BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Dalam penulisan ini terdapat lima referensi dan satu referensi yang diajukan penulis sebagai tinjauan pustaka meliputi:

Table 2. 1 Perbandingan Tinjauan Pustaka

Parameter / **Penulis**	**Objek**	**Metode**	**Bahasa Pemrograman**	**Interface**
Wahyu Rifa'i Dwi Septian	*Fullstack Framework* PHP (Yii, Codeigniter, CakePHP, Symfony, Zend)	Moose CK dan AHP	PHP, SQL	Text
Agung Rahmat Saleh	Performa Arsitektur Kelengkapan fitur	Perbandingan Yii, dan Zend	PHP, SQL	GUI
Delipetrev, Janev Mile, Ristova Suzana	*Framework* PHP Yii, Codeigniter, Zend	Perbandingan Framework berdasarkan *library database*	PHP, SQL	Text
Muhammad	*Active Record*,	Metode	PHP, SQL	GUI

Joddy Gorby Cendraraja	Koneksi Ke *Database*, Kecepatan Akses dan *Load Time*, *Layout*, *Security Web*	Kualitatif		
Muhammad Nur Hamid	Inventaris di HMJTI STMIK AKAKOM Yogyakarta	Perbandingan performa, cara akses *database*, dan fitur AJAX	PHP, SQL	GUI
Khoirul Anam	*Microframework* PHP (FatFree, Lumen, Phalcon-micro, Slim)	Perbandingan performa dengan *load testing*, Penanganan jenis uji pada plaintext, JSON *serialization*, *single* dan *multiple query*	PHP, SQL	Text

Penelitian yang dilakukan memiliki perbedaan dengan penelitian sebelumnya baik objek maupun metode penelitian yang digunakan. Objek yang akan gunakan

pada penelitian ini adalah *microframework* PHP (FatFree, Lumen, Phalcon-micro, Slim). Objek tersebut akan diuji performanya menggunakan metode *load testing*. Pengujian performa menghasilkan data *request per second* dan *latency* sebagai *output* dari jenis pengujian meliputi plaintext, JSON *serialization*, *single query*, dan *multiple query*.

2.2 Dasar Teori

2.2.1 Arsitektur N-Tier

Arsitektur *N-Tier* adalah perangkat lunak yang memiliki beberapa lapisan yang dapat membedakan peran pada masing masing lapisannya. Pada lapisan presentation akan memiliki perbedaan peran pada lapisan *logic* dan lapisan data (Martin & Amir, 2015).

N-Tier adalah sebuah sistem yang setidaknya memiliki tiga bagian *logic* yang terpisah. Setiap bagian tersebut bertanggung jawab atas fungsinya secara spesifik (Stephen Watts, 2017).

Dari definisi diatas dapat disimpulkan arsitektur N-Tier adalah seperangkat sistem dengan lapisan-lapisan yang memiliki fungsi dan kerja secara spesifik.

2.2.2 Middleware

Dalam definisinya, *middleware* adalah definisi umum dari perangkat lunak yang berperan untuk “mengelem bersama” sesuatu yang terpisah, yang sudah komplek atau sudah ada, pada program. Beberapa komponen perangkat lunak memiliki keterikatan dengan *middleware* termasuk aplikasi berskala

enterprise atau layanan web (Rouse, 2015).

Teknologi *middleware* merupakan sebuah definisi umum dari sebuah perangkat lunak, sedangkan API adalah sebuah definisi secara teknis pada implementasinya (Suominen, 2017).

Pada implementasinya, apa pun yang memenuhi jenis *request*, maka API tersebut akan disebut sebagai *middleware*. Sehingga dari definisi tersebut dapat disimpulkan, *middleware* merupakan sebuah perangkat lunak yang berfungsi sebagai jembatan penghubung antar lapisan sebuah sistem yang memberikan fasilitas untuk saling berkomunikasi melalui sebuah jalur komunikasi yang sudah diterapkan.

