yang berisi "tambah" dan parameter id yang berisi id produk sesuai dengan batik yang dipilih. Misalnya jika akan memilih batik "Clasic Creamy" maka akan

mengarah ke file keranjang.jsp dengan membawa nilai parameter aksi dan id produknya, dengan tampilan seperti pada gambar 7. Pada file keranjang.jsp terdapat pemanggilan file ckeranjang.jsp, dimana pada file ini didefinisikan objek dari file POBatik.java dengan nama oBatik.

Kedudukan file ckeranjang.jsp sebagai *Controller*, dimana pada file ini didefinisikan objek dari file KeranjangBean.java dengan nama oKeranjang. Jika pada isi parameter aksi yang berasal dari file keranjang.jsp, dimana jika isinya sama dengan "tambah" maka objek oKeranjang akan menjalankan fungsi masukkanKeKeranjang yang ada pada file KeranjangBean.java, jika isin-ya sama dengan "hapus" maka objek oKeranjang akan menjalankan fungsi kosongkan-Keranjang yang ada pada file Keranjang-Bean.java jika isinya sama dengan "Edit" maka objek oKatalog akan menjalankan fungsi updateKeranjang yang ada pada file Keranjang-Bean.java.

Gambar 7. Ketika *link* Tambahkan ke Keranjang di Klik

```
Vector vKrjBelanja = new Vector();
String tAksi;
String tPesan = "Keranjang belanja Anda
masih kosong...!";
int i=0;
int Total=0;
tAksi=request.getParameter("aksi");
if (tAksi != null) {
        if (tAksi.equals("tambah")) {
        String
        tIDProduk=request.getParameter("id")
        ;
         Integer iTmp =
Integer.valueOf(tIDProduk);
        int IDProduk = iTmp.intValue();
        if   (!oKeranjang.sudahAda(IDProduk))
{
        oKeranjang.masukkanKeKeranjang(IDPro
        duk); }
        vKrjBelanja =
oKeranjang.getKeranjang(); }
        if (tAksi.equals("hapus")) {
        String
        tIDKeranjang=request.getParameter("c
        artID");
         Integer iTmp = Integer.valueOf
        (tIDKeranjang);
        int iIDKeranjang = iTmp.intValue()-
1;
        oKeranjang.kosongkanKeranjang(iIDKer
an      jang);
        vKrjBelanja =
oKeranjang.getKeranjang();  }
         if (tAksi.equals("Edit")) {
        String
        tIDKeranjang=request.getParameter("
        artID");
        String
        tJumlah=request.getParameter("jumlah
        ");
        Integer iTmp =
        Integer.valueOf(tIDKeranjang);
        int iIDKeranjang = iTmp.intValue()-
1;
        Integer iTmp2 =
Integer.valueOf(tJumlah);
        int Jumlah = iTmp2.intValue();
        oKeranjang.updateKeranjang(iIDKeranj
        ang.Jumlah);
        vKrjBelanja =
oKeranjang.getKeranjang();  }
}
  else  {      vKrjBelanja =
oKeranjang.getKeranjang(); }
        int iUkuran = vKrjBelanja.size();
        String dUkuran =
Integer.toString(iUkuran);
```

File keranjang.jsp diposisikan sebagai *View*, dimana didalamnya terdapat pemanggilan file header.html dan navigasi.jsp yang diposisikan sebagai *View* dan cbayar.jsp yang difungsikan sebagai *Controller*. File cbayar.jsp untuk menetukan SubTotal dan Total harga Batik yang dibeli, dimana untuk menentukannya diperlukan Harga dari Batik tersebut, untuk itu diciptakan suatu objek dari file POBatik.java agar dapat menggunakan fungsi getHarga sehingga diperoleh harganya, disini objek yang diciptakan bernama oBtk.

Pada file keranjang.jsp juga menyediakan 3 aksi yaitu Hapus dan Edit yang mengarah pada dirinya sendiri dan Pesan yang mengarah ke file pesan.jsp. Pada file pesan.jsp terdapat perintah <%@ include file="cpesan.jsp" %>, untuk memanggil file cpesan.jsp, yang berkedudukan sebagai *Controller* dan pemanggilan file footer.html dengan perintah <%@ include file="footer.html" %> .

Pada file cpesan.jsp didefinisikan 2 objek dari file KeranjangBean.java dengan nama oKeranjang dan dari file InfoPelanggan-Bean.java dengan nama oPelang-gan.

