PENGEMBANGAN PENGUJIAN KODE PROGRAM OTOMATIS PADA PERKULIAHAN ALGORITMA DAN PEMROGRAMAN

Putra Prima Arhandi [1], Mungki Astiningrum[2], Moch Ma'ruf Amien[3]

[1,2,3] Teknik Informatika, Teknologi Informasi, Politeknik Negeri Malang
[1] putraprima@polinema.ac.id, [2]mungki.astingrum@polinema.ac.id, [3]artdeffend@gmail.com

Abstrak

Pendidikan pada bidang teknologi informasi sangat erat kaitannya dengan kemampuan mahasiswa dalam membuat sebuah kode program. Banyak cara yang dilakukan oleh lembaga pendidikan untuk meningkatkan kemampuan programming mahasiswa. Salah satunya adalah metode *automated assessment* dimana pada metode ini digunakan dengan bantuan komputer untuk melakukan penilaian secara otomatis. Banyak penelitian dilakukan untuk menguji kode program secara otomatis, salah satu nya Penelitian Arhandi pada tahun 2017 yang sudah mampu menyediakan *service* dengan performa yang baik dari sebelumnya tetapi perlu dilakukan perbaikan karena *online judge* yang dibuat hanya mampu untuk menguji benar atau salah dari kode program yang dikerjakan oleh siswa menggunakan bahasa pemograman C/C++ dan untuk mengakses *service online judge* tersebut pengguna harus menggunakan aplikasi *REST Service* seperti postman. Oleh karena itu pada penelitian ini dikembangkan sebuah pengujian kode program otomatis (*online judge*) yang bisa menangani pengujian dari kode program yang di *submit* mahasiswa yang menggunakan bahasa pemograman Java dan menyediakan sistem serta *frontend* (antarmuka) yang dapat mempermudah mahasiswa maupun dosen dalam mengakses *service* pengujian kode program otomatis. Dari hasil pengujian performa menggunakan aplikasi jMeter didapatkan bahwa sistem dapat menguji kode program sebanyak 120 mahasiswa dengan total 3600 *request* dengan rata-rata latency 22.308 ms.

Kata kunci : Pengujian Kode Program, Java, C/C++

1. Pendahuluan

Pendidikan pada bidang teknologi informasi sangat erat kaitannya dengan kemampuan mahasiswa dalam membuat sebuah kode program. Banyak cara yang dilakukan oleh lembaga pendidikan untuk meningkatkan kemampuan programming mahasiswa. Salah satunya adalah metode automated assessment dimana pada metode ini digunakan dengan bantuan komputer untuk melakukan penilaian secara otomatis [1]. Pada umumnya perkuliahan algoritma dan pemograman tidak hanya mengajarkan tentang teori saja, namun juga terdapat praktikum yang menggunakan bahasa pemograman. Sehingga perlu adanya pengujian kemampuan mahasiswa dalam membuat kode program pada praktikum ini.

Pada tahun 2001, Kurnia dalam penelitiannya membuat suatu automated assessment untuk masalah pengujian pemograman otomatis dimana siswa diberikan permasalahan dan menyelesaikannya dalam bentuk kode program, penelitian tersebut telah berhasil membuat online judge yang dapat memberikan penilaian otomatis terhadap kode program yang di submit oleh siswa. Penelitian ini berhasil mengurangi penilaian secara subjektif dan mengurangi keterlibatan manusia untuk menguji kode program yang di submit oleh siswa [2]. Kekurangan dalam penelitian ini adalah apabila diperlukan suatu perubahan baik pada interface atau antarmuka pengguna yang menyangkut pada sistem, atau pada sistem penilaian itu sendiri, maupun cara menguji kode program, maka struktur yang memiliki ketergantungan akan sangat sulit untuk disesuaikan sehingga banyak yang harus diperbaiki.

