Physical Sciences, Life Science and Engineering Volume: 1, Nomor 2, 2024, *Hal*: 1-13

Arsitektur Perangkat Lunak Berbasis Layanan Mikro pada Sistem Manajemen Informasi Kantin

Mukhammad Dayu Anwar[1], Irwan A. Kautsar[2]

[1,2] Program Studi Teknik Informatika, Universitas Muhammadiyah Sidoarjo, Indonesia; irwan@umsida.ac.id[1]

Abstrak: Microservices menjadi salah satu cara untuk memaksimalkan performa dari service data yang akan digunakan untuk membuat sistem. Selain itu mudah untuk dikembangkan, karena setiap service nya sudah terstruktur dan terpisah untuk data yang akan dikonsumsi oleh developer lain. Kasus yang digunakan dalam penelitian ini adalah aplikasi E-Kantin, yang dimana sirkulasi data dari mitra, customer dan transaksi yang akan dikonsumsi oleh user harus berjalan dengan lancar. Dengan diimplementasikan nya microservices pada Aplikasi E-Kantin ini, maka sirkulasi data, kecepatan data yang dihasilkan akan sangat membantu untuk memaksimalkan setiap service yang dipakai. Hasil dari pengujian microservices yang telah dibuat, menunjukkan bahwa sirkulasi data dan service yang telah digunakan menjadi sangat efisien dan stabil pada saat banyaknya request atau akses dari user pemakainya. Dengan hasil tersebut, maka pengaruh dari microservices untuk sirkulasi data yang digunakan menjadi sangat efisien dan stabil.

Katakunci: Microservices, E-Kantin, API

DOI: https://doi.org/10.47134/pslse.v1i2.196
*Correspondensi: Irwan A. Kautsar
Email: irwan@umsida.ac.id

Received: 03-01-2024
Accepted: 12-02-2024
Published: 27-03-2024

Abstract: *Microservices is one way to maximize the performance of the data service that will be used to create the system. In addition, it is easy to develop, because each service is structured and separate for data that will be consumed by other developers. The case used in this research is the E-Canteen Application, where the circulation of data from partners, customers and transactions that will be consumed by users must run smoothly. By implementing microservices in this E-Canteen Application, the circulation of data, the speed of the data generated will be very helpful to maximize each service used. The results of testing the microservices that have been created, show that the circulation of data and services that have been used become very efficient and stable when there are many requests or accesses from users. With these results, the effect of microservices for data circulation used becomes very efficient and stable.*

Keywords: *Microservices, E-Kantin, API*

Pendahuluan

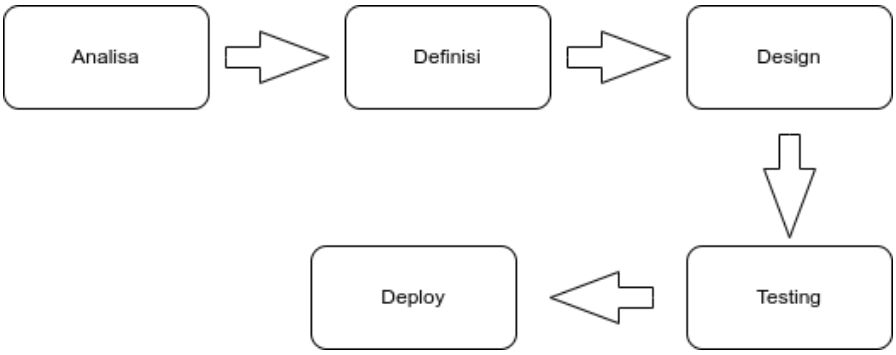
Pada perkembangan teknologi saat ini sebuah Arsitektur Microservice sangat dibutuhkan untuk memudahkan pengembangan sebuah aplikasi/sistem. Pada Microservice komunikasi antar setiap service atau layanan nya menggunakan API (*Application programming Interface*), karena cocok sebagai perantara service yang saling berhubungan. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berhubungan dan berinteraksi (Pratasik & Rianto, 2020). API sendiri adalah sebuah perantara yang bisa digunakan untuk menyatukan banyak service atau menghubungkan antara tampilan dan data di sebuah sistem. Keunggulan dari Microservices adalah leluasa saat ada pengembangan sistem yang rumit, data saat terdistribusi menjadi lebih lancar, akses untuk setiap service nya lebih mudah dan terstruktur (Engelenburg, 2019; Z. Yang, 2019).

Kasus yang digunakan untuk implementasi microservice ini adalah aplikasi E-Kantin. Yang dimana untuk pengembangan fiturnya akan memudahkan developer, karena servicenya sudah terstruktur sesuai dengan yang di rencanakan (Din, 2019; Pourvahab, 2019). Jika dibandingkan dengan struktur monolith yang dimana tampilan, logic dan data acces / querynya dijadikan 1 pada halaman tersebut, akan mempersulit pengembangan pada sistem. Selain itu jika fitur yang akan dikembangkan memiliki alur yang sangat rumit akan mempersulit developer untuk menambah atau mengubahnya, karena harus tau bagaimana tampilan, logic dan query pada fitur tersebut agar tidak merusak alur yang sudah berjalan (Nisar, 2020; Sharma, 2021).

