Perancangan dan Analisis *Virtual Machine* Untuk Fitur *Chatbot* Pada Aplikasi Muslim.In dengan Menggunakan Google Cloud Platform

1st Euis Amara Putri
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
euisamaraputri@student.telkomuniversity.ac.id

2nd Suryo Adhi Wibowo
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
suryoadhiwibowo@telkomuniversity.ac.id

3rd Thomhert Suprapto Siadari
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
thomhert@telkomuniversity.ac.id

***Abstrak* — Saat ini mahasiswa rentan terkena gangguan mental. Hal ini dikarenakan mereka tidak memiliki tempat untuk bercerita. Individu yang tidak memiliki hubungan sosial berada dalam resiko kesepian, yang tentunya menyebabkan perubahan perilaku. Maka dari itu, butuh sesuatu untuk menjadi tempat mereka bercerita. *Chatbot* dapat menjadi solusi untuk meringankan keadaan kesepian. Muslim.in adalah aplikasi serbaguna yang menawarkan berbagai fitur Islami. Muslim.in juga memiliki fitur utama yaitu *chatbot*. Fitur *chatbot* ini menjadi tempat untuk *user* bercerita terkait masalah yang dialami. Fitur *chatbot* dibuat dengan mengambil API OpenAI lalu difilter menggunakan model *deep learning*. Model *deep learning* dari aplikasi Muslim.in tidak disimpan dalam aplikasi karena takut memberatkan aplikasi. Model tersebut disimpan ke dalam Google Compute Engine (GCE) yang disediakan oleh Google Cloud Platfrom (GCP). GCE memungkinkan pengguna meluncurkan mesin virtual sendiri sesuai dengan kebutuhan. Dalam fitur *chatbot* ini, *user* akan menuliskan atau merekam suara untuk mengirimkan curhat. Teks tersebut akan dimasukkan ke dalam mesin virtual. Untuk mengetahui apakah mesin virtual dapat berjalan maka dilakukan pengujian menggunakan Postman. Pada pengujian tersebut API berhasil merespons dengan membalas jika itu curhat. Dan memberi tahu jika kalimat bukan termasuk curhat. Dengan demikian, mesin virtual sudah lancar digunakan untuk memproses dan menyimpan model untuk fitur *chatbot* di aplikasi Muslim.in**

***Kata kunci* — *chatbot*, GCE, GCP, VM.**

I. PENDAHULUAN

Pada era saat ini, sering terdengar bahwa mahasiswa rentan terkena gangguan mental. Bahkan di beberapa kasus ada yang sampai menghabiskan nyawanya sendiri. Hal ini dikarenakan mereka memendam sendiri masalah yang mereka hadapi. Dengan arti mereka kesepian, mereka tidak memiliki tempat untuk berkeluh kesah. Individu yang tidak memiliki hubungan sosial berada dalam resiko kesepian, yang tentunya menyebabkan perubahan perilaku [1]. Manusia membutuhkan tempat untuk memberikan keluh kesah ataupun masalah mereka sehari-hari. *Chatbot* dapat menjadi solusi untuk meringankan keadaan kesepian [2].

Fitur *chatbot* terdapat dalam aplikasi Muslim.in. Fitur ini dibuat dengan menggunakan metode *deep learning*. Cara agar Model *deep learning* dapat dijalankan di fitur *chatbot* pada aplikasi Muslim.in adalah dengan menyimpan model tersebut ke dalam Google Comute Engine (GCE) yang disediakan oleh Google Cloud Platfrom (GCP). Penyimpanan model ke dalam penyimpanan *compute engine* bertujuan agar aplikasi muslim.in tidak memiliki ukuran yang besar.

GCE memungkinkan pengguna meluncurkan mesin virtual sendiri sesuai dengan kebutuhan. Mesin Virtual. Pembuatan *Virtual Machine* (VM) dilakukan pada GCP pada menu VM Instances. Pada VM ini dijalankan servis flask, nginx, dan gunicorn. Dalam fitur *chatbot* ini, *user* akan menuliskan atau merekam suara untuk mengirimkan curhat. Teks tersebut akan dimasukkan ke dalam VM.