Penelitian selanjutnya dilakukan oleh Sunaryono pada tahun 2012, yaitu membuat pengujian kode program otomatis berbasis web service dengan menggunakan SOAP (Simple Object Access Protocol), dimana dilakukan pemisahan sistem antara sistem untuk *backend* sebagai service untuk pemrosesan dan penilaian dengan *frontend* sebagai antarmuka pengguna, sehingga dengan adanya web service ini dapat mengurangi ketergantungan sistem dan dapat mempermudah pengembangan aplikasi selanjutnya tanpa proses integrasi yang rumit [3]. Kekurangan dari penelitian ini adalah semakin banyak *user* yang melakukan *request* maka akan terjadi penurunan performa, hal ini bisa dilihat dari hasil pengujian *performance* di penelitian tersebut, dimana terdapat 100 *request* membutuhkan 26 detik waktu proses. Untuk memperbaiki *performance* dari online judge ini maka Arhandi dalam penelitian nya pada tahun 2017 berhasil mengembangkan sebuah sistem online judge yang dapat menangani *user request* yang besar secara bersamaan menggunakan nodejs dan mongodb dengan framework loopback. Dari hasil

pengujian sistem dengan 30 *user* melalukan 10 *request* per second selama 20 detik membutuhkan rata-rata 14,6 ms latency dengan jumlah *request* yang sukses 4324 dari 6000 *request*, yang berarti sekitar 72,0% *request* yang sukses [4].

Meskipun dalam penelitian Arhandi sudah mampu menyediakan service dengan performa yang baik dari sebelumnya. Perlu dilakukan perbaikan karena online judge yang dibuat hanya mampu untuk menguji benar atau salah dari kode program yang dikerjakan oleh siswa menggunakan bahasa pemograman C/C++ dan untuk mengakses service online judge pengguna harus menggunakan aplikasi REST Service seperti postman. Oleh karena itu pada penelitian ini dikembangkan sebuah pengujian kode program otomatis (online judge) yang bisa menangani pengujian dari kode program yang di submit mahasiswa yang menggunakan bahasa pemograman Java dan menyediakan sistem serta frontend (antarmuka) yang dapat mempermudah mahasiswa maupun dosen dalam mengakses service pengujian kode program otomatis.

Sistem yang akan dibuat menggunakan nodejs, karena mempunyai fitur non-blocking asyncronoush execution [5], dalam fitur non-blocking ini, setiap *request* akan dilaksakan langsung tanpa harus menunggu *request* yang lain selesai dilaksanakan. Dalam sistem ini juga menggunakan database mongodb, karena mongodb memilik peforma yang lebih baik dalam menangi datashet yang besar dari pada mysql[6]. Diharapkan dengan adanya pengembangan ini, sistem memiliki performa yang tinggi dalam menangani *request* yang besar, serta dapat mempermudah pengguna dalam mengakses service dan tidak hanya dapat menguji kode program dengan menggunakan bahasa pemograman C/C++ tetapi juga bahasa pemograman Java.

2. Tinjauan Pustaka

2.1 *Automatic Blackbox Testing*

Black Box Testing berfokus pada spesifikasi fungsional dari perangkat lunak. Tester dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada spesifikasi fungsional program. Black box testing memfokuskan kepada perilaku perangkat lunak berdasarkan kebutuhan fungsional (functional requirement) dari perangkat lunak yang bersangkutan. Dalam hal ini, *testcase* yang dipergunakan adalah *testcase* yang merepresentasikan kebutuhan fungsional perangkat lunak, terlepas dari jalannya perangkat lunak yang diujikan. Perlu diperhatikan bahwa penggunaan black box testing umumnya hanya sedikit membutuhkan akses ke sourcecode. Hal ini sesuai dengan karakteristiknya yang tidak memperhatikan bagaimana jalannya suatu aplikasi, namun lebih memperhatikan *input* dan *output* dari sistem yang dibuat [7].