Gambar 8. Memasukkan 2 Barang dengan Jumlah yang Telah di Edit ke Keranjang Belanja

Disini file KeranjangBean.java dan InfoPelangganBean.java berkedudukan sebagai *Model*. Objek oPelanggan akan menjalankan fungsi insertDataPesanan dan getIDPesananBaru yang ada di file InfoPelangganBean .java sedangkan Objek oKeranjang akan menjalankan fungsi kosongkanKeranjang yang ada di file KeranjangBean.java. Hasil dari pemanggilan file cpesan.jsp akan di tampilan pada file pesan.jsp seperti terlihat pada gambar 9.

```
<jsp:useBean id="oKeranjang" scope=
"session" class="Batik.jsf.KeranjangBean" />
<jsp:useBean id="oPelanggan" scope="page"
class="Batik.jsf.InfoPelangganBean" />
<%
        Vector vKrjBelanja = new Vector();
        String tUserName =
        session.getAttribute("User").toStrin
        g();
        vKrjBelanja =
        oKeranjang.getKeranjang();
        String tJudul="";
        String tPesan="";
        String tPesan2="";
        try {

if(oPelanggan.insertDataPesanan
                (vKrjBelanja, tUserName)) {
             int IDPesan =
             oPelanggan.getIDPesananBaru();
             tJudul = "Pesanan Anda telah
             diproses.";
             tPesan   =   "Terimakasih   "   +
             tUserName     +     ".     atas
             kepercayaannya.
              ID pemesanan Anda adalah :"
              + IDPesan;
              tPesan2 = "Barang yang Anda
              pesan akan segera kami
              kirim..!" ;

oKeranjang.kosongkanKeranjang
                 ();  }
                     else
                     {
               tJudul = "Ada kesalahan dalam
               pemesanan";
               tPesan = "Pemesanan
               bermasalah, silahkan
               diulang";
                     tPesan2 = "";  }
          }
          catch (Exception e)  {
          tJudul = "Ada kesalahan dalam
          pemesanan";
           tPesan = "Pemesanan bermasalah,
          silahkan diulang";
           tPesan2 = "";  }
%>
```

Fungsi InsertDataPesanan

```
public boolean insertDataPesanan(Vector
_vKeranjang, String  _tUserName) throws
SQLException {
           boolean bPesan = true;
           int total =
getTotalHarga(_vKeranjang);
          int id_pesan = getIDPesananBaru() +
          1;
          int id_pelanggan =
          getIDPelanggan(_tUserName);
           Iterator dat =
_vKeranjang.iterator();
          StringBuffer tBuffer = new
          StringBuffer();
          tBuffer.append("INSERT INTO Pesan
          VALUES (");
           tBuffer.append(id_pesan);
           tBuffer.append(",'");
           tBuffer.append(id_pelanggan);
           tBuffer.append("', '");
           tBuffer.append(total);
           tBuffer.append("')");
           String sql = tBuffer.toString();
           konek = konek();
           prth = konek.createStatement();
           bPesan = prth.execute(sql);
           diskonek();
            while (dat.hasNext()) {
                  Vector vIsi =
(Vector)dat.next();
            POBatik  oBatik = (POBatik)
          vIsi.elementAt(0);
          Integer jumlah = (Integer)
          vIsi.elementAt(1);
          Integer id_produk =
          Integer.valueOf(oBatik.getIDProduk(
          ));StringBuffer tBuffer1 = new
          StringBuffer();
          tBuffer1.append("INSERT INTO
          detail_pesan VALUES (");
                  tBuffer1.append(id_pesan);
                  tBuffer1.append(",'");

tBuffer1.append(id_produk.intValue());
                  tBuffer1.append("', '");

tBuffer1.append(jumlah.intValue());
                  tBuffer1.append("')");
                  String sql1 =
tBuffer1.toString();
                  konek = konek();
                  prth =
konek.createStatement();
                  bPesan = prth.execute(sql1);
                  diskonek();  }
         return bPesan = true;
         }
}
```

Gambar 9. Ucapan Terimakasih bahwa Proses Pemesanan Telah Selesai

Fungsi getIDPesananBaru

```
public int getIDPelanggan(String _tUserName)
 {
   int id_plg = 1;
   try  {
        String  sql  =  "SELECT  id_pelanggan
        FROM Pelanggan WHERE user_name     =
        '" + _tUserName + "'";
        konek = konek();
        rth = konek.createStatement();
        hasil = prth.executeQuery(sql);
        if(hasil.next())   {
        id_plg =   hasil.getInt("id_pelanggan
");
         }
        }
        catch(Exception ex) {
        ex.printStackTrace(); }
        finally {
        diskonek(); }
        return id_plg;
}
```

Proses Pendaftaran Pelanggan

Proses pendaftaran pelanggan diawali dengan mengklik *link* Pendaftaran, yang berarti mengarah ke file daftar.jsp, perintah ini ada pada file header.html (Pendaftaran).