Analisis pengujian VM dilakukan dengan menggunakan postman dengan metode post. API berhasil jika merespon balasan curhat jika termasuk curhat. Dan menyatakan kalimat bukan curhat jika kalimat bukan termasuk curhat. Dalam pengujian ini juga dapat mengetahui waktu API dalam merespons.

II. DASAR TEORI

Kajian teori berisi penjelasan tentang teori-teori yang berkaitan dengan variable-variabel penelitian. Teori yang dijelaskan terdiri dari beberapa bahasan. Pada bagian ini akan dijelaskan tentang google cloud platform, google compute engine, nginx, gunicorn, flask, dan postman.

A. Google Cloud Platform

Google Cloud Platform merupakan salah satu layanan *cloud computing* yang disediakan oleh Google. GCP memiliki banyak layanan *cloud computing* yang dapat digunakan untuk merancang infrastruktur *server* dengan tingkat keandalan dan ketersediaan yang tinggi, dimana setiap produk yang ada memiliki kemampuan dan keunggulan yang berbeda pada setiap fiturnya [3]. Adapun beberapa layanan produk GCP yang dapat digunakan untuk merancang bangun infrastruktur *server* tersebut adalah Compute Engine, Cloud SQL, Cloud Storage, dan Container Registry. Dengan menggunakan infrastruktur dari GCP, setiap orang dapat menciptakan infrastruktur yang bisa membuat pengembangan produk menjadi lebih cepat dan juga stabil.

B. Google Compute Engine

Google Compute Engine adalah layanan infrastruktur yang disediakan oleh Google Cloud Platform. Komponen inti dari Compute Engine adalah *virtual machine*. Compute Engine memungkinkan Anda untuk membuat kelompok VM berkinerja tinggi [4]. GCE membantu pengguna untuk

meluncurkan *virtual machine* sendiri sesuai dengan kebutuhan. GCE memungkinkan pengguna untuk memulai, menghentikan *instances*, menambahkan *disk storage*, dan mengkonfigurasi *network access*.

C. Flask

Flask adalah sebuah *web framework* yang tidak bisa lepas dari bahasa pemrograman python. Flask menjadi kerangka kerja dari python. Flask adalah *framework* yang masuk ke dalam kategori mikro, sehingga akan mengurangi ketergantungan pada *library* dan *extensi* pihak luar. *Framework* ini memungkinkan pengguna untuk membuat *core* yang sederhana namun tetap dapat dengan mudah untuk ditambahkan. Namun, meskipun sederhana, Flask tetap kuat dan dapat digunakan untuk membangun aplikasi web yang kompleks dan skala besar. Flask menyediakan alat yang cukup untuk memulai dengan cepat, namun tetap memberikan fleksibilitas untuk mengembangkan aplikasi web yang lebih kompleks.

D. Nginx

Nginx adalah *software open-source* yang bertindak sebagai *server* web untuk memaksimalkan kinerja dan stabilitas situs web [5], [6]. *Software* ini tidak hanya dapat digunakan sebagai layanan web, tetapi juga dapat melakukan *reverse proxy*, *caching*, *load balancing*, *streaming* media, dan lainnya . Nginx memiliki keunggulan di antaranya adalah sebagai *Load Balancer All-In-One*, *software* multifungsi, dan memiliki dokumentasi lengkap. Nginx sebagai *reverse proxy* bertindak sebagai penghubung antara *host* (*client*) dan *server* [6], [7]. Nginx juga digunakan untuk memuat seimbang lalu lintas web di antara beberapa *server backend*. Dalam konfigurasi ini, Nginx bertindak sebagai titik masuk yang menerima permintaan klien dan meneruskannya ke *server backend* yang sesuai. Ini memungkinkan distribusi beban yang merata dan peningkatan kinerja sistem secara keseluruhan.

