WEB SERVER DEVELOPMENT SYSTEM WITH PHP, MYSQL, AND JSP CONFIGURATION

Rio Yohanes[1], Suyanto Edward Antonius[2]
[1,2]Program Studi Teknik Informatika, Universitas Katolik Soegijapranata
[1]13020101@student.unika.ac.id, [2]seantonius@unika.ac.id

Abstract

Web server is a service to accept request from client and gives response to the client. This project builds a web server using Java Programming Language and Java Socket Programming. Java Socket Programming is used to make a connection between client and server. The web server will accept and reply the requests. This web server can execute PHP scripts and JSP scripts, including PHP scripts which access MySQL databases.

Keywords: Client-Server Programming, Java, Socket, Web Server, JSP, PHP, MySQL

Pendahuluan

Web server adalah sebuah servis untuk menerima *request* dari *client* dan memberikan *response* kepada *client*. Gambaran mudahnya yaitu ketika *client* mengakses sebuah halaman web dari *web server*, pada dasarnya *client* mengirimkan *request* kepada *web server*. Lalu, *web server* menerima *request* tersebut dan memberikan *response* balik kepada *client*. Jadi *client* bisa menerima *response* tersebut berupa halaman *web* yang di telah *request*.

*Web server*akan dibuat berdasarkan bahasa pemrograman Java. Ada beberapa fitur yang diberikan oleh Java untuk membantu dalam proses pembuatan *web server* contohnya seperti *java.net.ServerSocket*, *java.net.Socket*, dan lain lain. Hasil akhir proyek ini adalah *web server* dapat digunakan dengan mudah, *web server* dapat mengeksekusi fungsi PHP dan JSP serta *web server* dapat mengakses MySQL *database*.

Landasan Teori

Web Server

Tohmasz Muldner menulis dalam jurnalnya “Analysis of Java *Client*/Server and Web Programming Tools for Development of Educational Systems” mendefinisikan webserver sebagai sebuah aplikasi terpusat adalah sebuah aplikasi yang berjalan di dalam sebuah mesin. Sebuah aplikasi yang

didistribusikan dapat terdiri dari sejumlah komponen yang terdapat di beberapa jaringan komputer. Untuk tipe terakhir dari sebuah aplikasi, client adalah komponen dari sebuah aplikasi yang membuat *request* kepada komponen yang lain dari dari sebuah aplikasi yang dapat disebut *server*. Hal ini dapat disederhanakan bahwa *web server* adalah sebuah servis untuk menerima *request* dari *client* dan mengirim *response* kepada *client*.

Java Socket Programming

Malik Prerna menulis dalam jurnalnya "Network Programming in Java Using Sockets IJIRT Volume 1 Issue 6" mendefiniskan*Socket* menyediakan mekanisme komunikasi antara dua komputer menggunakan TCP. Program *client* membuat sebuah *socket* di akhir komunkasinya dan berupaya untuk menyambungkan *socket* tersebut dengan *server*. Ketika komunikasi sudah terbentuk, *server* membuat *socketobject* di akhir komunikasinya. *Client* dan *server* sekarang dapat berkomunikasi denga menulis dan membaca dari *socket*tersebut ". Hal ini dapat di sederhanakan bahwa *client* dan *server* menggunakan *socket* untuk berkomunikasi. *Socket* menyediakan komunikasi antar *client* dan *server* menggunakan TCP. *client* membuat *socket* di akhir komunikasinya dan mencoba untuk terhubung dengan *server*.

PHP5 CLI (Command Line Interface)

PHP berfokus pada *Server-side scripting.* Di dalam kasus ini, *web server* menggunakan PHP5-CLI untuk mengeksekusi fungsi PHP yang tertanam di halaman *web*. Cara kerja PHP5-CLI tegolong sederhana, Java akan mengeksekusi PHP melalui *command line interface* untuk mengeksekusi fungsi PHP yang terdapat dalam halama*web*. Lalu, hasilnya akan dikirim kepada *client*.

Metodologi Penelitian

Pembuatan proyek ini terbagi menjadi 6 (enam) langkah, yakni:

1. Menentukan Aplikasi

   Proyek yang akan dikerjakan adalah aplikasi *web server*. *web server* akan dibuat berbasis bahasa pemrograman Java. *Web server*akan ditugaskan utnuk menerima *request* dari *client* dan mengirimn*response* kepada *client*.

2. Mencari Referensi

   Pencarian referensi berfungsi untuk memahami hal dalam bagaimana *web server* bekerja, bagaimana *web server* dapat berkomunikasi dengan *client* dan seterusnya.

3. Analisis

   Menemukan cara bagaimana *web server* dapat menerima *request*, mengirim *response*, mengeksekusi fungsi PHP, mengeksekusi fungsi JSP dan mengakses *database* MySQL.

4. Desain

   Merencanakan tentang cara kerja *web server* dan memasukannyakedalam*use case diagram, flow chart,* dan *class diagram*.

