INFOMATEK
Volume 18 Nomor 2 Desember 2016

MODEL INTERAKSI APLIKASI WEB PADA TEKNOLOGI JAVA

Ade Sukendar[*)]

Program Studi Teknik Informatika
Fakultas Teknik – Universitas Pasundan

Abstrak: Aplikasi web adalah aplikasi yang berjalan dengan menggunakan koneksi jaringan menggunakan protokol HTTP. Interaksi aplikasi web dimulai ketika pengguna meminta (*request*) ke server sampai server memberikan tanggapan (*response*) kepada pengguna. Setiap interaksi yang terjadi akan mengakses bagian program, semakin lingkupnya besar maka program yang terdapat pada aplikasi akan semakin banyak pula. Penelitian ini akan melakukan studi terhadap model interaksi web pada teknologi Java dan keterkaitannya dengan proses perawatan terhadap model interaksi yang digunakan. Pada penelitian ini melakukan studi literatur dan melakukan berbagai eksplorasi tehadap penelitian-penelitian serupa yang sudah dilakukan sebelumnya berdasarkan referensi. Hasil akhir dari penelitian ini yaitu memberikan perbandingan setiap model interaksi yang digunakan pada tingkatan program dengan maupun tanpa framework.

Kata kunci: *Aplikasi Web, Model Interaksi, Java Web, Perawatan*

I. PENDAHULUAN

Latar Belakang

Aplikasi web dapat dieksekusi karena adanya interaksi antara pengguna dan server menggunakan protokol HTTP (*Hyper Text Transfer Protocol*). Dalam interaksi tersebut, pengguna akan meminta layanan ke *server* dan selanjutnya *server* memberikan layanan tanggapan yang diminta oleh pengguna (Pressman [1]). Pengguna meminta layanan ke *server* untuk melakukan fungsi atau proses tertentu pada halaman web yang ditempilkan di *web browser*. Permintaan data dari pengguna akan dikirimkan lewat jaringan dengan menggunakan protokol HTTP ke *server* atau dikenal dengan nama *web server.* Kemudian *web server* akan memproses permintaan dari pengguna untuk memproses fungsi tertentu dan kemudian mengambil halaman web yang diminta. Setelah itu, *web server* memberikan *response* ke pengguna berupa kode-kode HTML (*Hypertext Markup Language*) ataupun dokumen *markup language* yang akan membentuk halaman web di *web browser*, sehingga pengguna bisa mendapatkan fungsi dan halaman web yang diinginkan. Struktur program aplikasi web terdiri dari program untuk tampilan aplikasi web dalam bentuk bahasa *markup* dan *business logic* sebagai fungsional dari aplikasi

[*)] ade.sukendar@unpas.ac.id

tersebut [1]. Program untuk tampilan aplikasi web khusus untuk menampilkan antarmuka ke pengguna sedangkan program *business logic* berfungsi sebagai pemroses permintaan yang diminta oleh pengguna.

Seiring dengan perkembangan teknologi web, perubahan pengembangan aplikasi web ikut mengalami perubahan. Penambahan fitur, komponen, *library* dan lainnya akan ditambahkan ke program aplikasi web yang akan dibangun (Chung [2]). Dengan demikian, aplikasi web yang dibangun akan terus mengalami penambahan atau perubahan sehingga program menjadi bertambah besar dan rumit (Bereton [3]). Sehingga dibutuhkan suatu model yang dapat menangani hal tersebut, supaya aplikasi web yang dibangun bisa ditangani dengan mudah (Sukendar [4]).

Identifikasi Masalah

Berdasarkan dari latar belakang yang dijelaskan di atas, didapatkan beberapa identifikasi masalah sebagai berikut :

1. Model interaksi apa yang dapat diterapkan pada pengembangan aplikasi web berbasis Java?
2. Bagaimana implementasi model interaksi aplikasi web pada pengembangan aplikasi web berbasis Java dengan dan tanpa *framework*?
3. Bagaimana pengaruh implementasi model interaksi aplikasi web berbasis Java terhadap bagian-bagian program?

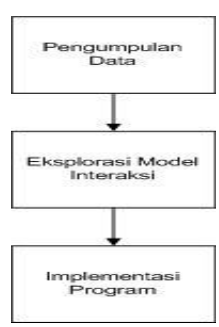
Tujuan Penelitian

Berikut ini merupakan tujuan dari penelitian yang dilakukan:

1. Mempelajari model interaksi aplikasi web pada teknologi Java.
2. Memahami dan membandingkan model interaksi aplikasi web pada teknologi Java.
3. Membuat aplikasi web berbasis teknologi Java berdasarkan model dan framework yang digunakan.

II. METODOLOGI

Pada penelitian ini terdiri dari tiga metode yaitu pengumpulan data, eksplorasi model interaksi dan implementasi program setiap model interaksi. Metode tersebut digunakan untuk mengetahui model interaksi dan implementasinya di lingkungan web berbasis Java yang diperlukan sebagai tahapan untuk menyelesaikan perancangan aplikasi pada penelitian ini (Gambar 1).