E. Gunicorn

Gunicorn merupakan singkatan dari "*Green Unicorn*". Gunicorn adalah *server* HTTP untuk Python, berdasarkan Antarmuka Gerbang *Server* Web (WSGI), yang dirancang khusus untuk lingkungan mirip Unix [8]. Gunicorn digunakan bersama dengan web *server* seperti Nginx atau Apache. Nginx bertindak sebagai *reverse proxy* yang meneruskan permintaan klien ke Gunicorn, yang kemudian akan menangani permintaan tersebut dan mengirimkan respons kembali ke Nginx. Kombinasi Gunicorn dan Nginx sering digunakan dalam pengaturan produksi untuk meningkatkan kinerja dan keandalan aplikasi web Python [9], [10].

F. Postman

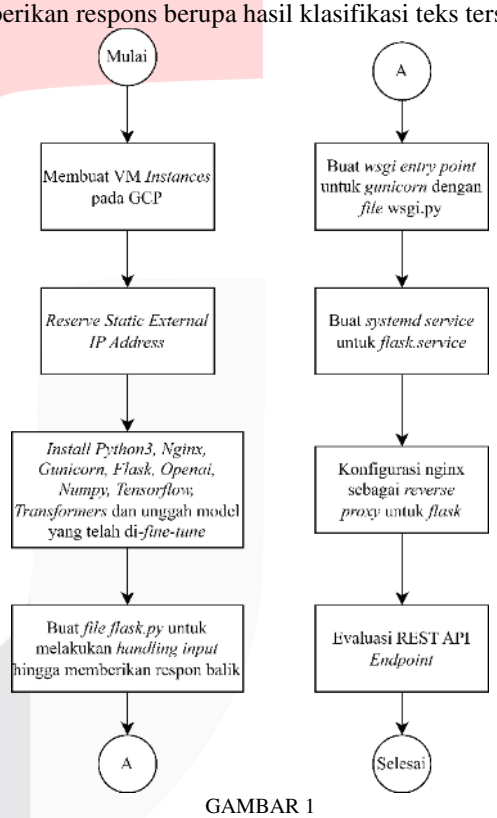
Postman adalah aplikasi komputer yang digunakan untuk pengujian API. Postman mengirimkan permintaan API ke *server* web dan menerima respons, apa pun itu. Tidak diperlukan konfigurasi kerja atau kerangka kerja tambahan saat mengirim dan menerima permintaan di Postman. Dengan antarmuka pengguna yang sederhana dan berbagai fitur yang kuat, Postman membantu mempercepat proses pengembangan dan memastikan kualitas dan kehandalan API yang dibangun. Ada beberapa fitur utama dari postman, fitur tersebut terdiri dari pengujian API, penyusunan permintaan, *environments* dan *variables*, kolaborasi tim, dan dokumentasi API. Selain fitur-fitur tersebut, Postman juga menyediakan banyak alat bantu lainnya, seperti pengelolaan koleksi permintaan, penjadwalan permintaan, pengujian otomatis, dan pemantauan API.

III. PERANCANGAN SISTEM

Perancangan sistem menjelaskan tentang bagaimana pembuatan sistem. Dalam perancangan sistem ini terdapat dua perancangan yang akan dijelaskan. Perancangan itu berupa perancangan VM dan perancangan pengujian dengan Postman.

A. Perancangan Virtual Machine

Virtual machine akan digunakan untuk melakukan *deployment* model *deep learning* agar dapat digunakan pada aplikasi tanpa membebani ukuran aplikasi. Layanan *Virtual Machine* yang akan digunakan harus dapat menerima masukan berupa HTTP POST dengan menyediakan REST API *endpoint* untuk menerima teks dari pengguna dan memberikan respons berupa hasil klasifikasi teks tersebut.