5. Implementasi

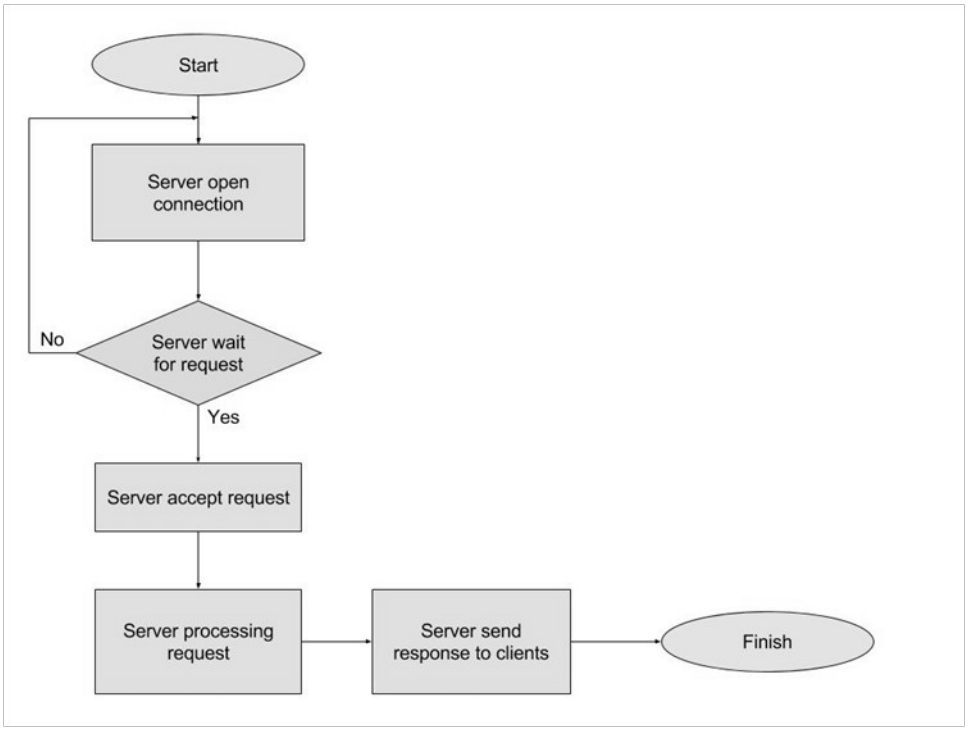
   Setelah itu, implementasikan*web server* dan mencobanya di *server* lokal.

6. Testing

   Membuat halaman *web* untuk menguji fitur seperti PHP dan JSP serta membuat *log* untuk menunjukan*request* dari *client*.

Hasil dan Pembahasan

Berikut adalah proses proyek *web server* bekerja disajikan dalam bentuk diagram berikut ini:



Gambar 1: Diagram Web Server

Server membuka *socket* untuk menerima *request* dari *client*. *Server*akan menunggu sampai mendapatkan *request* dari *client*. Setelah menerima *request*, *web server*akan memproses *request* tersebut dan mengirim *response* kepada *client.*

Proyek *web server* ini terbagi menjadi 9 *class*yaitu *Main class, ConnectionHandler class, HttpRequest class, configReader class, HttpResponse class, IkomLog class, JspResponse class, JavaResponse class dan PhpResponse class*. *Main class* berfungsi untuk pembuatan *socket* dan menerima reques dari *client* melalui *socket*. Lalu, *ConnectionHandler class*akan membaca *request* dari *client* dan meneruskan *request* ke class berikutnya untuk di proses. Proses *request* berlangsung dari *HttpRequset class* yang berfungsi untuk pengambilan nama halaman *web* yang di *requestclient*. Lalu, nama yang sudah diperoleh dibawa ke *class* untuk menyusun *response* disertakan *headerresponse* dan halaman *web*. Sebelum di kirim ke *client*, *HttpResponse class*akan melakukan pengecekan halaman *web*terleih dahulu apakah terdapat fungsi-fungsi seperti PHP atau JSP agar dieksekusi terlebih dahulu karena *client* tidak bisa mengeksekusi fungis PHP maupun JSP. Setelah itu, baru *web server* mengirimkan *response* kepada *client*.

Ada beberapa *class* pendukung *web server* yaitu *ConfigReader class* dan *IkomLog class*. *ConfigReader class* berfungsi untuk membaca *file configuration* bertipe *properties* untuk mempermudah konfigurasi *web server* tanpa harus menyentuh *source code*. *IkomLog class*berfungsi untuk pembuatan *log* yang mencatat informasi *client* yang terhubung dan *request* yang masuk.

Dalam pengeksekusian PHP, IKOM Web Server menggunakan program PHP 5 CLI. *Web server* mengeksekusi PHP melalui *command line interface* untuk mendapatkan hasil dari fungsi PHP. Dalam pengeksekusian JSP, IKOM Web Server menggunakan program Tomcat 6.