Untuk mengatasi masalah ini, peneliti menerapkan sebuah Architecture Microservices agar pengembangan pada sistem menjadi mudah dan data yang dikonsumsi akan menjadi lebih lancar (C. Yang, 2020). Dengan menggunakan Microservice ini diharapkan agar data yang dikonsumsi akan semakin maksimal performanya, karena setiap layanan datanya terstruktur dan terdistribusi dengan baik.

Dengan adanya permasalahan di atas maka peneliti membuat solusi dengan judul "Arsitektur Perangkat Lunak Berbasis Layanan Mikro Pada Sistem Manajemen Informasi Kantin". Agar distribusi data dari aplikasi tersebut bisa lancar dan pengembangannya menjadi lebih mudah maka, Architecture Microservices harus diterapkan untuk konsumsi data setiap servicenya.

Metode



Gambar 1. Metode Agile

Pada gambar 1 diatas, menjelaskan tentang metode yang digunakan yaitu metode agile. Agile development merupakan pendekatan lebih lanjut dari SDLC (*System Development Life Cycle*) untuk memfasilitasi pengembangan aplikasi yang membutuhkan waktu yang singkat, dan memberikan tingkat keberhasilan pengembangan aplikasi lebih baik dari metode desain terstruktur (Ramadhani, 2015). Ditahap pertama adalah analisa, yang dimana menggunakan referensi jurnal terdahulu. Tahap kedua definisi, menjelaskan tentang teknologi, bahasa pemrograman dan framework yang digunakan. Tahap selanjutnya desain dari microservice yang akan dibuat, setelah desain dibuat selanjutnya adalah testing. Yang dimana menggunakan aplikasi yang bernama postman. Tahap terakhir adalah deploying yang dimana *microservice* akan disebarkan dan diakses oleh *frond end developer* (Zhao, 2019).

A. Analisa

Pada penelitian ini, penelitian terdahulu dijadikan sebagai masukan dan referensi untuk membantu penelitian sehingga dapat menambah teori dan wawasan. Jurnal penelitian yang digunakan untuk referensi adalah penelitian (Kautsar et al., 2023) menjelaskan proses migrasi sistem alat pendukung rapid prototyping dari monolit ke arsitektur microservice yang akan digunakan sebagai implementasi *Project Based Learning*. Penelitian yang dilakukan oleh (Suryotrisongko, 2017) yang berjudul "Arsitektur *Microservice* untuk Resiliensi Sistem Informasi". Didalam penelitian tersebut dijelaskan bahwa dengan menggunakan Microservices menunjukkan peningkatan kualitas pada aspek resiliensi, misalkan ketika beberapa service mengalami gangguan, sistem dapat tetap berjalan sebagaimana mestinya. Penelitian yang dilakukan oleh (Sinambela et al., 2021) yang berjudul "Implementasi Arsitektur Microservices pada Rancang Bangun Aplikasi Marketplace Berbasis Web" (Hu, 2019). Aplikasi tersebut menggunakan arsitektur microservices telah berhasil dibangun dengan hasil persentase pengujian black box 100% dari 25 aktivitas berhasil dan hasil persentase pengujian endpoint api 100% dari 29 aktivitas berhasil.

B. Definisi

Paragraf ini akan menjelaskan tentang definisi apa saja yang digunakan untuk membuat sebuah *microservices.*

a. Microservices

Microservices adalah kumpulan layanan yang saling berkaitan untuk membangun sebuah sistem yang terstruktur, agar sistem tersebut berjalan dengan maksimal. Aplikasi dibagi menjadi bagian-bagian kecil yang berfungsi spesifik (*high cohesion*) dan tidak bergantung pada komponen program lainnya (*loose coupling*), dengan antarmuka API (*Application Programming Interface*) (Newman, 2015). Komunikasi antar setiap service atau layanan nya menggunakan API, karena cocok sebagai perantara service yang saling berhubungan (Priyadarsini, 2021). Microservices berguna untuk memaksimalkan layanan dari setiap data yang digunakan dan mempermudah developer mengembangkan sistem tersebut.

b. Python

Python merupakan salah satu bahasa pemrograman yang banyak digunakan oleh perusahaaan besar maupun para developer untuk mengembangkan berbagai macam aplikasi berbasis desktop, web dan mobile (Schuerer & Maufrais, 2010). Bahasa pemrograman Python menggunakan struktur orientasi objek yang bertujuan untuk memudahkan seorang programmer menulikan kode yang jelas, bagus dan logis untuk project skala kecil mapun besar.

c. Framework Flask

Framework Flask merupakan micro web framework yang menggunakan bahasa python (Ghimire, 2020). Micro framework ini berguna untuk mempercepat pembangunan suatu aplikasi karena sudah ada struktur, library dan komponen lain nya untuk menunjang developer tanpa harus membuat aplikasi dari awal atau nol. Flask dapat menggunakan ekstensi untuk menambahkan fitur dan komponen yang sudah disediakan oleh pihak ketiga dan

tidak terpasang secara standar pada Flask seperti *Form Validation, Upload Handling*, dan Database (Aslam et al., 2015).

d. API

API merupakan sebuah sistem penyedia layanan yang berjalan pada sisi server (Rulloh et al., 2017; Xu et al., 2019). *Application Programming Interface* (API) juga dapat diartikan sebagai perantara atau penghubung antara service database dengan interface aplikasi. API memudahkan developer untuk membangun aplikasi karena service dari data dipisah dan tidak dijadikan satu dengan aplikasi atau monolith, sehingga developer akan mudah untuk mengembangkan aplikasi tersebut meskipun banyak fitur yang sudah berjalan.