Automatic BlackBox Testing merupakan teknik untuk menguji sebuah sorcecode secara otomatis berdasarkan *testcase* yang diberikan, jika hasil dari pengujian tersebut sama dengan answer case dari sourcecode tersebut, maka sourcecode tersebut dianggap benar. Di dalam penelitian ini metode pengujian sourcecode menggunakan Automatic BlackBox Testing yang digunakan untuk menguji sourcecode yang di submit mahasiswa, bahasa pemograman yang akan di uji adalah Java dan C/C++.

2.2 *Web Service*

Web service merupakan sebuah mekanisme interaksi antar sistem yang bertujuan untuk suatu kepentingan integrasi data yang dapat diakses melalui internet oleh banyak pihak. Selain itu, web service dapat diimplementasikan dengan menggunakan platform apapun dan dapat dibangun dengan menggunakan bahasa pemrograman apa saja. [8]

2.3 NodeJs

Node.js merupakan salah satu platform pengembang yang dapat digunakan untuk membuat aplikasi berbasis cloud. Node.js dikembangkan dari engine javascript yang dibuat oleh google untuk browser chrome ditambah dengan libuv serta beberapa pustaka lainnya. Node.js menggunakan javascript sebagai bahasa pemrograman dan event-driven, non-blocking I/O (*asynchronous*) model yang membuatnya ringan dan efisien. Node.js memiliki fitur built-in HTTP server library yang menjadikannya mampu menjadi sebuah web server tanpa bantuan software lainnya seperti Apache dan Nginx [5].

2.4 MongoDB

MongoDB adalah sebuah document oriented database yang bersifat open source. MongoDB merupakan salah satu database NoSQL yang memiliki sebuah konsep penyimpanan data non-relational. Istilah NoSQL merupakan kepanjangan dari “Not Only SQL” yaitu sistem manajemen database yang berbeda dari sistem manajemen database rasional dalam beberapa cara. Penyimpanan data tanpa perlu adanya tabel skema dan tidak ada bahasa sql yang terlibat dalam pemakaian database. MongoDB dikembangkan sejak Oktober 2007 oleh 10Gen dan dirilis ke public sejak februari 2009 yang mempunyai lisensi GNU AGPL 3.0 dan Apache License untuk driver nya [5].

Dalam MongoDB tidak mengenal adanya tabel, kolom dan baris jadi tidak ada skema dalam MongoDB (schema-less). Unit paling kecil pada dari MongoDB adalah dokumen, sedangkan kumpulan dari dokumen adalah collection. Seperti halnya

dalam database rasional document, ibarat sebuah record dan collection pada sebuah tabel dokumen dalam MongoDB dapat memiliki atribut yang berbeda dengan dokumen yang lainnya walaupun pada satu collection. MongoDB mempunyai peforma yang lebih cepat dibandingkan dengan Mysql [6].

2.5 Angular

Angular adalah sebuah *framework* yang dikembangkan secara *open-source* oleh Google, komunitas programmer dan perusahaan yang berkepentingan untuk membuat aplikasi yang berjalan dalam satu halaman saja. Angular juga punya keunggulan yaitu dapat dikembangkan di web, mobile dan desktop

Angular dikembangkan sejak tahun 2009 oleh Misko Hevery dan Adam Abrons pada saat bekerja di Brat Tech LLC. Awalnya, Angular menjadi salah satu *software* dibalik layanan *online storage* JSON yang dikembangkan. Layanan *storage* JSON ini disasarkan pada pasar enterprise namun belum mampu menyedot perhatian cukup banyak pengguna. Pada awalnya Angular diperkenalkan di *domain* "GetAngular.com". Karena kurang ramai, akhirnya *library* ini pun dirilis secara *opensource*. Setelah menjadi *library opensource*, Angular kemudian terus dikembangkan oleh Hevery pada saat dia bekerja di Google. Hingga saat ini angularjs diganti dengan angular-cli atau angular versi 2, dimana perintah instalasi dilakukan di terminal ataupun di *command prompt*. Dalam penenelitian ini Angular digunakan untuk membuat *front end* yang nantinya akan diakses oleh mahasiswa dan dosen, Angular yang dipakai adalah Angular JS versi 2. [9]