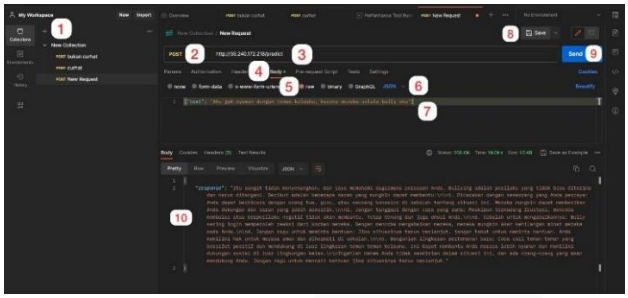
GAMBAR 1
Flowchart Perancangan *Cloud Vm*

Flowchart proses perancangan pada *cloud VM* dapat dilihat pada Gambar 1. Perancangan ini dilakukan untuk membuat servis API yang mampu melakukan prediksi kalimat curhat menggunakan model *deep learning* dan mengirimkan kalimat ke OpenAI API. Pembuatan VM dilakukan pada GCP pada menu VM Instances. Konfigurasi VM yang digunakan adalah VM e2-custom dengan 1 core vCPU dan 6 GB vMemory. Dari konfigurasi ini didapatkan IP *Address External* yang nantinya digunakan untuk menjalankan servis. Hal yang dilakukan selanjutnya adalah dengan meng-*install Python3m Nginx, Gunicorn, Flask, OpenAI, Numpy, Tensorflow, Transformers* dan unggah model yang telah di *fine-tune*. Nginx berguna sebagai *reverse proxy*. Gunicorn digunakan sebagai jembatan antara nginx dan flask atau berperan sebagai wsgi *server*. Flask digunakan

untuk menyimpan model. *Library* OpenAI berguna untuk tokenizer. Numpy, tensorflow, dan transformers digunakan untuk *deep learning*. Pada VM ini dijalankan servis flask, nginx, dan gunicorn. Sedangkan, flask digunakan untuk melakukan *handling* terhadap data yang dikirimkan ke API. Pada servis flask, akan dilakukan prediksi kalimat yang dimasukkan oleh pengguna. Hasilnya berupa "curhat" dan "bukan curhat". Apabila hasilnya "curhat", maka flask akan melakukan HTTP POST ke OpenAI API dan menerima respons berupa prompt hasil dari jawaban terhadap kalimat yang dikirimkan. Respons yang didapatkan dari OpenAI API akan dikirimkan kembali sebagai respons ke aplikasi.

B. Proses Pengujian dengan Postman.

Pengujian API dilakukan dengan metode POST menggunakan aplikasi postman. Metode POST berguna untuk mengirimkan data ke API. Proses pengujian API dengan postman dilakukan untuk mengetahui apakah API berhasil merespons apa yang dikatakan oleh *user*. Dalam pengujian ini juga dapat diketahu berapa lama waktu API untuk merespons. Langkah pengujian yang akan dijelaskan adalah pengujian menggunakan metode POST. Berikut langkah-langkah yang dilakukan dalam pengujian API *chatbot* setelah melakukan instalasi aplikasi Postman dan tertera pada Gambar 2:



GAMBAR 2
Langkah Pengujian *Postman*

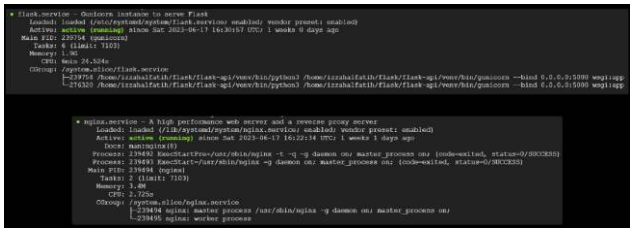
1) Membuat koleksi baru dengan memilih tombol "+".
2) Memilih metode POST.
3) Mengisi *Uniform Resource Locators* (URL) target yaitu http://35.240.172.218/predict.
4) Memilih menu *body*.
5) Memilih pilihan *raw*.
6) Mengubah tipe data menjadi JSON.
7) Mengisi data yang akan diuji berupa *dictionary* dengan *key*="text" dan *value* dengan data teks yang akan diuji.
8) Memilih tombol "*save*".
9) Memilih tombol "*send*" untuk memulai melakukan test.
10) Hasil akan ditampilkan di bagian bawah aplikasi.