e. Firebase

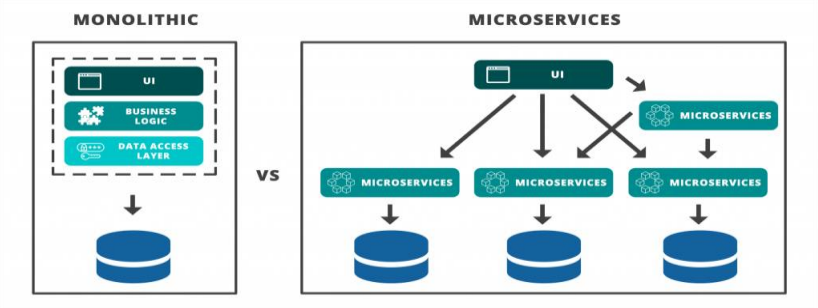
Firebase Database merupakan penyimpanan basis data nonSQL yang memungkinan untuk menyimpan beberapa tipe data (Sandy et al., 2017). Firebase memudahkan developer untuk menyimpan data karena berbasis BaaS (Backend as a Service), Sehingga dalam mengembangkan aplikasi developer tidak memberikan effort yang besar untuk urusan backend. Firebase dapat digabungkan dengan framework lain seperti node, java, javascript, dan lain-lain (Sanad, 2019).

f. Postman

Postman adalah sebuah aplikasi (berupa plugin) untuk browser chrome, yang berfungsi sebagai REST Client, yang digunakan untuk melakukan uji coba REST API (Lewiani et al., 2017). Cara penggunaan dari postman sendiri yaitu request ke url yang akan kita tuju dan isi semua parameter yang sudah dibuat oleh developer, agar data yang muncul sesuai dengan yang anda inginkan.

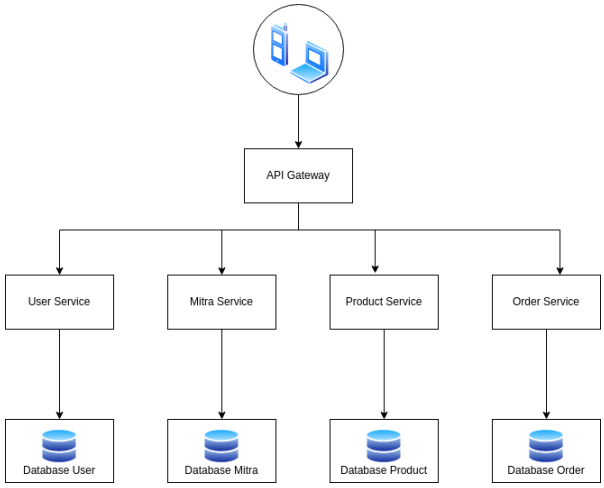
C. Design

Setelah definisi dari *microservices*, tahap selanjutnya adalah desain untuk microservices. Pada gambar 2 dibawah menjelaskan antara monolit dan microservices yang berjalan, monolit merupakan model yang dibangun dalam satu codebase yang sama atau ui dan data nya dibuat pada halaman yang sama. Sedangkan *microservices* dibangun secara terpisah dari ui atau tampilan nya, dan berbagai services nya juga dibuat terpisah agar memaksimalkan performa dari request data yang akan dipanggil atau dipakai.



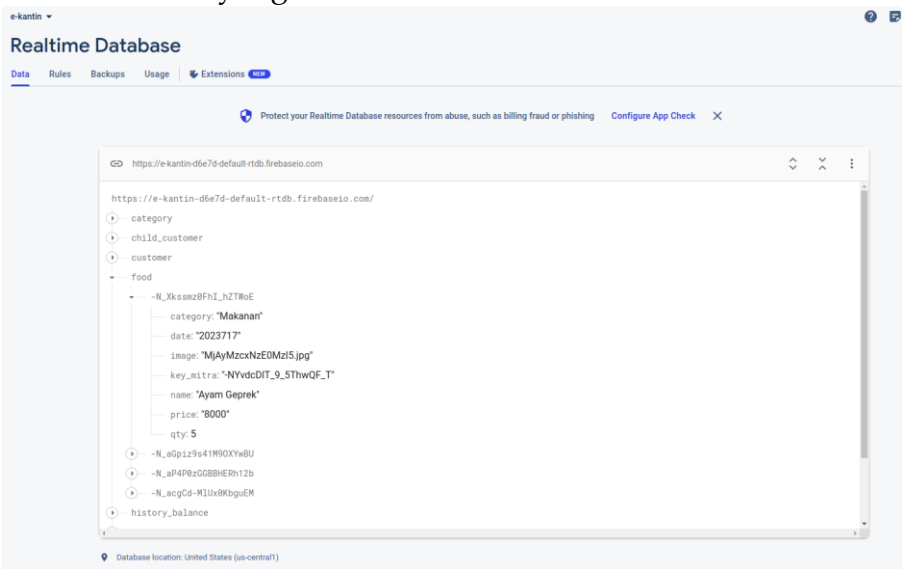
Gambar 2. Monolith dan Microservices

Pada gambar 3 dibawah adalah alur dari setiap services yang akan diakses atau dikonsumsi oleh *front end developer* dan service akan mengakses setiap database yang sudah ditentukan.