2.6 Apache Jmeter

Apache Jmeter adalah aplikasi pengujian atau testing guna mengetahui dan mengukur kemampuan aplikasi yang dibuat dalam menangani suatu kondisi yang tidak normal dari sisi volume ataupun kuantitas. Misalkan untuk mengetahui dan mengukur kekuatan sebuah website dalam menangani pengunjung dalam satu waktu secara bersamaan. Secara umum Apache Jmeter adalah sebuah tools yang memiliki fungsi sebagai berikut :

1. Sebuah Tool atau alat yang digunakan untuk melakukan performace test pada sebuah software.
2. Apache JMeter dapat memberikan *request* dalam jumlah yang sangat banyak secara bersamaan dalam satu waktu pada server
3. Apache JMeter dapat memberikan Analisa dan laporan dari hasil pengujian

2.7 Async

Async adalah fitur yang hadir sejak ES2017. Fitur ini mempermudah programmer dalam menangani proses asynchronous. Nodejs dikenal sebagai bahasa yang single-threaded, non-blocking. akibat menggunakan non-blocking adalah beberapa operasi yang ada di nodejs merupakan operasi yang asynchronous. Callback function adalah salah satu metode yang paling umum yang digunakan untuk menghandle return value dari operasi asynchoronous [8].

Callback sendiri adalah sebuah regular function dan ditaruh di argumen paling belakang dari sebuah asynchronous function. Layaknya function biasa, callback juga dapat menerima parameter dan mengembalikan nilai. Penggunaan callback ini memiliki kelemahan yaitu nilai kembalian dari callback hanya dapat dipanggil di dalam callback function tersebut, hal ini membuat para developer untuk menulis kode dengan struktur nested sehingga menyebabkan callback hell. Oleh karena itu dalam penelitian ini nantinya menggunakan async.waterfall yang digunakan untuk menajalankan function secara berurutan, serta menggunakan async.map yang digunakan untuk menguji *input* dan *output* yang berjalan bersama-sama.

3. Metodologi

3.1 Metode Pengambilan Data

Tahapan penelitian dimulai dengan melakukan pengumpulan data, pada tahap ini dilakukan pengumpulan data berupa contoh soal dan jawaban untuk soal pemrograman dengan menggunakan Bahasa pemograman C/C++ dan Java yang sesuai dengan materi di mata kuliah Algoritma dan Pemograman.

Setelah data soal dikumpulkan selanjutnya data soal ini dibuatkan kunci jawaban, *test case* dan *answer case* nya, kunci jawaban berupa sebuah kode program yang menjawab pertanyaaan dari soal yang diberikan. Sedangkan *test case* merupakan sample *input* dari kunci jawaban, *test case* yang dibuat diberikan sebanyak mungkin untuk menghindari kecurangan. Untuk penelitian ini *test case* yang di buatkan minimal 10 *test case* untuk masing masing soal.

3.2 Metode Pengolahan Data

Setelah data soal, kunci jawaban, dan *test case* berhasil dibuat proses selanjutnya adalah pembuatan service online judge menggunakan nodejs, express dan mongodb. Service dibuat dengan memanfaatkan RESTfull api dari nodejs dan expressjs, dengan membuatkan sebuah url yang dapat diakses melalui RESTfull servie yang dapat digunakan oleh aplikasi client untuk menguplod file / kode program yang kumpulkan oleh mahasiswa, setelah itu jawaban jawaban yang dikumpulkan ini akan dicompile secara otomatis oleh server online judge yang hasil kompilasi ini akan dibandingkan dengan *test case* yang ada.

Proses selanjutnya dengan menggunakan data dari hasil *test case* ini sistem online judge dapat menyimpulkan apakah kode program jawaban dari siswa dapat dianggap benar atau salah. Setelah didapakan nilai nya server online judge mengembalikan respon kepada aplikasi yang meminta RESTfull service berupa nilai dari kode program yang dikumpulkan.

