BAB 2 LANDASAN KEPUSTAKAAN

Bab ini membahas tinjauan pustaka yang meliputi kajian pustaka dan dasar teori yang nantinya digunakan untuk mendukung serta menjadi dasar untuk penelitian ini. Kajian pustaka membahas penelitian sebelumnya yang telah dilakukan dan berhubungan dengan penelitian ini. Dasar teori membahas tentang teori yang berhubungan dan diperlukan untuk menyusun penelitian.

2.1 Kajian Pustaka

Terdapat beberapa penelitian yang telah dilakukan sebelumya yang berhubungan dengan penelitian yang akan dilakukan ini. Tabel dibawah menunjukkan kajian pustaka yang dilakukan.

Tabel 2.1 Kajian Pustaka

No	Judul	Tahun	Penulis	Hasil Penelitian	Penelitian Penulis
1	Automatic Evaluation System for Student Code	2015	Pratik Saraf, Shankar Ramesh, Sachin Patel dan Prof. Sujata Pathak	Sistem evaluasi kode bahasa Java berbasis web	Menambahkan penggunaan *docker* untuk menjalankan kode program dan menambah bahasa yang bisa diuji.
2	Docker as Platform for Assignments Evaluation	2015	František Špaček, Radomír Sohlich dan Tomáš Dulík	Sistem evaluasi kode dengan menggunakan Docker	Menambahkan beberapa sistem penguji untuk menambah efisiensi dan mempercepat pengujian dan menggunakan

					load balancing untuk membagi tugas antar sistem serta menggunakan RPC.
	Security Testing as a Service with Docker Containerization	2017	P.P.W. Pathirathna, V.A.I. Ayesha, W.A.T. Imihira, W.M.J.C. Wasala, Nuwan Kodagoda, E. A. T. D. Edirisinghe	Sistem untuk Menguji Keamanan Aplikasi Berbasis Web menggunakan Docker Containerization	Menambahkan beberapa sistem penguji untuk menambah efisiensi dan mempercepat pengujian dan menggunakan *load balancing* untuk membagi tugas antar sistem serta menggunakan RPC.

2.2 Dasar Teori

Untuk melaksanakan penelitian ini dibutuhkan referensi dan dasar teori untuk melakukan penelitian. Dengan adanya referensi dan penelitian maka akan memperlancar proses perancangan dan penelitian. Berikut adalah beberapa teori yang akan digunakan untuk menjadi dasar penelitian:

2.2.1 *Unit Testing*

Unit Testing merupakan praktek pengujian fungsi atau unit tertentu pada sebuah kode tertentu(McFarlin,2012). Dengan melakukan unit testing maka penulis kode program bisa menguji apakah fungsi yang terdapat pada kode program memberikan keluaran yang sesuai. Melakukan unit testing sangat membantu dalam meningkatkan kualitas sebuah kode. Selain itu jika pembuat program sering melakukan unit testing, maka akan mempermudahnya untuk menuliskan kode program lain lebih baik lagi.

Untuk melakukan *unit testing* dapat dilakukan dengan cara membuat *test case*. *Test case* dapat dibuat dengan mempersiapkan input yang akan diujicobakan ke kode program dan *file* jawaban. *File* jawaban akan dicocokkan dengan keluaran dari kode program untuk menentukan benar atau tidaknya keluaran kode program. Dari beberapa test case yang dijalankan hasilnya akan dirangkum sehingga mengetahui apakah kode program sudah benar-benar berjalan sesuai dengan yang diinginkan.

2.2.2 Python

Python merupakan bahasa pemrograman tingkat tinggi yang berorientasi objek. *Python* memiliki sintaks yang mudah dibaca, sehingga cocok untuk orang yang baru belajar memahami program. *Python* dapat digunakan untuk berbagai macam jenis pemrograman seperti pengembangan web, pemrograman jaringan, dan komputasi matematika. *Programmer python* merupakan salah satu dari 5 *programmer* dengan bahasa lain yang paling dicari dan memiliki gaji tinggi(Heath,2017). Itulah mengapa pada penelitian ini akan melakukan evaluasi dari kode bahasa Python.