Gambar 3. Alur Service Database

Selanjutnya adalah gambar 4, tampilan database firebase yang digunakan. Pada firebase tampilan data yang masuk akan disimpan seperti JSON dan disinkronkan secara realtime ke setiap user penggunanya, yang memudahkan mengelola suatu data yang berskala besar.



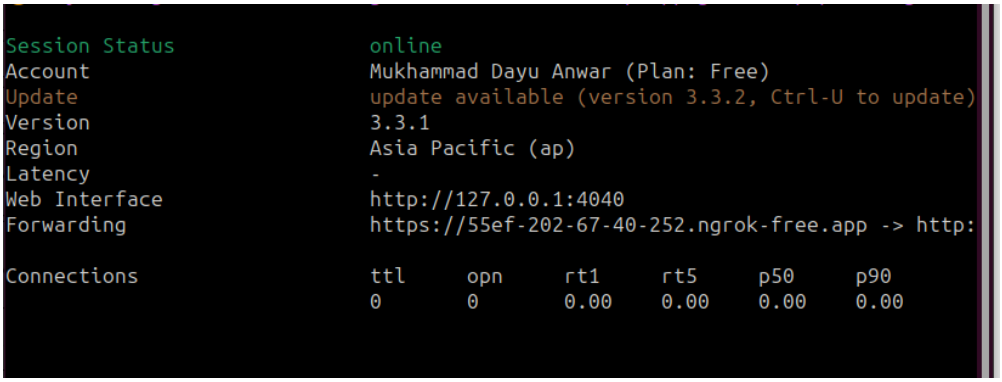
Gambar 4. Firebase

D. Testing

Untuk tahap testing API menggunakan aplikasi postman, karena untuk memastikan seluruh services yang dibuat telah berjalan sesuai yang diharapkan. Pada tahap ini sangat penting mengevaluasi dan menyesuaikan jika ada salah satu services yang tidak sesuai dengan apa yang telah disepakati.

E. Deploy

Deploy adalah menyebarkan sistem yang sudah dibuat agar bisa diakses banyak orang. Pada gambar 5 dibawah adalah deploy menggunakan aplikasi ngrok. Aplikasi ini cocok untuk membuat sebuah sistem agar bisa diakses secara online dan gratis, karena dapat membuka jaringan private melalui firewall dan menghubungkan localhost ke internet dengan tunel yang aman. Ngrok sendiri dapat membantu membuat bagian dengan aman serta memberikan URL server (Ghorpade et al., 2019).



Gambar 5. Run Ngrok

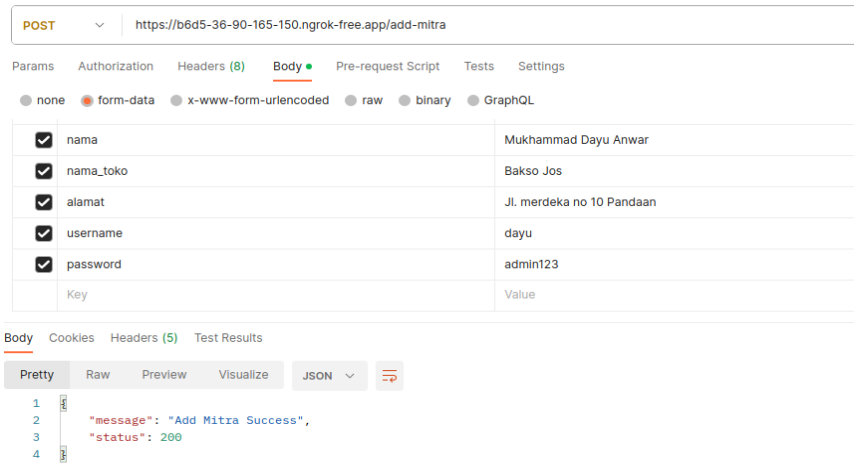
Hasil dan Pembahasan

A. Implementasi

Pada bab ini akan menjelaskan mengenai hasil dan pembahasan dari API yang telah dibuat berdasarkan analisa yang telah dijelaskan diatas. Format data yang akan ditampilkan pada API dibawah adalah json, yang dimana format data ini kompatible dengan banyak bahasa pemrograman, environment dan library (Lee, 2021). Berikut ini adalah tampilan dari API yang telah dibuat dan diakses menggunakan postman.

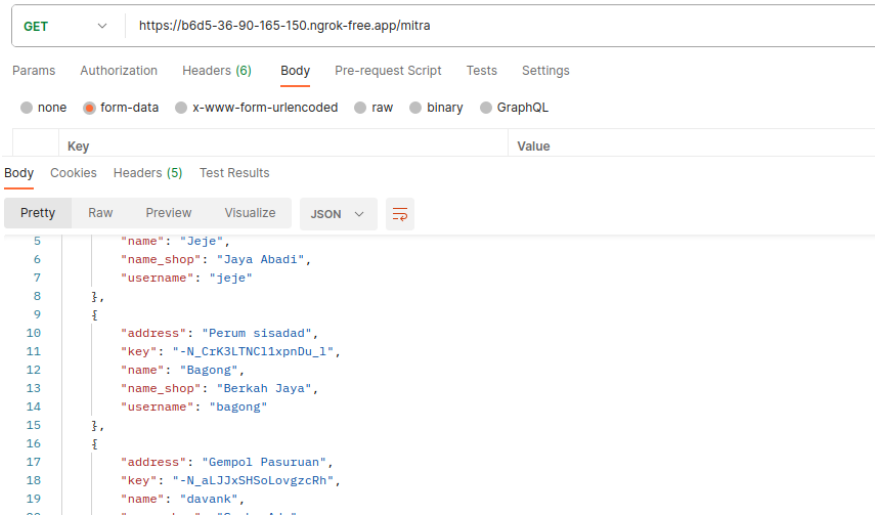
a. API Mitra

Pada gambar 6 dibawah ini, menjelaskan bagaimana cara untuk menambahkan mitra. Pada tulisan post itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service tambah mitra. Untuk body dan form data ialah parameter yang harus dibawa untuk menambahkan mitra, setelah itu akan mengembalikan sebuah json data yang berisi “berhasil menambahkan mitra”.