Dalam ini dilakukan dua kali langkah pengujian di atas. Kedua pengujian ini dilakukan berdasarkan kelas dari filter model *deep learning*. Pengujian yang pertama mengirimkan teks curhat, dan pengujian yang kedua mengirimkan teks bukan curhat.

IV. PENGUJIAN DAN ANALISIS

Pengujian dan analisis menjelaskan hasil dari pengujian dan analisis. Terdapat dua pengujian dan analisis yang akan dijelaskan. Pengujian itu terdiri dari hasil *virtual machine* dan hasil pengujian dari Postman.

A. Hasil *Virtual Machine*

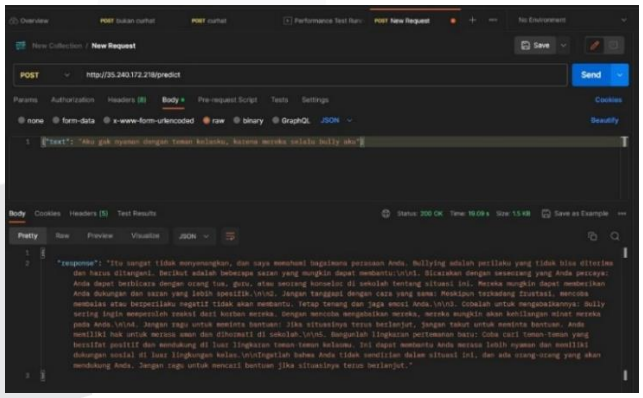


GAMBAR 3
Hasil *Virtual Machine*

Hasil dari perancangan VM dapat dilihat dari status servis flask dan servis nignx. Hasil dari langkah yang telah dilakukan dalam perancangan VM dapat terlihat pada Gambar 3. Bagian atas pada gambar memperlihatkan bahwa servis flask sudah berjalan. Hal itu terlihat dari bagian *active* pada flask.service – gunicorn instance to serve flask berwarna hijau dengan status *active* (*running*). Bagian bawah pada gambar memperlihatkan bahwa servis NGINX sudah berjalan. Hal itu terlihat dari bagian active pada nginx.service berwarna hijau dengan status *active* (*running*)

B. Hasil Pengujian Postman

Berikut adalah hasil yang didapatkan setelah melakukan dua kali langkah pengujian. Pengujian yang pertama dengan mengirimkan teks curhat lalu API merespons dengan mengirimkan teks balasan dari curhat yang telah dikirimkan. Hasil tersebut dapat dilihat pada Gambar 4. Pengujian yang kedua dengan mengirimkan teks bukan curhat, lalu API merespons dengan mengirimkan teks yang menyatakan bahwa teks yang dikirimkan bukan merupakan curhat. Hasil tersebut dapat dilihat pada Gambar 5.



GAMBAR 4
Hasil Pengujian Kalimat Curhat

GAMBAR 5
Hasil Pengujian Kalimat Bukan Curhat

Dari kedua hasil pengujian yang telah dilakukan, didapatkan hasil yang sesuai dengan yang diharapkan. Ketika mengirimkan teks curhat, API merespons dengan teks balasan dari curhat yang telah dikirimkan. Sedangkan ketika mengirimkan teks bukan curhat, API merespons dengan teks yang menyatakan bahwa teks yang dikirim bukan termasuk curhat. Dalam pengujian ini didapatkan pula waktu yang dibutuhkan API dalam memberikan respons. Waktu yang dibutuhkan untuk merespons curhat adalah 19,09 detik, sedangkan waktu yang dibutuhkan untuk merespons bukan curhat adalah 1,356 detik.