Untuk menjalankan *master*, akan dibuat *server* berbasis bahasa python. Python digunakan karena mendukung pemrograman jaringan. Pada python, terdapat dua level akses ke servis jaringan. Pada *low level*, kita bisa mengakses *socket* pada sistem operasi sehingga bisa membuat koneksi. Python juga menyediakan *library* yang bisa mengakses level yang lebih tinggi pada application layer seperti HTTP,FTP, dan lain-lain. Pada penelitian ini akan digunakan salah satu *library* yang ada yaitu RPyC untuk melakukan RPC yang nanti akan dijelaskan lebih lanjut pada subbab RPC.

2.2.3 Java

Bahasa pemrograman Java merupakan bahasa pemrograman yang berbasis pemrograman berorientasi objek. Bahasa Java telah banyak digunakan untuk mengembangkan aplikasi pada berbagai *environment*. Karena itulah *programmer* Java juga merupakan *programmer* yang paling banyak dicari dengan gaji tinggi. Java merupakan turunan dari bahasa C, sehingga sintaksis dari Java mirip dengan bahasa C. Karena alasan yang telah disebutkan, maka pemrograman dengan bahasa java juga digunakan dalam penelitan ini.

2.2.4 C++

Bahasa pemrograman C++ juga bisa diuji pada sistem yang akan dibuat. Bahasa C++ juga seringkali dipakai untuk mengembangkan program. Bahasa ini sejak tahun 1998 telah distandarisasi oleh ISO. Bahasa ini juga disusun langsung ke bahasa mesin, sehingga jika dioptimalkan bisa menjadi bahasa tercepat. Bahasa C++ memiliki banyak *library* yang dapat digunakan. Selain itu bahasa ini bisa untuk berbagai paradigma, salah satunya pemrograman berorientasi objek.

2.2.5 Remote Procedure Call

RPC merupakan protokol dimana sebuah program dapat meminta servis dari program lain di sebuah jaringan. RPC termasuk jenis model arsitektural jaringan yang melihat dari sisi berkomunikasi/*Communication Paradigm* yaitu Remote Invocation. RPC menggunakan model klien-*server*. Program yang melakukan request adalah klien sedangkan yang memberikan servis adalah *server*. RPC merupakan operasi yang tersinkronisasi sehingga jika terdapat operasi berjalan, program lain yang meminta servis akan ditunda terlebih dahulu. Tetapi dengan menggunakan thread, maka operasi dapat dilakukan secara parallel.

2.2.5.1 RPyC

RPyC merupakan *library* python untuk RPC, klastering dan komputasi terdistribusi. RPyC menggunakan teknik object-proxying, sehingga remote object dapat dimanipulasi seolah-olah objek tersebut lokal. Untuk menginstall RPyC dapat menggunakan perintah “pip install rpyc”. Untuk memakainya, harus terlebih dahulu memasukkan *library* pada kode program dengan perintah “import rpyc”. *Library* ini akan digunakan untuk komunikasi antara *master* dan *slave*.

2.2.6 Docker

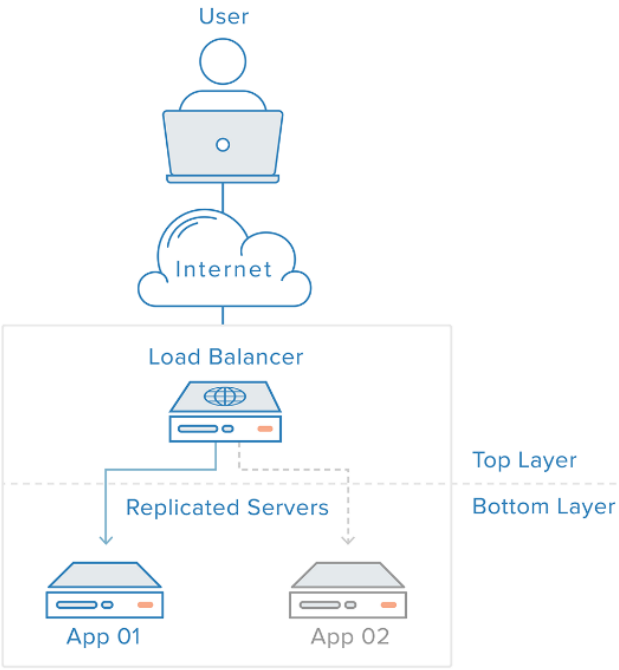
Docker merupakan aplikasi *open source* yang berfungsi untuk membungkus perangkat lunak dan semua dependensinya(Sîrbu, 2017). Pengaturan dan hal pendukung lainnya akan dibangun menjadi sebuah *image*. Hasil instansiasi dari image tersebut dinamakan *container*. *Container* inilah yang nantinya bisa digunakan untuk menjalankan perangkat lunak.