Gambar 6. API request untuk menambah data mitra

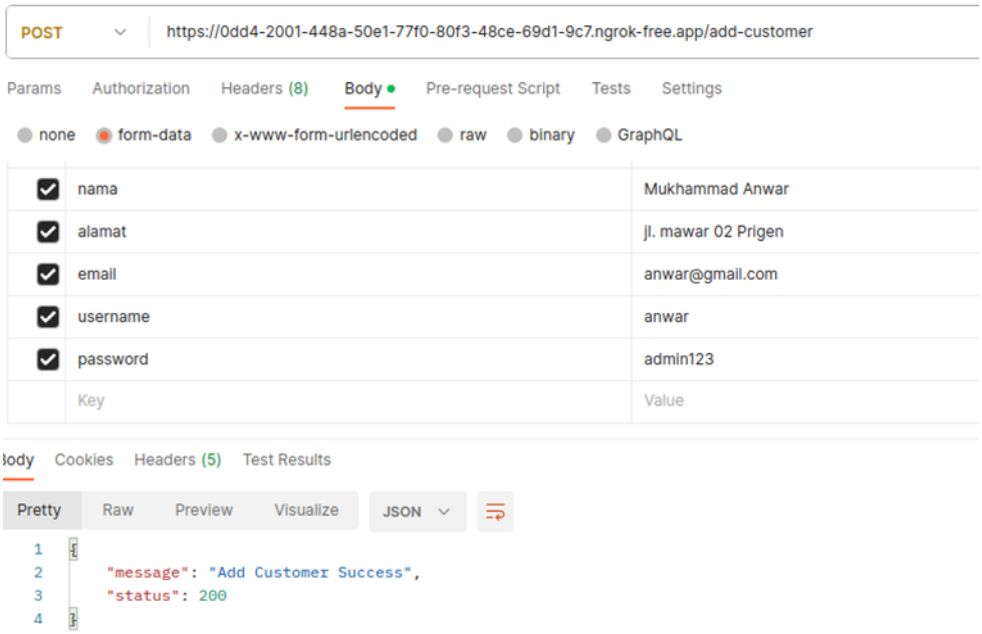
Pada gambar 7 dibawah ini, menjelaskan bagaimana cara untuk menampilkan data semua mitra (Su, 2020). Pada tulisan get itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service list mitra. Setelah itu akan mengembalikan sebuah json data yang berisi semua data mitra.



Gambar 7. API list semua data mitra

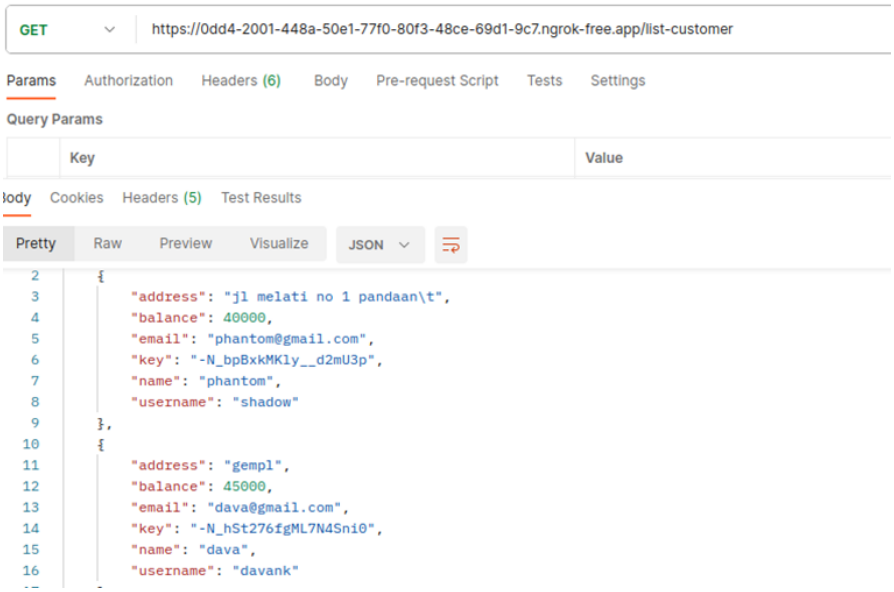
b. API Customer

Pada gambar 8 dibawah ini, menjelaskan bagaimana cara untuk menambahkan customer. Pada tulisan post itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service tambah customer (Jiang, 2020). Untuk body dan form data ialah parameter yang harus dibawa untuk menambahkan customer, setelah itu akan mengembalikan sebuah json data yang berisi "berhasil menambahkan customer".