Gambar 1
Tahapan Penelitian

Pengumpulan Data

Tahapan penelitian ini dilakukan pengumpulan data dengan melakukan studi literatur yang berkaitan dengan topik yang diangkat. Studi literatur yang dilakukan yaitu dengan dengan mencari informasi terkait dengan penelitian yang sedang dilakukan berdasarkan penelitian yang sudah pernah dilakukan berdasarkan *paper*, jurnal atau artikel.

Eksplorasi Model Interaksi

Tahapan penelitian ini dilakukan untuk mengetahui model interaksi aplikasi web pada teknologi Java. Pada tahapan ini akan meninjau lebih dalam terhadap bagian program aplikasi web. Bagian program tersebut yaitu program untuk antarmuka pengguna, *business logic* dan tempat penyimpanan data. Bagian program ini akan disesuaikan dengan model interaksi aplikasi web.

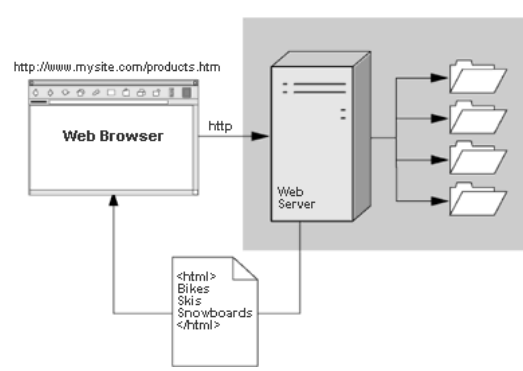
Setiap model interaksi akan diidentifikasi karakteristik dan penempatanya terkait dengan bagian program yang lain. Selain itu akan dikaitkan juga dengan fitur Java pada pengembangan aplikasi web. Sehingga setiap model mempunyai karakteristik yang berbeda. Selain itu, model interaksi yang diteliti terdiri dari dua yaitu model interaksi web tanpa dan dengan menggunakan framework.

Pada setiap model interaksi aplikasi web yang diteliti akan dibuatkan program-program kecil untuk mengetahui bagian-bagian pembentuk program. Selain bagian program dipergunakan juga diagram dan daftar file yang dapat mengidentifikasi setiap model interaksi yang sedang diteliti.

III. LANDASAN TEORI

Aplikasi Web

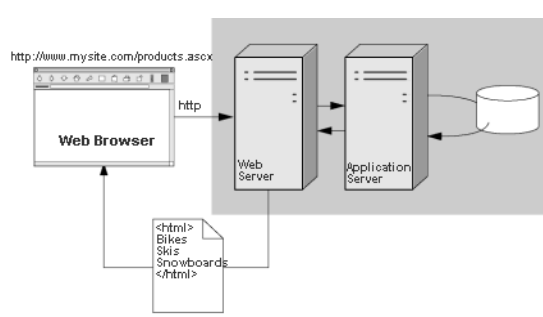
Aplikasi Web adalah perangkat lunak komputer yang terdiri dari kumpulan instruksi dan data yang menyediakan informasi dan fungsi bagi pengguna akhir [1]. Konten aplikasi web terdiri dari dua yaitu statis dan dinamis. Aplikasi web statis adalah halaman web yang tradisional dengan sifatnya tetap dan tidak bisa diubah, dalam kondisi yang sama halaman tersebut akan selalu tetap sama [5][6]. Permintaan dari banyak pengguna untuk halaman statis yang sama semuanya akan menerima hasil yang sama.



Gambar 2
Mekanisme Web Statis [5]

Pengkasesan *halaman web* statis, pengguna meminta *halaman web* dari *server* dan *server*

memberikan tanggapan dengan mengirimkan *file* yang diminta oleh pengguna. Selanjutnya pengguna menerima *halaman web* yang berada di *server*. Halaman *web* dinamis merupakan halaman *web* yang dihasilkan sesuai dengan waktu permintaan dari pengguna, mengubah konten sesuai dengan kategori yang sudah ditentukan [5], (Horstmann [6]). Setiap *request* yang berbeda dari *client* akan menghasilkan halaman *web* yang berbeda pula, sehingga konten *web* yang dihasilkan akan dibuat *on-the-fly* (atau *on-demand*) berdasarkan permintaan dari pengguna. Sebagai contoh, halaman *web* yang menampilkan waktu sekarang adalah dinamis, karena konten halaman *web* menggambarkan waktu yang sekarang. Halaman *web* dinamis dibangkitkan oleh aplikasi *server*, menerima masukan dari pengguna dan kemudian memberikan tanggapan terhadap permintaan tersebut.