3.3 Metode Pengujian

Untuk menguji keberhasilan sistem dapat dilakukan dengan beberapa cara, pada penelitian ini menggunakan pengujian berupa stress test dengan mengirimkan banyak *request* ke RESTfull service pada server, sehingga dapat menguji keandalan dan kekuatan sistem yang dibuat.

Skenario pengujian dilakukan dengan membuatkan sebuah client sederhana untuk mengirim file test ke server online judge, server ini dibuat dengan menggunakan node js yang memanfaatkan teknologi non blocking sehingga dapat mengirim lebih dari satu test file ke server dalam waktu yang bersamaan, untuk masing masing file akan di ukur dan di hitung response time yang dibutuhkan. Ini dilakukan untuk menguji keandalan server online judge dalam menilai dan mengkoreksi test file yang dikirim.

4. Perancangan

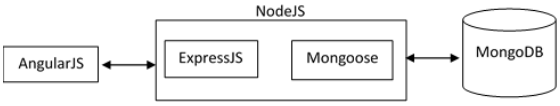
4.1 Deskripsi Sistem

Online judge merupakan sebuah sistem yang digunakan untuk melakukan penliaian otomatis terhadap tugas pemrograman. Online judge memanfaatkan kemampuan mesin untuk mengevaluasi hasil dari sebuah kode program dan membandingkannya dengan template / hasil yang diharapkan. Dengan cara ini online judge yang dibuat dapat memberikan penilaian secara independen tanpa terpengaruh oleh faktor faktor subjektif [2].

Kelebihan dengan menggunakan online judge seorang mahasiswa dapat melaporkan tugasnya kapan saja tanpa harus menunggu seorang dosen memberikan feedback karena hasil dan nilai dari evaluasi program otomatis dihitung oleh mesin online judge. Kelebihan selanjutnya online judge dapat mengurangi beban kerja seorang dosen karena proses pemeriksaan dilakukan otomatis dan tidak tergantung kepada dosen untuk memeriksa kode program yang dilaporkan mahasiswa.

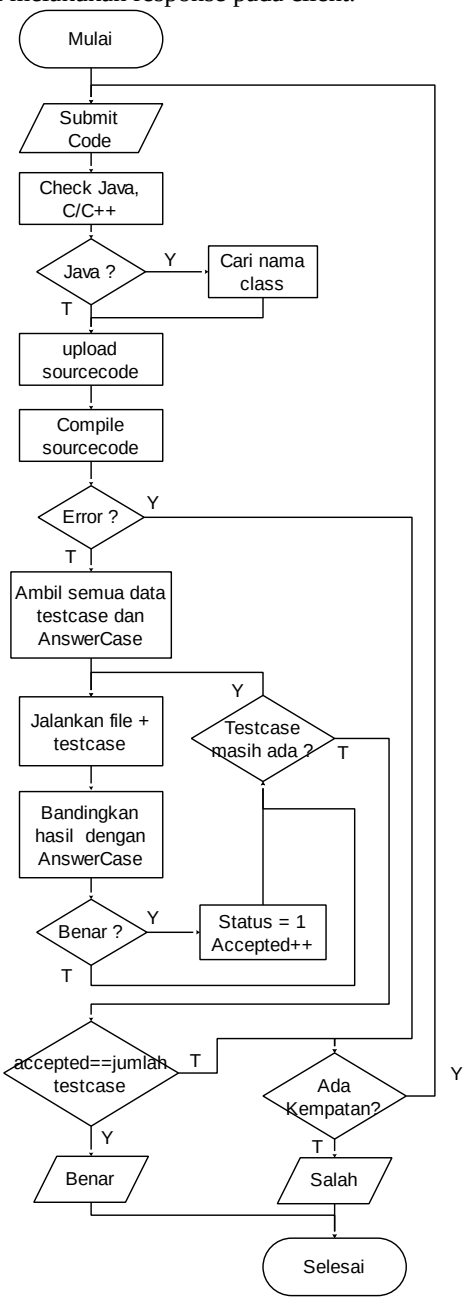
4.2 Analisis Sistem

Dalam penelitian ini akan dilakukan analisa terhadap kebutuhan sistem. Untuk lebih detail nya bisa dilihat pada Gambar 1 dibawah ini :