Gambar 8. API request untuk menambahkan data customer

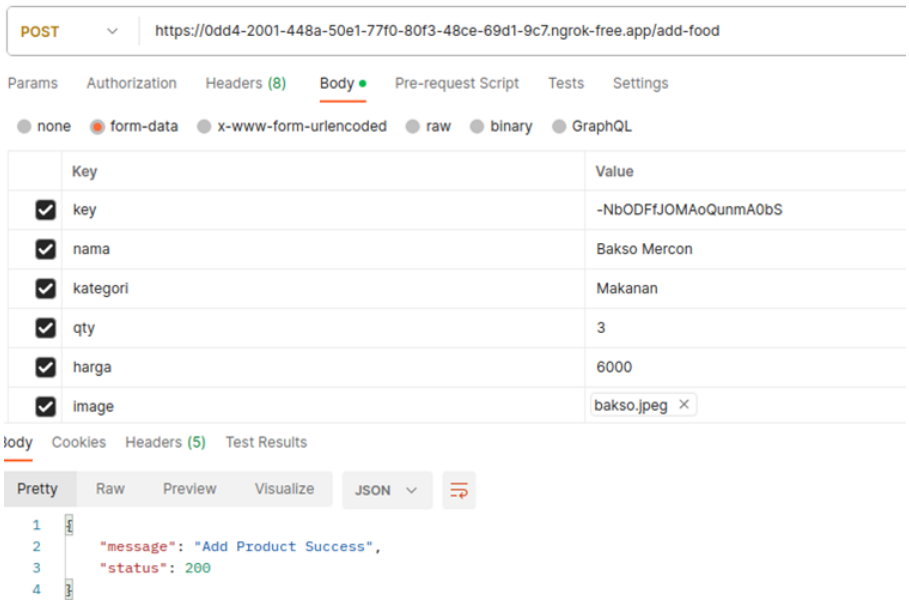
Pada gambar 9 dibawah ini, menjelaskan bagaimana cara untuk menampilkan data semua customer. Pada tulisan get itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service list customer. Setelah itu akan mengembalikan sebuah json data yang berisi semua data customer.



Gambar 9. API untuk list semua data customer

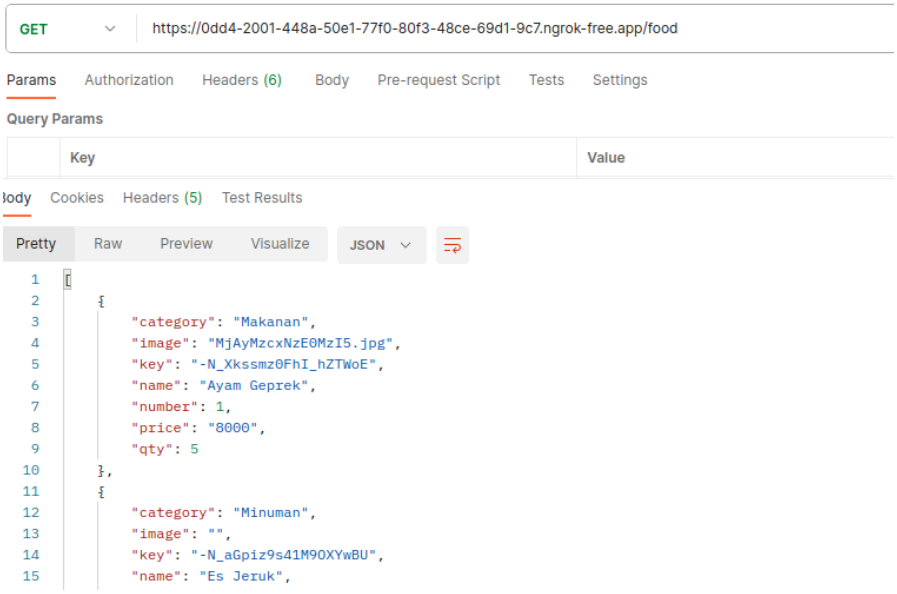
c. API Produk

Pada gambar 10 dibawah ini, menjelaskan bagaimana cara untuk menambahkan produk. Pada tulisan post itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service tambah produk (Bi, 2019). Untuk body dan form data ialah parameter yang harus dibawa untuk menambahkan produk, setelah itu akan mengembalikan sebuah json data yang berisi "berhasil menambahkan produk".



Gambar 10. API request untuk menambah data produk

Pada gambar 11 dibawah ini, menjelaskan bagaimana cara untuk menampilkan data semua produk. Pada tulisan get itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service list produk. Setelah itu akan mengembalikan sebuah json data yang berisi semua data produk.