V. KESIMPULAN

Perancangan *virtual machine* dilakukan dengan menggunakan salah satu infrastruktur GCP yaitu GCE. *Virtual Machine* berhasil dirancang sehingga dapat digunakan pada fitur *chatbot* dalam aplikasi Muslim.in. Arsitektur dari *virtual machine* berhasil menerima *request* dan memberikan *response* yang tepat kepada pengguna aplikasi Muslim.in. Dalam hal ini juga didapatkan Rest API endpoint yang dapat digunakan untuk pengujian API dengan postman. Dari hasil pengujian postman didapatkan bahwa API berhasil merespon kalimat yang dikirimkan mau itu berbentuk curhat ataupun bukan curhat. Pada pengujian postman juga didapatkan waktu API dalam merespons, untuk kalimat curhat selama 19,09 detik dan bukan curhat selama 1,356 detik. Waktu respons tersebut termasuk ke dalam kategori baik karena tidak membuat pengguna fitur *chatbot* menunggu terlalu lama. Maka dari itu, sebagai penyedia layanan komputasi awan, GCP unggul dalam mendukung aplikasi sistem.

REFERENSI

[1] C. M. Masi, H. Y. Chen, L. C. Hawkley, dan J. T. Cacioppo, “A meta-analysis of interventions to reduce loneliness,” *Personality and Social Psychology Review*, vol. 15, no. 3. SAGE Publications Inc., hlm. 219–266, 2011. doi: 10.1177/1088868310377394.

[2] S. Valtolina dan L. Hu, “Charlie: A chatbot to improve the elderly quality of life and to make them more active to fight their sense of loneliness,” dalam *ACM International Conference Proceeding Series*, Association for Computing Machinery, Jul 2021. doi: 10.1145/3464385.3464726.

[3] N. Ramsari dan A. Ginanjar, “Implementasi Infrastruktur server berbasis cloud computing untuk web service berbasis teknologi google cloud platform,” dalam *Conf. Senat. STT Adisutjipto Yogyakarta*, 2022.

[4] S. P. T. Krishnan dan J. L. U. Gonzalez, “Google Compute Engine,” dalam *Building Your Next Big Thing with Google Cloud Platform: A Guide for Developers and Enterprise Architects*, S. P. T. Krishnan dan J. L. U. Gonzalez, Ed., Berkeley, CA: Apress, 2015, hlm. 53–81. doi: 10.1007/978-1-4842-1004-8_4.

[5] W. Reese, “Nginx: the high-performance web server and reverse proxy,” *Linux Journal*, vol. 2008, no. 173, hlm. 2, 2008.

[6] R. Soni, *Nginx*. Springer, 2016.

[7] X. Chi, B. Liu, Q. Niu, dan Q. Wu, “Web Load Balance and Cache Optimization Design Based Nginx under High-Concurrency Environment,” dalam *2012 Third International Conference on Digital Manufacturing & Automation*, IEEE, Jul 2012, hlm. 1029–1032. doi: 10.1109/ICDMA.2012.241.

[8] B. Chesneau, “Gunicorn - WSGI server,” 2023. https://docs.gunicorn.org/en/stable/ (diakses 1 Juli 2023).

[9] H. Hojaiji, O. Goldstein, C. E. King, M. Sarrafzadeh, dan M. Jerrett, “Design and calibration of a wearable and wireless research grade air quality monitoring system for real-time data collection,” dalam *2017 IEEE Global Humanitarian Technology Conference (GHTC)*, IEEE, Okt 2017, hlm. 1–10. doi: 10.1109/GHTC.2017.8239308.

[10] V. Patel, S. Kanani, T. Pathak, P. Patel, M. I. Ali, dan J. Breslin, “A Demonstration of Smart Doorbell Design Using Federated Deep Learning,” Okt 2020.