Gambar 3
Mekanisme Web Dinamis [5]

Pengguna melakukan permintaan ke *server* dengan menggunakan aplikasi *web browser*, kemudian *web server* akan menerima permintaan dan memberikan permintaan pengguna ke dalam aplikasi *server* untuk pemrosesan dengan keluaran berupa data dalam bentuk HTML dan kemudian dikembalikan ke *web server*. W*eb server* akan mengirimkan tanggapan berupa hasil halaman HTML ke *browser*, dan halaman tersebut yang akan ditampilkan ke pengguna.

Protokol HTTP

HTTP merupakan kepanjangan dari *Hipertext Transfer Protocol*. HTTP memiliki sifat yang *generic* dan ringan, sehingga digunakan untuk mengirimkan teks *Hypertext Markup Language* (HTML) dan *Extensible Markup Language* (XML) (Jendrock [7]). HTTP merupakan protokol utama yang digunakan untuk mengrimkan informasi lewat *World Wide Web*.

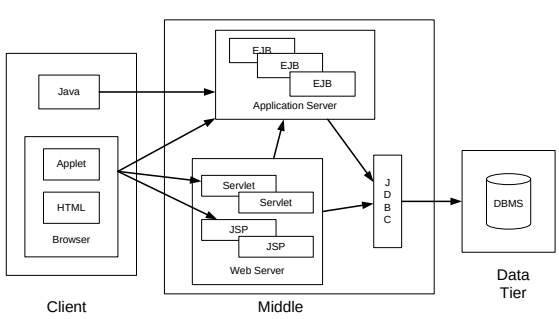
HTTP merupakan protokol yang tidak memiliki keterhubungan dengan permintaan dari pengguna. Dengan kata lain, ketika pengguna melakukan permintaan ke *server* maka *server* akan memberikan tanggapan dan kemudian transaksi akan ditutup. Permintaan yang kedua dari pengguna yang sama secara keseluruhan merupakan transaksi baru, tidak ada keterkaitan dengan permintaan yang sebelumnya.

Java Web

Java Web berada pada lingkungan platform Java edisi *enterprise* atau disebut dengan

Java Enterprise Edition (Java EE). Java EE adalah teknologi Java yang dirancang untuk memberikan solusi pengembangan aplikasi web dalam skala *enterprise,* merupakan suatu *blue print* dan sekumpulan API untuk pengembangan aplikasi *enterprise* berbasis Java (Falkner [8]), [9].

Java EE mempunyai tiga lapisan arsitektur terhadap sistem *enterprise*. Sistem yang mempunyai tiga lapisan atau lebih, membuktikan sistem tersebut lebih terukur dan fleksibel dibandingkan sistem *client server*. Sistem yang mempunyai berbagai lapisan yang telah dirancang dengan baik, setiap lapisan hanya akan bergantung pada lapisan yang berada dibawahnya. Misalnya perubahan pada suatu basis data tidak mengakibatkan terjadinya perubahan pada tampilan halaman web. Keterhubungan ketiga lapisan pada Java EE seperti terlihat pada gambar berikut [7]:



Gambar 4
Arsitektur Aplikasi Java EE

Berikut ini merupakan penjelasan dari tiga lapisan utama yang dimiliki Java EE:

a. *Enterprise Information Sistem (EIS)* yaitu lapisan yang juga disebut sebagai lapisan integrasi adalah lapisan yang terdiri dari sumber daya yang harus diakses oleh aplikasi Java EE dalam melakukan pekerjaannya, di dalamnya juga termasuk manajemen sistem basis data.
b. *Middle tier* yaitu lapisan yang terdiri dari objek bisnis yang dimiliki aplikasi dan menghubungkan sumber daya lapisan EIS. Komponen dari lapisan ini berada terpisah dari tampilan yang akan dipilih oleh user. Jika menggunakan EJB, maka lapisan ini akan dibagi menjadi dua bagian yaitu : EJB dan objek yang menggunakan EJB untuk mendukung tampilan.
c. *User interface tier* yaitu lapisan yang bertugas menampilkan objek bisnis pada *middle tier* kepada pengguna. Pada sebuah aplikasi web, lapisan ini terdiri dari *servlets*, *helper class* yang digunakan oleh *servlet*, dan komponen untuk menampilkan seperti halaman JSP.

Komponen Aplikasi Java EE

Platform Java EE menyediakan model-model komponen sebagai penyusunan aplikasi. Java EE memberikan kemudahan untuk membagi dan memecah sebuah aplikasi dengan menyusun komponennya kemudian mengintegrasikan menjadi sebuah aplikasi.

Setiap komponen tersebut dieksekusi didalam sebuah kontainer.