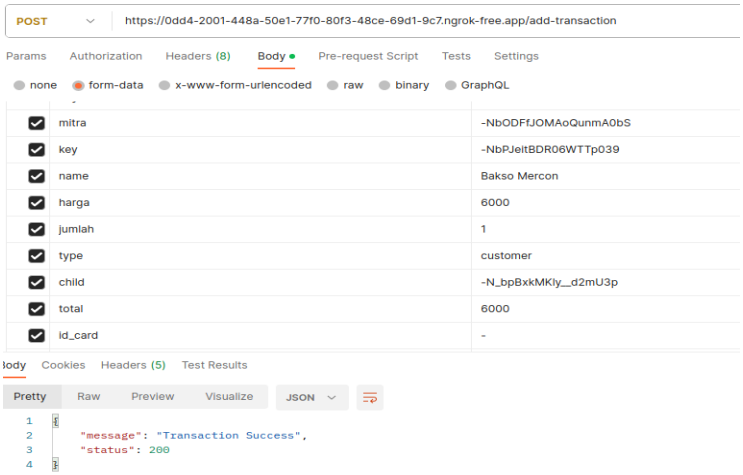


Gambar 11. API list semua data produk

d. API Transaksi

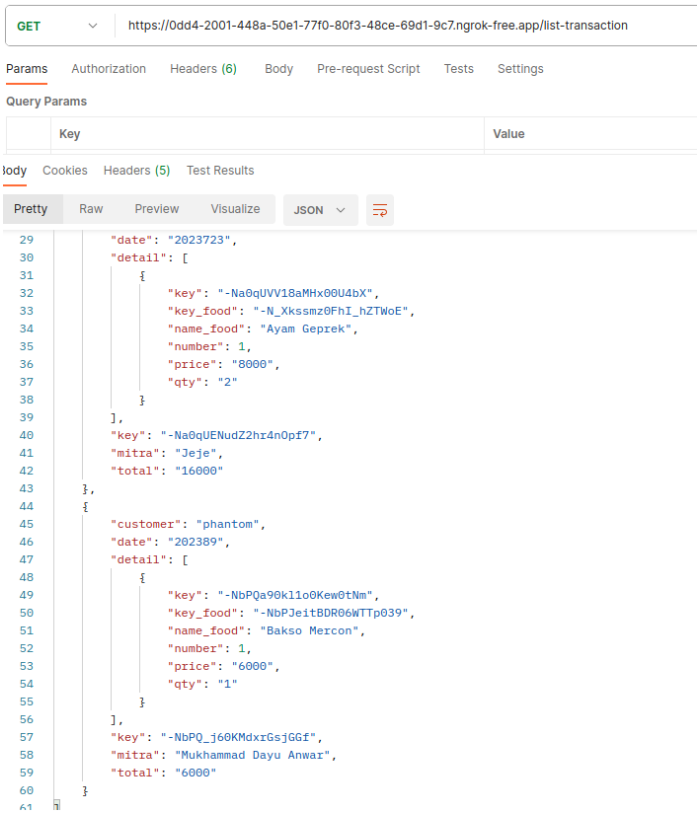
Pada gambar 12 dibawah ini, menjelaskan bagaimana cara untuk melakukan transaksi pembelian produk. Pada tulisan post itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service transaksi pembelian produk (Varadharajan, 2019). Untuk body dan form data ialah parameter yang harus dibawa untuk bisa melakukan

transaksi pembelian produk, setelah itu akan mengembalikan sebuah json data yang berisi “transaksi berhasil”.



Gambar 12. API request transaksi pembelian atas customer

Pada gambar 13 dibawah ini, menjelaskan bagaimana cara untuk menampilkan data semua transaksi yang berhasil masuk. Pada tulisan get itu adalah method yang akan direquest, disamping nya itu adalah url yang harus diisi untuk menuju ke service transaksi yang berhasil masuk. Setelah itu akan mengembalikan sebuah json data yang berisi semua data transaksi yang berhasil masuk.



Gambar 13. API request list semua data transaksi yang berhasil masuk

B. Pengujian Services

Dari pengujian API menggunakan postman diatas, diketahui pada tabel 1 dibawah ini menunjukan kalau setiap services nya berjalan dengan lancar dan sesuai dengan kebutuhan yang sudah direncanakan. Dan juga pada tabel 1 dibawah ini akan menjelaskan bahwa pengujian terhadap endpoint yang ada pada setiap services nya telah berhasil dibuat dan dijalankan tanpa ada masalah.

Tabel 1. Pengujian endpoint / services API menggunakan Postman

No	Skenario Pengujian	Hasil Yang Diharapkan	Hasil Pengujian
1.	API Tambah Mitra	Menampilkan data json yang berhasil	[✓] Berhasil [ ] Tidak Berhasil
2.	API List Mitra	Menampilkan list mitra yang sudah dimasukkan	[✓] Berhasil [ ] Tidak Berhasil
3.	API Tambah Customer	Menampilkan data json yang berhasil	[✓] Berhasil [ ] Tidak Berhasil
5.	API List Customer	Menampilkan list customer yang sudah dimasukkan	[✓] Berhasil [ ] Tidak Berhasil
6.	API Tambah Produk	Menampilkan data json yang berhasil	[✓] Berhasil [ ] Tidak Berhasil
7.	API List Produk	Menampilkan list produk yang sudah dimasukkan	[✓] Berhasil [ ] Tidak Berhasil
8.	API Transaksi	Menampilkan data json yang berhasil	[✓] Berhasil [ ] Tidak Berhasil
9.	API List Transaksi Berhasil	Menampilkan list transkasi yang berhasil	[✓] Berhasil [ ] Tidak Berhasil

Simpulan

Berdasarkan penelitian diatas disimpulkan bahwa microservices yang telah dibuat dapat membantu memaksimalkan performa data yang akan dikonsumsi oleh user pengguna. Selain itu untuk pengembangan dengan mengikuti bahasa pemrograman / framework yang terbaru dan terpopuler akan sangat mudah, karena struktur dan setiap servicesnya mudah untuk diubah dan ditambah. Dari sisi Frontend / Tampilan pun mudah dikonsumsi dan meskipun services nya berubah tidak harus mengubah semua tampilan yang ada, hanya menyesuaikan services yang diubah saja.