2.2.3 REST (Representational State Transfer)

REST *(Representational State Transfer)* merupakan standar arsitektur komunikasi berbasis web yang sering diterapkan dalam pengembangan layanan berbasis web. REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000. Arsitektur REST pada umumnya dijalankan melalui HTTP *(Hypertext Transfer Protocol)*, melibatkan proses pembacaan halaman web tertentu yang memuat sebuah file XML atau JSON. Setiap permintaan bersifat independen, *server* tidak menyimpan keadaan permintaan apapun. Metode yang digunakan dalam REST diantaranya:

- GET, menyediakan hanya akses baca pada *resource*.
- POST, digunakan untuk memperbarui *resource* yang ada atau membuat *resource* baru.

- PUT, digunakan untuk menciptakan *resource* baru.
- DELETE, digunakan untuk menghapus *resource*.
- OPTIONS, digunakan untuk mendapatkan operasi yang didukung pada *resource*.

2.2.4 Uji Performa

Definsi dari uji performa adalah sebuah proses yang mengindentifikasi efektivitas dari sebuah komputer, jaringan, perangkat lunak atau perangkat. Proses yang dilakukan yaitu mengindentifikasi secara kualitatif di dalam laboratorium, yaitu mengukur waktu respon (Hass, 2008).

Uji performa adalah untuk memberikan validasi atas kecepatan, kestabilan dan keleluasaan sebuah sistem supaya memberikan sebuah gambaran pada hasil akhir dari sebuah aplikasi yang diujikan berdasarkan ketentuan-ketentuan yang akan dilaksanakan selama proses pengujian (Murawski, Keck, & Schnaible, 2014).

Metode pengujian performa pada penelitian ini menggunakan *load testing*. Tujuan dari *load testing* adalah untuk melakukan verifikasi terhadap aktivitas dari aplikasi uji apabila diberikan situasi yang normal untuk sebuah sistem yang diujikan.

2.2.5 Phalcon-micro

Project Phalcon sebenarnya sudah dimulai sejak 2012, kemudian baru meraih *stable release* tepatnya pada 6 Juni 2014. *framework* ini ditulis dalam bahasa C, C++, dan PHP. Catatan penting yang harus diketahui, ternyata

wujud Phalcon adalah PHP C-Extension. Phalcon tidak ditulis dalam script PHP dengan tujuan untuk menangani lebih banyak *request*. Pada perkembangannya, Phalcon mempunyai beberapa fitur salah satunya adalah Phalcon-micro. Aplikasi mikro dari Phalcon tersebut memungkinkan pengembang aplikasi untuk membuat fungsi seperti *microframework* dengan baris kode minimal.

2.2.6 FatFree

FatFree *framework* adalah salah satu *microframework* PHP yang dapat digunakan untuk memudahkan pembuatan aplikasi web dinamis. *framework* FatFree adalah *framework* PHP yang *open source*, ditulis oleh Bong Cosca pada tahun 2009 sampai sekarang. Filosofi dibelakang *framework* ini dan pendekatannya ke arsitektur software adalah dengan meminimalkan dalam komponen struktural, dan menghindari kompleksitas aplikasi dan memberi keseimbangan antara kode, performa aplikasi dan produktivitas pengembang aplikasi.

2.2.7 Lumen

Lumen adalah sebuah PHP *framework* terbaru yang dibangun oleh Taylor Otwell yang tidak lain adalah pencipta dari Laravel. Lumen adalah sebuah *microframework* yang dirancang khusus untuk kebutuhan *micro services* dan juga kebutuhan API. Karena *framework* ini bersifat *micro*, Taylor Otwell lebih mengedepankan soal kecepatan untuk akses data.

2.2.8 Slim

Slim *framework* dibangun oleh Josh Lockhart, seorang senior *developer*

dari newmediacampaigns.com dan dia adalah *"The man behind" PHP The Right Way*. Dikatakan *microframework* karena Slim adalah *framework* yang fokus pada kebutuhan pokok yang diperlukan sebuah aplikasi web seperti: menerima sebuah HTTP *request*, mengirimkan *request* tersebut pada *code* yang sesuai, dan mengembalikan HTTP *response*. Slim *framework* merupakan *framework* yang ditulis dengan menggunakan bahasa pemrograman PHP. Slim mengakomodasi fungsi-fungsi yang terkait dengan pembuatan API *(Apllication programming interfaces)*.