Berikut ini merupakan komponen-komponen yang dimiliki oleh Java EE [7]:

a. Komponen *client* terdiri dari aplikasi *client* yang mengakses server Java EE dan biasanya diletakkan pada mesin yang berbeda dengan *server*. Komponen client terdiri dari JavaBean dan Applet.

b. Komponen aplikasi web terdiri dari komponen-komponen yang dijalankan pada server. Komponen aplikasi web terdiri dari *servlet*, JSP dan tag lib.

c. Komponen bisnis terdiri dari komponen-komponen bisnis yang dijalankan pada server. Komponen ini terdiri dari *session bean*, *entity bean* dan *message driven bean.*

Model-View-Controller (MVC)

Permasalahan desain yang ditangani oleh MVC secara sederhananya terdiri dari tiga fungsi utama yaitu [9]:

a. Mengelola data di tempat penyimpanan data.

b. Membangun layer presentasi untuk *end-user*.

c. Mengelola logika pengkondisian yang memutuskan *screen* mana yang akan ditampilkan ke pengguna, apa yang terjadi ketika tampil *error*; dan mengetahui bagaimana dan kapan data diubah.

Arsitektur MVC membagi aplikasi kedalaman tiga *layer*, yaitu *model*, *view* dan *controller*. Setiap layer mempuyai tugas dan tanggung jawab masing-masing, selain itu menangani tugas yang spesifik dan mempunyai tanggung jawab yang spesifik untuk layer yang lain.

Organisasi Komponen Web

Presentation Layer

Presentation Layer atau lapisan presentasi adalah lapisan yang berinteraksi dengan antarmuka pengguna. Lapisan ini berisi logika presentasi untuk menampilkan informasi kepada pengguna. Logika tersebut berinteraksi dengan antarmuka pengguna dan elemen-elemen berbasis web, seperti HTML, *tag library* ataupun XML. Lapisan ini menjelaskan bagaimana informasi ditampilkan kepada pengguna, bukan bagaimana informasi tersebut diambil atau dipilih oleh aplikasi.

Controller Layer

Controller layer atau lapisan *controller* adalah lapisan yang mengatur alur aplikasi dan memberikan *service* sebagai penghubung antara antarmuka pengguna (*presentation layer*) dan aplikasi (*application layer*). Lapisan *controller* berisi logika *controller*, yang berguna untuk menerima dan menerjemahkan HTTP

request selanjutnya memutuskan langkah berikutnya dari aplikasi berdasarkan permintaan dari pengguna.

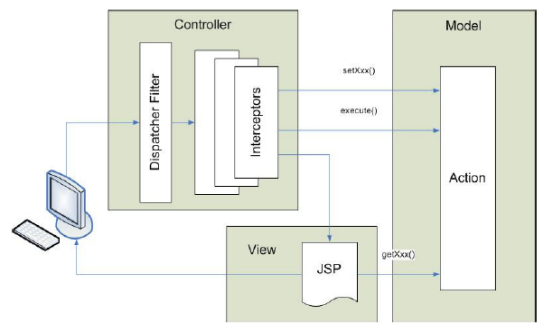
Application Layer

Application layer atau lapisan aplikasi adalah pusat dari aplikasi. Lapisan ini berisi logika aplikasi yang bertanggung jawab untuk proses aplikasi. Sebagai contoh, logika aplikasi di sistem registrasi berbasis web akan terdiri dari *submit* nama, mengecek validasi ke basisdata, tambahkan ke basisdata dan kemudian mengembalikan hasilnya.

Framework Struts2

Struts2 adalah *framework* yang menggunakan *design pattern* Model 2 menggunakan *interface* Filter (Kurniawan [10]), (Newton [11]), (Roughley [12]). *Framework* ini merupakan *framework action-based* dengan melakukan pemetaan URL *request* ke unit kerja yang disebut *action*. *Design pattern* Model-View-Controller di Struts2 mempunyai lima komponen utama, yaitu : *actions, interceptors*, nilai stack atau OGNL, tipe *result* dan *results* atau teknologi *view*.

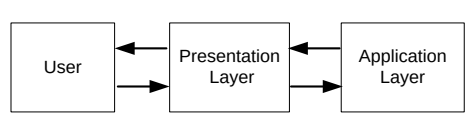
Gambar 5 merupakan arsitektur *high-level* Struts2 (Turner [13]) yang terdiri dari Model, View dan Controller. Controller diimplementasikan dengan *filter dispatch* Struts2 sebagai interceptors, Model diimplementasikan dengan actions dan View sebagai kombinasi tipe result dan results. Stack dan OGNL memberikan *thread*, *linking* dan kemampuan integrasi dengan komponen lain.



Gambar 5
Arsitektur Framework Struts2

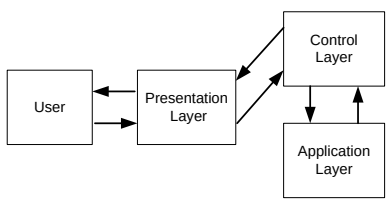
IV. EKSPLORASI

Teknologi Servlet dan JSP merupakan solusi untuk pengembangan aplikasi web berbasis Java. Teknologi JSP merupakan perluasan dari teknologi Servlet untuk membangun aplikasi Web dengan menyatukan kode Java ke dalam halaman HTML (Wijono [14]). Teknologi JSP dibagi ke dalam dua organisasi komponen, pembagian tersebut berdasarkan peran dan tanggung jawab setiap komponen. Organisasi ini memisahkan antara logika presentasi dari logika aplikasi. Pemisahan organisasi ini sering disebut dengan Model 1 atau di sebut juga dengan *page-centric* [15].