Gambar 1. Arsitektur Sistem

Pada Gambar 1 menjelaskan bagaimana arsitektur yang digunakan untuk membangun aplikasi yang terdiri dari server (*back end*) dan client (*front end*). Pada sisi server menggunakan NodeJS yang didalamnya terdapat framework ExpressJS dan mongoose agar terkoneksi dengan MongoDB. Pada sisi client menggunakan javascript (AngularJS). Server dan client menjalin komunikasi melalui HTTP Method (get, post, put, delete) dengan melakukan *request* pada client, selanjutnya server akan melakukan response pada client.



Gambar 2. Flowchart Pengujian kode program C/C++ dan Java

Alur sistem untuk menguji kode program C/C++ dan Java pertama kali mahasiswa mengerjakan kode program sesuai dengan soal yang diberikan, setelah selesai mahasiswa bisa men-submit nya, sistem akan mengecek Bahasa pemograman yang dipakai, jika Bahasa pemograman yang dipakai adalah java, maka sistem akan mencari nama class dari kode program tersebut. Setelah itu sistem akan mengirim sourcecode ke dalam server. Server akan meletakkan nya pada folder uploads/ mahasiswa/ :idmahasiswa /:idquestion/:idresult dan path ini disimpan kedalam database, file ini harus di upload karena dibutuhkan untuk membandingkan hasil program yang di buat oleh mahasiswa dengan answercase yang disediakan oleh sistem. Setelah jawaban mahasiswa berhasil di upload, sistem akan mengkompilasi program yang sudah di upload menggunakan perintah gcc untuk C/C++ dan perintah javac mengkompilasi Java, pada nodejs perintah gcc dan javac dapat dipanggil dengan menggunakan library childprocess. Setelah itu jika kode program yang di kirim oleh mahasiswa dapat dikompilasi maka dilanjutkan ke proses selanjutnya jika tidak maka proses berhenti dan response message error dikirim ke mahasiswa dan mengecek apakah masih ada kesempatan. jika benar sistem akan mencari daftar file *testcase* dan answercase yang sebelumnya sudah di siapkan untuk soal yang dikerjakan oleh mahasiswa. Setelah itu kode program yang berhasil, dilakukan testing terhadap executable dari kode program. Untuk menguji nya executable hasil kompilasi program di berikan *input* dari semua *test case* yang sudah disediakan dan kompilasi kode program akan dijalankan secara bersamaan, jika *output* dari executable ini sama dengan answer case maka kode program dianggap lolos uji test dan variable status menjadi 1, variable accepted akan bertambah, hal ini dilakukan terus menerus sampai semua kode program selesai di kompilasi. Jika accepted sama dengan jumlah *testcase* maka sistem akan me response ke mahasiswa bahwa jawabannya sudah benar jika accepted kurang dari *testcase* maka diangap jawaban dari mahasiswa tersebut salah. Jika kode program dianggap salah maka jika ada kesempatan untuk memperbaiki, mahasiswa bisa mengerjakan kode program dan men submit nya lagi sampai jumlah kesempatan mengerjakan soal yang diberikan oleh dosen.

5. Implementasi

Implementasi adalah penulisan bahasa pemograman pada komputer, agar kode program dapat berjalan sesuai dengan rancangan. Berikut langkah langkah pada tahap implementasi.

5.1 Instalasi

Instalasi dimulai dari memasang nodejs ke dalam operating sistem, selanjutnya memasang mongodb sebagai tempat penyimpanan database, dan yang terahir angularjs sebagai antarmuka *user*.