Gambar 10. Tampilan *Form* Pendaftaran Pelanggan

File header.html dan daftar.jsp, berkedudukan sebagai *View*. Pada file daftar.jsp terdapat pemanggilan file cdaftar.jsp dengan perintah <%@ include file="cdaftar.jsp" %>. Kedudukan file cdaftar.jsp sebagai *Controller*, dimana pada file ini didefinisikan objek dari file InfoPelangganBean.java dengan nama oPelanggan.

```
<jsp:useBean id="oPelanggan" scope=
"page" class="Batik.jsf.InfoPelanggan
Bean" />
```

Jika pada isi parameter aksi yang berasal dari file daftar.jsp, dimana jika isinya sama dengan "catat" maka objek oPelanggan akan menjalankan fungsi insertDataPelanggan yang ada pada file InfoPelangganBean.java.

```
public boolean
insertDataPelanggan(String_tUserName, String
_tPassword, String_tNamaLengkap, String
_tEmail, String_tAlamat,
String _tNotel ) {
        boolean bDaftar = true;
        try {
                int iUserID =
                getIDPelangganTerbaru() + 1;
                StringBuffer tBuffer = new
                StringBuffer();
                tBuffer.append("INSERT INTO
                Pelanggan VALUES (");
                tBuffer.append(iUserID);
                tBuffer.append(",'");
                tBuffer.append(_tUserName);
                tBuffer.append("', '");
                tBuffer.append(_tNamaLengkap)
                ;
                tBuffer.append("', '");
                tBuffer.append(_tPassword);
                tBuffer.append("', '");
                tBuffer.append(_tEmail);
                tBuffer.append("', '");
                tBuffer.append(_tAlamat);
                tBuffer.append("', '");
                tBuffer.append(_tNotel);
                tBuffer.append("')");
                String sql =
tBuffer.toString();
                konek = konek();
                prth =
konek.createStatement();
                bDaftar = prth.execute(sql);
                diskonek();
                return bDaftar = true;
                  }
                catch(SQLException ex) {
                ex.printStackTrace();
                return bDaftar = false;  }
          }
```

Apabila fungsi insertDataPelanggan dapat menginputkan ke tabel Pelanggan, maka akan mengisi variabel tJudul dengan "Pendaftaran Sukses" dan jika tidak dapat menginputkan maka

akan mengisi variabel tJudul dengan "Pendaftaran Gagal".

Gambar 11. Tampilan jika Pendaftarannya Sukses

```
String tJudul="Pendaftaran Pelanggan";
String tAksi=request.getParameter("aksi");
if (tAksi == null) {
tAksi = "tampil";
}
     else if (tAksi.equals("catat")) {
        String
        tUserName=request.getParameter("user
        _name");
        String
        tPassword=request.getParameter("user
        _password");
        String
        tNamaLengkap=request.getParameter("n
        ama_lengkap");
        String
        tEmail=request.getParameter("email")
        ;
        String
        sAlamat=request.getParameter("alamat
        ");
        String
        tNotel=request.getParameter("notel")
        ;
if (oPelanggan.insertDataPelanggan(
   tUserName,tPassword,tNamaLengkap,
   tEmail,sAlamat,tNotel)) {
       tJudul="Pendaftaran Sukses";
         session.setAttribute( "Status",
               "Login" );
         session.setAttribute( "User",
    tUserName  ); }
    else
        {
       tJudul="Pendaftaran Gagal";
        }
}
```

Informasi Tentang Web Site

Untuk mengetahui informasi tentang web site "www.BatikJawa.com", klik *link* Tentang Kami dimana perintah *link* ini ada di file header.html (Tentang Kami), perintah ini akan mengarah ke file tentangkami.jsp, hasilnya seperti terlihat pada gambar 12.