Docker memiliki kelebihan untuk dapat mengisolasi container dari *host* maupun *container* lain(Sîrbu, 2017). Ini karena docker menggunakan *kernel namespace*. Pada saat *container* dijalankan, docker akan membuat satu set *namespace* dan *control group* sehingga proses dari container tidak akan mengganggu *host* dan container lain. *Container* juga memiliki pengaturan jaringan tersendiri sehingga sebuah container tidak akan memiliki hak akses dari *socket* atau *interface* container lain.

2.2.7 *Load balancing*

Load balancing digunakan untuk mengarahkan pengguna ke salah satu *server* yang telah direplika sebelumnya. Dengan adanya *load balancing*, maka

beban *server* dalam menangani permintaan user akan terbagi. Pada gambar berikut merupakan ilustrasi dari *load balancing*.



Gambar 2.1 Ilustrasi *Load balancing*(Anderson,2017)

Pada *load balancing* terdapat beberapa algoritme yang dapat digunakan untuk menentukan *server* mana yang dipilih. Salah satunya adalah *round robin*. *Round robin* akan memilih *server* secara bergantian. Salah satu cara untuk mengimplementasikan algoritme ini adalah dengan menggunakan rumus modulo. *Load balancing* akan diimplementasikan pada *server* yang terdapat pada *master*.

2.2.8 Apache JMeter

Aplikasi ini termasuk aplikasi open-source. Aplikasi ini dibuat menggunakan Java dan digunakan untuk melihat sifat dan mengukut performa dari suatu program. Aplikasi ini lebih sering digunakan untuk melakukan tes pada aplikasi web, meskipun sekarang bisa digunakan untuk jenis aplikasi lainnya. Cara kerja dari aplikasi ini adalah dengan mensimulasikan *load* yang berat pada *server* untuk mengetahui kemampuan *server* atau menganalisis performa *server* pada tipe *load* yang berbeda-beda. Pada penelitian ini akan digunakan salah satu plugin yang terdapat pada JMeter yang digunakan untuk mengetes *websocket server* yaitu *websocket request-response sampler.*

2.2.9 SYSSTAT

SYSSTAT adalah paket dari banyak *tool* yang terdapat pada unix. Alat-alat pada SYSSTAT digunakan untuk memantau penggunaan sistem dan memantau performa dari suatu sistem. Terdapat dua alat pada SYSSTAT yang nantinya akan digunakan pada komputer ini yaitu mpstat dan sar.

Mpstat merupakan alat khusus untuk memantau CPU. Untuk menggunakan alat ini cukup menuliskan perintah "mpstat (*interval*) (*count*)". Perintah interval digunakan untuk mengatur setiap berapa detik alat ini akan menangkap penggunaan CPU dan menampilkannya. Perintah *count* digunakan untuk mengatur berapa kali alat ini akan memperlihatkan penggunaan cpu dari komputer. Pada salah satu kolom di keluaran mpstat akan menunjukkan berapa persen penggunaan CPU saat itu. Pada baris paling akhir semua nilai yang sudah dicatat akan dihitung rata-ratanya. Rata-rata dari persentase penggunaan CPU inilah yang nanti akan dicatat dan digunakan untuk mendapatkan hasil pengujian.

Sar adalah alat yang dapat digunakan untuk memantau penggunaan memori dari komputer. Untuk menggunakan sar untuk memantau penggunaan memori dapat menggunakan perintah "sar –r (*interval*) (*count*)". Perintah interval digunakan untuk mengatur setiap berapa detik alat ini akan menangkap penggunaan memori dan menampilkannya. Perintah *count* digunakan untuk mengatur berapa kali alat ini akan memperlihatkan penggunaan memori dari komputer. Salah satu kolom pada hasil keluaran sar akan menampilkan persentase penggunaan memori. Lalu semua hasil akan dihitung rata-ratanya. Hasil penghitungan rata-rata tersebut dapat digunakan untuk menentukan berapa persen memori yang digunakan pada saat menjalankan suatu program.