Gambar 6
Pemisahan Logika Presentasi

Organisasi komponen yang kedua memisahkan logika *controller* dari logika presentasi. Pemisahan komponen ini disebut dengan Model 2. Model ini telah memisahkan komponen interaksi antara antarmuka pengguna, aplikasi dan interaksi pengguna. Model ini sering menunjukkan sebagai model berorientasi Servlet, karena dalam implementasinya Model 2 ini kebanyakan menggunakan halaman JSP dan Java Servlet [15].

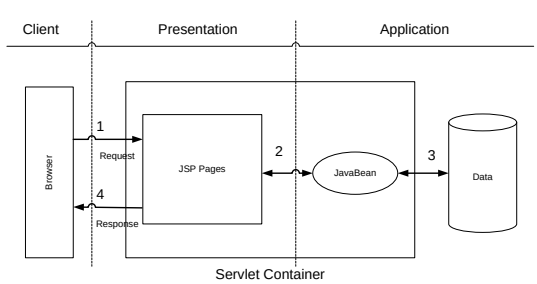


Gambar 7
Pemisahan Logika Controller

Model 1

Arsitektur Model 1

Arsitektur Model 1 yaitu kode fungsionalitas akan di simpan di manapun fungsionalitas diperlukan. Pendekatan ini sangat populer karena sederhana dan memberikan kecepatan dalam pengembangan aplikasi.



Gambar 8
Arsitektur Model 1

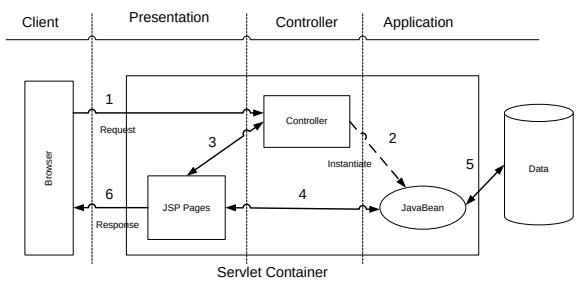
Aplikasi *browser* memberikan aplikasi web lewat sekumpulan halaman JSP sebagai lapisan presentasi. Sekumpulan halaman JSP tersebut diakses oleh *user* untuk diproses dari page pertama ke page yang berikutnya. Setiap halaman JSP dapat menggunakan JavaBean untuk melakukan proses aplikasi. Lapisan aplikasi ditangani oleh JavaBean untuk melakukan manipulasi ataupun mengolah informasi dari atau ke *database*. Hal yang perlu digarisbawahi dari arsitektur Model 1 ini bahwa setiap halaman JSP memproses *input user* oleh dirinya sendiri. Selain halaman JSP, sebuah servlet atau halaman HTML dapat mengantikan halaman tersebut.

Model 2

Arsitektur Model 2

Model 2 disebut juga MVC. Arsitektur Model ini mengambil manfaat dari kelebihan Servlet dan JSP. Servlet memiliki kelebihan dalam menangani tugas *processing-intensive,* sedangkan JSP memiliki kelebihan dalam presentasi data. Model 2 sudah melakukan pemisahan antara logika *controller* dan logika presentasi dalam lapisan yang berbeda. Logika *controller* akan berada di lapisan *controller* sedangkan logika presentasi akan tetap berada di dalam lapisan presentasi tanpa logika *controller*. Logika *controller* berfungsi untuk mengatur alur aplikasi yang akan di proses. Pemisahan Model 2

menggunakan istilah *design pattern* MVC, yaitu : Model, View dan Controller. Komponen *Controller* bisa menggunakan salah satu dari Servlet atau Filter yang dirancang untuk menerima permintaan dan memberikan tanggapan ke pengguna yang berada di lapisan *controller*. Berikut di bawah ini gambar arsitekur model 2.



Gambar 9

Arsitektur Model 2

V. IMPLEMENTASI PROGRAM

Model 1

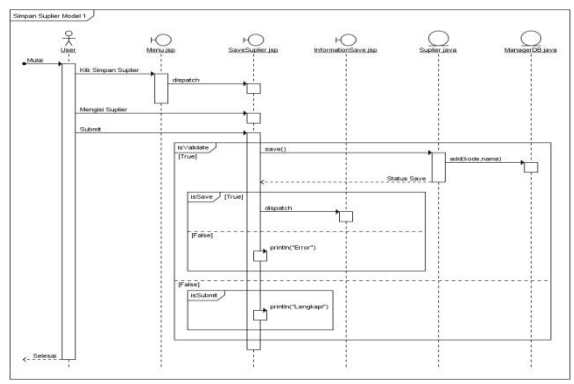
Implementasi Model 1 akan menggunakan kebutuhan pada tabel kebutuhan. Berikut di bawah ini adalah tabel kebutuhan aplikasi pengelolaan suplier.