5.2 Membuat *Schema*

Pada tahapan ini dimulai dengan mendesain schema database yang merepresentasikan bentuk table pada database, database yang digunakan pada penelitian ini adalah database mongodb. Pada database ini dibuat beberapa collection dan disesuaikan dengan perancangan class diagram.

5.3 Membuat *Endpoint*

Proses implementasi selanjutnya yaitu membuat end point yang nanti nya akan diakses oleh frontend. Endpoint yang dibuat menggunakan library nodejs yaitu expressjs.

5.4 Membuat Antarmuka

Proses yang terakhir yaitu membuat tampilan yang akan diakses oleh mahasiswa dan dosen dan terintegrasi dengan endpoint yang sebelumnya telah dibuat.

6. Pengujian

Melakukan testing pada sistem yang sudah dibuat. Tahap pengujian diperlukan sebagai ukuran bahwa sistem dapat dijalan kan sesuai tujuan atau belum. Ada beberapa bentuk pengujian dalam penelitian ini, yaitu :

6.1 Pengujian Fungsional

Pengujian Fungsional digunakan melihat sistem sudah berjalan atau belum. Skenario pengujian dilakukan dengan menguji fitur-fitur dari aplikasi yang dirancang, mulai dari dosen login, membuat soal, jawaban sampai mahasiswa dapat mensubmit kode program dan keluar hasilnya.

6.2 Pengujian Performa

Pengujian performansi digunakan untuk melihat kemampuan sistem dalam menangani jumlah variasi *request* oleh *user* per second. Dalam pengujian ini dilakukan konfigurasi *user* sebanyak 30, 60, 90, 120 *user* untuk melakukan *request*, jumlah ini disesuaikan dengan rata-rata jumlah mahasiswa perkelasnya. Untuk melakukan pengujian sistem digunakan sebuah aplikasi jMeter, aplikasi ini dapat membuat *request* otomatis terhadap endpoint yang dapat di atur jumlah *user*.

Untuk menguji *endpoint submit* kode program berbeda dengan pengujian sebelumnya, kita atur ada 30, 60, 90, 120 mahasiswa yang mengerjakan 10 soal masing-masing soal terdapat 3 kali kesempatan dengan lama waktu 1 jam, jika jumlah mahasiswa 30 dan menggunakan seluruh kesempatan dalam waktu 1 jam maka jumlah *request* sama dengan 900 dalam waktu 1 jam. Berikut ini hasil dari pengujian submit kode program.

Tabel 1. Hasil pengujian

No	Nama *endpoint*	Jumlah *User*	Respon sukses	Avg *latency*
1	/resulttest	30	900	1.686 ms
2	/resulttest	60	1800	1.641 ms
3	/resulttest	90	2700	17.261 ms
4	/resulttest	120	3600	22.308 ms

Hasil pengujian ini menunjukkan dengan jumlah 29 mahasiswa mengerjakan 10 soal 3 kali kesempatan dalam waktu 1 jam, terdapat total submit kode 317 *request* dengan rata-rata 1.579 ms latency.

Sehingga dari hasil pengujian menggunakan aplikasi jmeter dengan pengujian yang dilakukan dikelas menunjukkan hasil yang hampir sama, ini dibuktikan dengan hasil pengujian endpoint submit kode program, dimana ketika menggunakan aplikasi jmeter dengan jumlah *user* 30 membutuhkan rata-rata waktu 1.648 ms, sedangkan hasil pengujian kelas dimana terdapat 29 mahasiswa yang mengerjakan kodeprogram membutuhkan waktu 1.579 ms.

Pada penelitian sebelumnya menggunakan framework loopback yang dimana dapat meresponse 4324 dari 6000 *request* atau sekitar 72.0% dapat merespons sukses, sedangkan pada penelitian ini menggunakan framework expressjs yang dapat meresponse 100% dari 6300 *request*.