Gambar 4.10 Tampilan Tentang Informasi Web

Diagram MVC dari Aplikasi yang Dibuat

Dari penjelasan skrip program diatas dapat dibuat diagram MVC dari aplikasi tersebut.

Gambar 4.11 Diagram MVC dari Aplikasi

Kesimpulan

1. Dengan menerapkan MVC model 1 pada aplikasi ini, akan terlihat jelas perbedaan fungsi file pada *Model*, *View* dan *Controller*.
2. Penerapan MVC model 1 ini akan memberikan secara jelas batas antara orang yang bertanggung jawab terhadap kode JSP (*View* dan *Controller*) dan orang yang bertanggung jawab terhadap kode Java (*Model*).
3. Dengan penerapan MVC model 1 ini dapat dijadikan sebagai dasar untuk pemahaman dan pembuatan suatu aplikasi dengan arsitektur MVC model 2 menggunakan Struts atau JavaServerFaces (JSF).

Saran

Dalam aplikasi ini masih perlu ditambahkan layanan-layanan yang lain, misalnya : konfirmasi pesanan dari pelanggan ke email dan layanan informasi biaya kirim.

DAFTAR PUSTAKA

[1]. Abdul Kadir, **Dasar Pemrograman Java 2**, Penerbit Andi, Yogyakarta.

[2]. Agus Setyabudi, Albert Samuel, **Aplikasi E-Commerce dengan Java Servlet dan JSP**, Penerbit Elex Media Komputindo, Jakarta.

[3]. Benny Hermawan, **Menguasai Java 2 & Object Oriented Programming**, Penerbit Andi, Yogyakarta.

[4]. Faisal Wiryasantika, **Aplikasi E-Ticketing Berbasiskan Model-View-Controller (Nvc) Pattern**, http://digilib.itb.ac.id/gdl.php?mod=browse&op=read&id=jbptitbpp-gdl-faisalwiry-29448&q=Java

[5]. Frans Thamura, **Teknik Mengembangkan Aplikasi Enterprise dengan Teknologi Open Source berbasis Java**, Penerbit Andi, Yogyakarta.

[6]. Imam Fahrur Rozi, **MVC - Model, View, Controller**, http://ifrozi.wordpress.com/2008/01/09/mvc-model-view-controller/

[7]. Joyce Avestro, **JENI Pemrograman Web**, Jardiknas, Malang.

[8]. Kocil, **Java Web Application Framework**, http://www.benpinter.net/article.php?story=20030512050949 6

[9]. Matius Soesilo Wijono, G. Sri Hartati Wijono, S.Si., B. Herry Suharto, **Java 2 SE dengan JBuilder**, Penerbit Andi, Yogyakarta.

[10]. Rizqi Ardiansyah, **Mengenal Java Web TeknologiKomponen,** http://miimlc.metrodata.co.id/forum/blogs/ardiansyah/archive/2008/12/27/mengenal-java-web-teknologi-komponen.aspx

[11]. Sri Hartati Wijono, S.Si., B. Herry Suharto, Matius Soesilo Wijono. **Pemrograman Java Servlet dan JSP dengan NetBeans**, Penerbit Andi, Yogyakarta.

[12]. Wikipedia , **Java**, http://id.wikipedia.org/wiki/Java#Versi_Awal

Sertifikat

Diberikan kepada

Adi Kusjani, S.T.

Atas peran sertanya sebagai

Penyaji

Seminar Nasional Riset Teknologi Informasi 2009
dengan tema

"Ubiquitous Computing"

diselenggarakan di STMIK AKAKOM pada tanggal 8 Agustus 2009

Yogyakarta, 8 Agustus 2009
Ketua STMIK AKAKOM

Prof. Dr. Ir. Prayoto, M.Sc.

Ketua Panitia

Ir. Totok Suprawoto, M.M., M.T.

Sertifikat

Diberikan kepada

Adi Kusjani, S.T.

Atas peran sertanya sebagai

Peserta

Seminar Nasional Riset Teknologi Informasi 2009
dengan tema

"Ubiquitous Computing"

diselenggarakan di STMIK AKAKOM pada tanggal 8 Agustus 2009

Yogyakarta, 8 Agustus 2009
Ketua STMIK AKAKOM

Prof. Dr. Ir. Prayoto, M.Sc.

Ketua Panitia

Ir. Totok Suprawoto, M.M., M.T.