Tabel 1

Kebutuhan Aplikasi Pengelolaan Suplier

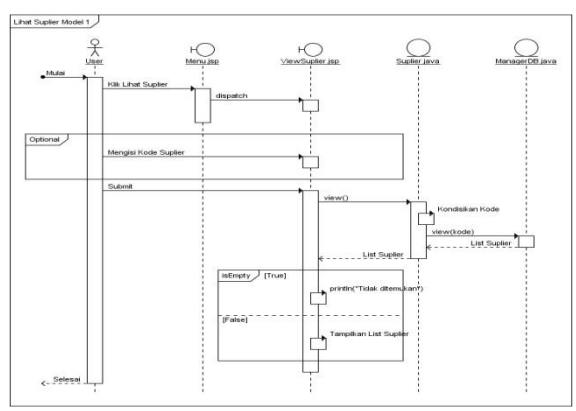
No	Kebutuhan	Penjelasan
1.	Simpan Suplier	Kebutuhan aplikasi untuk menyimpan data suplier.
2.	Lihat Suplier	Kebutuhan aplikasi untuk melihat data suplier sekaligus pencarian suplier.
3.	Ubah Suplier	Kebutuhan aplikasi untuk mengubah data suplier.
4.	Hapus Suplier	Kebutuhan aplikasi untuk menghapus data suplier.

Untuk mengetahui aliran proses dan objek yang terlibat dalam kebutuhan aplikasi maka digunakan diagram sequence. Berikut di bawah ini adalah gambar diagram *sequence* berdasarkan kebutuhan aplikasi di atas.



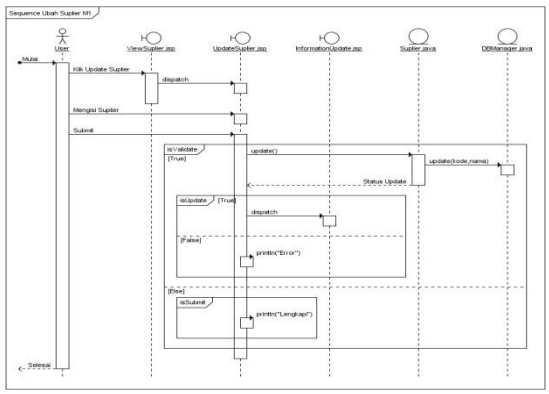
Gambar 10

Diagram Sequence Simpan Suplier Model 1



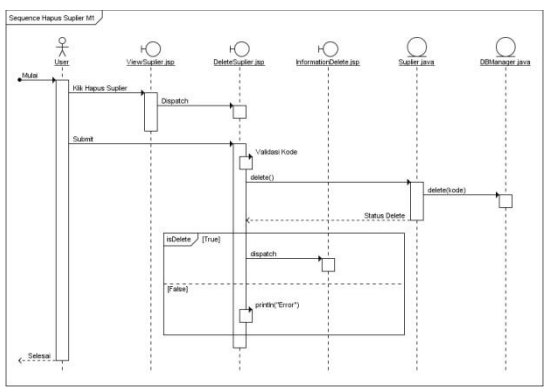
Gambar 11

Diagram Sequence Lihat Suplier Model 1



Gambar 12

Diagram Sequence Ubah Suplier Model 1



Gambar 13

Diagram Sequence Hapus Suplier Model 1

Berdasarkan diagram *sequence* diatas, contoh aplikasi tersebut memiliki sekumpulan halaman JSP dan JavaBean. Halaman JSP sebagai lapisan presentasi akan berinteraksi dengan pengguna, sedangkan JavaBean sebagai lapisan aplikasi melakukan pemrosesan aplikasi. Berikut ini daftar file aplikasi yang dibutuhkan sesuai dengan lapisan organisasi komponen Model 1.

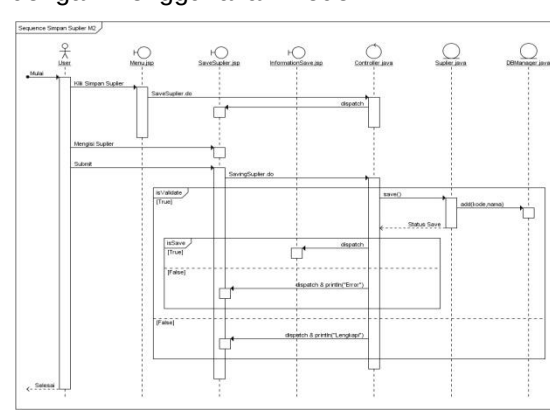
Tabel 2

Daftar File Aplikasi Model 1

Lapisan	File
Lapisan Presentasi	Menu.jsp SaveSuplier.jsp UpdateSuplier.jsp DeleteSuplier.jsp ViewSuplier.jsp InformasiSave.jsp InformasiUpdate.jsp InformasiDelete.jsp
Lapisan Aplikasi	Suplier.java ManagerDB.java