Berikut adalah tampilan dari halaman *web* yang disediakan oleh *web server* untuk menguji fitur-fitur seperti PHP, JSP, MySQL, dan lain-lain.

Halaman Test PHP ini melakukan uji coba terhadap fungsi php di bawah ini.

```
<?php
echo "Hello World";
?>
```

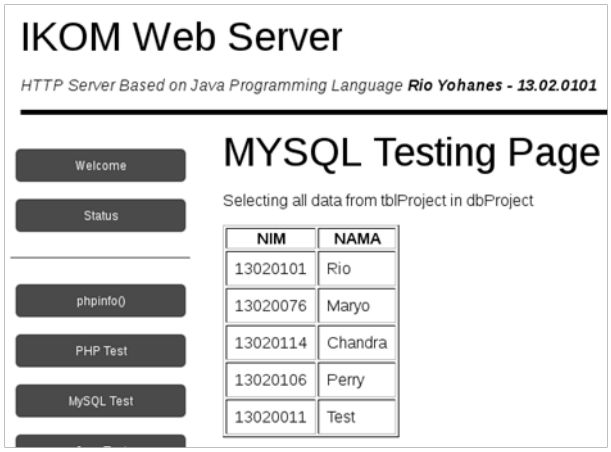
Hasilnya dapat dilihat pada Gambar di bawah ini.

Gambar 2: Halaman Test PHP

Hasil eksekusi fungsi PHP untuk menampilkan data dari *database*, disajikan dalam bentuk kode program di bawah ini

```
<?php
$dbhost = 'localhost';
$dbuser = 'root';
$dbpass = '';
$conn = mysql_connect($dbhost, $dbuser, $dbpass);
mysql_select_db('dbProject', $conn);
if(! $conn ) {die('<i>ERROR - Could not connect </i> : <b>' .
mysql_error());}
$sql = 'SELECT * FROM tblProject';
$retval = mysql_query( $sql, $conn );
if(! $retval ) {die('<i>ERROR - Could not get data </i> : <b>' .
mysql_error());}
echo
" <table border = 1>
<tr><th>NIM</th><th>NAMA</th></tr>";
while($row = mysql_fetch_array($retval, MYSQL_ASSOC)) {
echo
"
<tr>
<td>".$row['nim']."</td>
<td>".$row['nama']."</td>
</tr><?php include 'footer.php'; ?>
"; }
echo "</table>";
mysql_close($conn);?>
```

Hasil eksekusi program di atas, disajikan dalam bentuk gambar berikut ini:



Gambar 3: Halaman Test MySQL

Kode program dan gambar berikut adalah hasil eksekusi fungsi JSP untuk menampilkan *String* dan *Switch case*, berikut kode yang di uji coba.

```
<% out.println("Hello World From JSP Sidang"); %>
public String getQuarter(inti){
String quarter;
switch(i){
case 1: quarter = "Winter";
break;
case 2: quarter = "Spring";
break;
case 3: quarter = "Summer I";
break;
case 4: quarter = "Summer II";
break;
case 5: quarter = "Fall";
break;
default: quarter = "ERROR";}
return quarter;}
%><% out.println(getQuarter(1)); %>
```

Gambar 4: Halaman Test JSP

Kesimpulan

Dalam proyek akhir ini dapat disimpulkan bahwa membuat *web server* berbasis bahasa pemrograman Java adalah suatu hal yang tidak mustahil. Java menyediakan bebrapa *library* untuk memebatu dalam pembuatan *web server* seperti *java.net.ServerSocket*, *java.net.Socket* untuk komunikasi antara *client* dan *server*. Untuk menerima *request* dan mengirim *response*, digunakan *library* seperti *BufferedReader* dan *PrintWriter.*

Bagian tersulit dari proses pembuatan *server* adalah penambahan fitur untuk mengeksekusi fungsi PHP dan JSP. Jadi, untuk membuat sebuah *web server* memerlukan implementasi yang cukup sulit.

Daftar Pustaka

[1] Malik P, Rawat P. (2014).Network Programming in Java Using Socket. *International Journal of Innovative Research in Technology,* Vol. 1, No. 6, 1377-1387.

[2] Tasneem S, Ammar R. (2012, August). Performance Study of a Distributed Web Server: An Analytical Approach. *Journal of Software Engineering and Applications*, 5, SciRes, 855- 863.

[3] Kalita, L. (2014). Socket Programming. *International Journal of Computer Science and Information Technologies,* Vol. 5, No. 3, 4802-4807.

[4] Falkner J, Jones K. (2003, September). *Servlets and Server Pages.* Boston, Pearson Education, 109.

[5] Mckenzie, C. (2011, March). Understanding How the Application Server's Web Container Works. Accessed at November 22, 2016 from: http://www.theserverside.com/feature/Understanding-How-the-Application-Servers-Web- Container-Works.