7. Kesimpulan dan Saran

Berdasarkan hasil analisis, perancangan, implementasi dan pengujian yang dilakukan, terdapat beberapa kesimpulan yang didapat yaitu :

1. Sistem yang dibuat sudah mampu melakukan penilaian benar atau salah untuk jawaban kode program yang disubmit mahasiswa yang menggunakan bahasa C/C++ dan Java.
2. Sistem yang dibuat sudah ada antarmuka yang dibangun menggunakan angularjs yang nantinya diakses oleh mahasiswa maupun dosen.
3. Sistem yang dibuat mampu menangani *request* dari *user* dan memberikan respon yang benar sampai dengan 120 *user* yang melakukan 10 *request* per second ke service pengujian kode program ini.

Berdasarkan penelitian yang dilakukan, dapat diajukan beberapa saran sebagai berikut :

1. Pemanfaat resource API dapat dikembangan lagi kedalam desktop app ataupun mobile app.
2. Untuk menguji kode program, masukan tidak hanya melalui parameter main kode program, tetapi juga bisa menggunakan standard I/O yang digunakan oleh bahasa pemograman java atau C/C++.

Daftar Pustaka:

Aghi, Rajat dkk. 2015. "A comprehensive comparison of SQL and MongoDB databases". Department of Computer Science, Maharaja Surajmal Institute of Technology, Janakpuri, N.delhi 110058, India.

Arhandi, Putra Prima dkk. 2017. "Pengembangan Service Online Judge Untuk Matakuliah Algoritma Berbasis Rest Menggunakan Nodejs Dan Mongodb". Jurusan Teknologi Informasi Politeknik Negeri Malang.

Haviv, Amos Q. 2016. "MEAN Web Development Second Edition". Published by Packt Publishing Ltd. Livery Place 35 Livery Street Birmingham B3 2PB, UK.

Kurnia, Andy dkk. 2001. "Online Judge". Department of Computer Science, National University of Singapore.

Markey, Philip dkk. 2013. "A *performance* analysis of WS-* (SOAP) and RESTful Web Services for Implementing Service and Resource Orientated Architectures". The 12th Information Technology and Telecommunications (IT&T) Conference, Athlone IT.

Mustaqbal, M. Sidi dkk. 2016. "Pengujian Aplikasi Menggunakan Black Box Testing Boundary Value Analysis (Studi Kasus : Aplikasi Prediksi Kelulusan Snmptn)". Jurusan Teknik Informatika, Fakultas Teknik, Universitas Widyatama, Bandung.

Nurseitov, Nurzhan dkk. 2009. "Comparison of JSON and XML Data Interchange Formats: A Case Study". Department of Computer Science Montana State University Bozeman.

Ramanathan, Ramakrishnan. 2014. "Software Service Architecture to Access Weather Data Using RESTful Web Services". Department of Electrical Engineering and Computer Science, University of Applied Sciences, 32657 Lemgo, Germany.

Satheesh, Mithun dkk. 2015. "Web Development with MongoDB and NodeJS Second Edition". Published by Packt Publishing Ltd. Livery Place 35 Livery Street Birmingham B3 2PB, UK.

Seppala, Otto. 2012. "Advances in Assessment of Programming Skills". Department of Computer Science and Engineering, Aalto University.

Sunaryono, Dwi dkk. 2012. "Penjurian Online Berbasis Web Service". Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya.

Wibowo, Wahyudi. (2017, July 11). "Bagaimana Javascript Menghandle Proses Asynchronous - Callback, Promise, Coroutine, dan Async/Await". Available : https://medium-.com/koding-kalaweekend/bagaimana-javascr-iptmenghandle-proses-asynchronous-callback-promise-coroutine-dan-async-await-928326575289.

Zhao, Gansen dkk. 2013. "Modeling MongoDB with Relational Model". School of Computer Science, South China Normal University, Guangzhou, China.