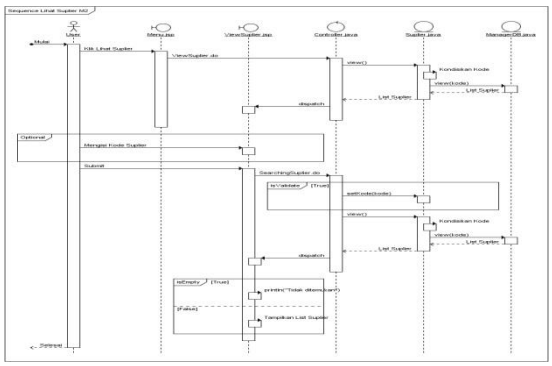
Model 2

Implementasi Model 2 akan menggunakan kebutuhan yang sama dengan Model 1 pada tabel kebutuhan x. Berikut di bawah ini adalah gambar diagram *sequence* diagram berdasarkan kebutuhan aplikasi di atas dengan menggunakan Model 2.



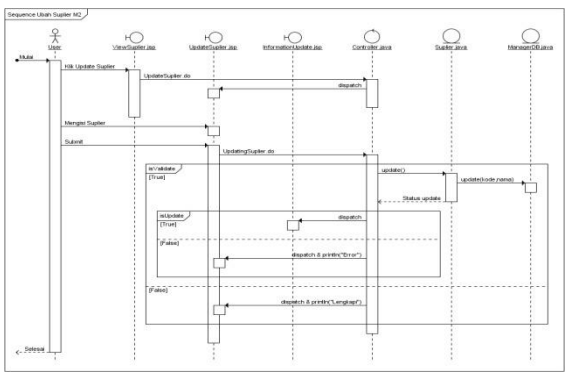
Gambar 14

Diagram Sequence Simpan Suplier Model 2



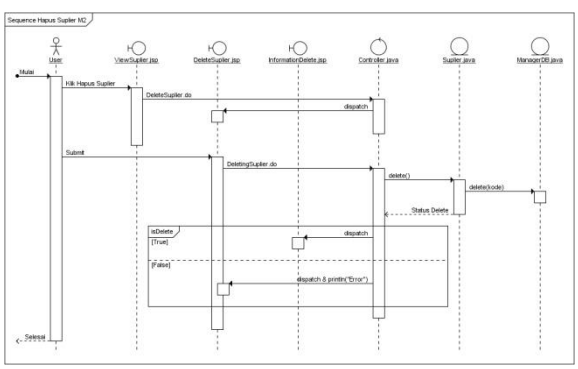
Gambar 15

Diagram Sequence Lihat Suplier Model 2



Gambar 16

Diagram Sequence Ubah Suplier Model 2



Gambar 17

Diagram Sequence Hapus Suplier Model 2

Diagram *sequence* di atas berbeda dengan diagram *sequence* Model 1, pada diagram tersebut mempunyai satu *layer* baru yaitu sebuah *controller*. Controller dibuat karena adanya pemisahan logika antara presentasi dan *controller*. Berikut ini daftar file aplikasi yang dibutuhkan sesuai dengan lapisan organisasi komponen Model 2.

Tabel 3

Daftar File Aplikasi Model 2

Lapisan	File
Lapisan Presentasi	Menu.jsp SaveSuplier.jsp UpdateSuplier.jsp DeleteSuplier.jsp ViewSuplier.jsp InformasiSave.jsp InformasiUpdate.jsp InformasiDelete.jsp
Lapisan Controller	Controller.java
Lapisan Aplikasi	Suplier.java ManagerDB.java

Berdasarkan tabel di atas, terlihat adanya lapisan *controller* yang digunakan untuk mengatur alur aplikasi. Lapisan tersebut hanya memiliki satu file yaitu Controller.java. Semua *event* dari pengguna akan ditangani oleh *controller* ini.

Controller yang berada di lapisan *controller* dapat menggunakan kelas Servlet maupun *interface* Filter. *Controller* yang menggunakan kelas Servlet harus melakukan *inheritance* dari kelas HTTPServlet, sedangkan *controller* yang menggunakan *interface* Filter harus melakukan *implement* terhadap *interface* Filter.

Pembahasan Model 1 dan Model 2

Perbedaan yang pertama adalah alur model interaksi di dalam sistem. Perbedaan tersebut bisa di lihat Tabel 4.

Tabel 4
Perbandingan Alur Model Interaksi

Model	Interaksi
Model 1	User → JSP → JavaBean → Database → JavaBean → JSP → User
Model 2	User → JSP → Controller → JavaBean → Database → JavaBean → Controller → JSP → User

Pembahasan yang kedua adalah pemisahan kode controller dari presentasi. Perbedaan tersebut bisa di lihat tabel di bawah ini :

Tabel 5
Perbandingan Kode Controller dan Presentasi

Lapisan	Model 1	Model 2
Presentasi	Validasi data Manipulasi data Dispatch Tampil data Pesan user	Tampil data Pesan user
Controller	-	Validasi data Manipulasi data Dispatch

Perbedaan yang ketiga adalah terletak pada pengembangan aplikasi. Perbedaan tersebut bisa di lihat tabel di bawah ini :

Tabel 6
Perbandingan Pengembangan Aplikasi

Model 1	Model 2
Berbasis Halaman JSP	Model View Controller

Model 2 Dengan Framework Struts2

Diagram *sequence* di atas berbeda dengan diagram sequence Model 2 sebelumnya. Dalam gambar tersebut memiliki sebuah struts.xml sebagai *controller* dan SuplierAction sebagai kelas *action*. Berikut ini daftar file aplikasi yang dibutuhkan :

Tabel 7
Daftar File Aplikasi Dengan Struts2

Lapisan	File
Lapisan Presentasi	Menu.jsp SaveSuplier.jsp

	UpdateSuplier.jsp DeleteSuplier.jsp ViewSuplier.jsp InformasiSave.jsp InformasiUpdate.jsp InformasiDelete.jsp
Lapisan Controller	struts.xml
Lapisan Aplikasi	SuplierAction.java Suplier.java ManagerDB.java xx-validation.xml

Berdasarkan tabel di atas, terlihat adanya file struts.xml sebagai *controller* yang digunakan untuk mengatur alur aplikasi. Semua *event* dari pengguna akan ditangani oleh *controller*.

VI. KESIMPULAN

Dari hasil eksplorasi model interaksi web pada teknologi Java didapatkan beberapa kesimpulan. Berikut ini beberapa kesimpulan yang didapatkan dari hasil penelitian:

1. Model 1 merupakan model pengembangan aplikasi web berbasis halaman JSP sedangkan Model 2 sudah memisahkan logika *controller* dari layer presentasi dan membentuk *layer* sendiri yaitu *layer controller*.
2. Model 2 terlihat lebih modular dibandingkan dengan Model 1 dengan adanya tambahan objek program sehingga menambah pekerjaan dalam pembuatan programnya.
3. Pemisahan logika presentasi dari lapisan presentasi memudahkan program untuk dibaca dan dipelajari.
4. Pembuatan aplikasi dengan menggunakan framework tidak terlalu membutuhkan pengkodean, untuk beberapa fitur aplikasi bisa dilakukan dengan cara konfigurasi sehingga bisa lebih produktif dalam mengembangkan aplikasi. Namun pengembang harus memahami *framework* yang akan digunakan untuk pengembangan aplikasinya.

VII. DAFTAR RUJUKAN

[1] Pressman, R.S., dan Lowe, D., *Web Engineering: A practitioners approach*. McGraw-Hill. New-York. 2009.

[2] Chung, S. dan Lee, Y. Modeling Web Applications Using Java And XML Related Technologies. *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)*. IEEE. 2002.

[3] Brereton, P., Budgen, D., dan Hamilton, G., Hypertext: The Next Maintenance Mountain. *IEEE Computer. December 1998. pp. 49-55*.

[4] Sukendar, Ade. Model Interaksi Web Pada Teknologi Java. Universitas Pasundan, Bandung, April 2010.

[5] About Dynamic Content. Data diperoleh melalui internet: http://www.adobepress.com/articles/ article.asp?p=31062&seqNum=3. Diunduh pada Januari 2010.

[6] Horstmann, C., *Core JSP*. San Jose, Prentice Hall. 2000.

[7] Jendrock, E., dkk. *The Java EE 5 Tutorial, Third Edition : For Sun Java System Application Server Platform Edition 9*. California, Sun Microsystems, Inc. 2006.

[8] Falkner, J. dan Jones, K., *Servlets and JavaServer Pages™: The J2EE™ Technology Web Tier*. Boston, Pearson Education. 2006.

[9] Integrating Servlet and JSP : The Model View Controller (MVC). Data diambil dari internet: J2EE training from the author: http://courses.coreservlets.com/. Diunduh pada Januari 2010.

[10] Kurniawan, B., *Struts2 Design and Programming: A Tutorial*. Brainy Software. 2008.

[11] Newton, D., *Apache Struts 2 Web Application Development*. Birmingham-Mumbai, Packt Publishing. 2009.

[12] Roughley, I., *Practical Apache Struts2 web 2.0 Projects*. USA, Apress. 2007.

[13] Turner, J. dan Berdell, J., Struts Kick Start. Pearson Education. 2003.

[14] Wijono, S. H., Suharto, H., Wijono, M. S., *Pemrograman Java Servlet dan JSP dengan NetBeans*. Yogyakarta, Penerbit Andi. 2006.

[15] JSP Architecture. Data Diperoleh dari internet: http://www.brainopolis.com/jsp/jspArchitecture.html. Diunduh pada Januari 2010.