Daftar Pustaka

Aslam, F. A., Mohammed, H. N., Musab, J., & Munir, M. (2015). Efficient Way Of Web Development Using Python And Flask. *International Journal of Advanced Research in Computer Science*, *6*(2), 54–57.

Bi, Y. (2019). Software Defined Space-Terrestrial Integrated Networks: Architecture, Challenges, and Solutions. *IEEE Network*, *33*(1), 22–28. https://doi.org/10.1109/MNET.2018.1800193

Din, S. (2019). 5G-enabled Hierarchical architecture for software-defined intelligent transportation system. *Computer Networks*, *150*, 81–89. https://doi.org/10.1016/j.comnet.2018.11.035

Engelenburg, S. (2019). Design of a software architecture supporting business-to-government information sharing to improve public safety and security: Combining business rules, Events and blockchain technology. *Journal of Intelligent Information Systems*, *52*(3), 595–618. https://doi.org/10.1007/s10844-017-0478-z

Ghimire, D. (2020). *Comparative Study on Python Web Frameworks: Flask and Django*.

Ghorpade, D., Jadhav, S., Gunjal, S., Bogir, S., & Tambe, P. S. (2019). Survey on Intelligent System for College. *International Research Journal of Engineering and Technology (IRJET)*, *6*(10), 1744–1745.

Hu, P. (2019). An open internet of things system architecture based on software-defined device. *IEEE Internet of Things Journal*, *6*(2), 2583–2592. https://doi.org/10.1109/JIOT.2018.2872028

Jiang, W. (2020). Hardware/Software Co-Exploration of Neural Architectures. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *39*(12), 4805–4815. https://doi.org/10.1109/TCAD.2020.2986127

Kautsar, I. A., Maika, M. R., Budiman, A. N., Setyawan, A. B., & Awali, J. Y. (2023). Microservice Based Architecture: The Development of Rapid Prototyping Supportive Tools for Project Based Learning. *2023 IEEE World Engineering Education Conference (EDUNINE)*, 1–6. https://doi.org/10.1109/EDUNINE57531.2023.10102884

Lee, S. (2021). Hardware architecture and software stack for PIM based on commercial DRAM technology: Industrial product. *Proceedings - International Symposium on Computer Architecture*, *2021*, 43–56. https://doi.org/10.1109/ISCA52012.2021.00013

Lewiani, N., Lisnawaty, & Akifah. (2017). Proses Pengelolaan Klaim Pasien BPJS Unit Rawat Inap Rumah Sakit Dr. R. Ismoyo Kota Kendari Tahun 2016. *Jurnal Ilmiah Mahasiswa Kesehatan Masyarakat*, *4*, 250–731.

Newman, S. (2015). *Building Microservices*. O'Reilly Media, Inc.

Nisar, K. (2020). A survey on the architecture, application, and security of software defined networking: Challenges and open issues. *Internet of Things (Netherlands)*, *12*. https://doi.org/10.1016/j.iot.2020.100289

Pourvahab, M. (2019). An efficient forensics architecture in software-defined networking-IoT using blockchain technology. *IEEE Access*, *7*, 99573–99588. https://doi.org/10.1109/ACCESS.2019.2930345

Pratasik, S., & Rianto, I. (2020). Pengembangan Aplikasi E-DUK Dalam Pengelolaan SDM Menggunakan Metode Agile Development. *CogITo Smart J.*, *6*(2), 204. https://doi.org/10.31154/cogito.v6i2.267.204-216

Priyadarsini, M. (2021). Software defined networking architecture, traffic management, security, and placement: A survey. *Computer Networks*, *192*. https://doi.org/10.1016/j.comnet.2021.108047

Ramadhani, M. F. (2015). Pembangunan Aplikasi Informasi, Pengaduan, Kritik, Dan Saran Seputar Kota Cimahi Pada Platform Android. *J. Ilm. Komput. Dan Inform.*, 9.

Rulloh, A., Mahmudah, D. E., & Kabetta, H. (2017). Implementasi REST API pada Aplikasi Panduan Kepaskibraan Berbasis Android. *Teknikom: Teknologi Informasi, Ilmu Komputer Dan Manajemen*, *1.2*, 85–89.

Sanad, E. A. W. (2019). Pemanfaatan Realtime Database di Platform Firebase Pada Aplikasi E-Tourism Kabupaten Nabire. *J. Penelit. Enj.*, *22*(1), 20–26. https://doi.org/10.25042/jpe.052018.04

Sandy, L. A., Januar, R., & Hariadi, R. R. (2017). Rancang Bangun Aplikasi Chat pada Platform Android dengan Media Input berupa Canvas dan Shareable Canvas untuk Bekerja Dalam Satu Canvas secara Online. *JURNAL TEKNIK ITS.*, *6*(2), 2337–3520.

Schuerer, K., & Maufrais, C. (2010). *Introduction to Programming using Python*. Pearson.

Sharma, O. (2021). Recent advances in motion and behavior planning techniques for software architecture of autonomous vehicles: A state-of-the-art survey. *Engineering Applications of Artificial Intelligence*, *101*. https://doi.org/10.1016/j.engappai.2021.104211

Sinambela, A., Ernawati, & Coastera, F. F. (2021). Implementasi Arsitektur Microservices pada Rancang Bangun Aplikasi Marketplace Berbasis Web. *Jurnal Rekursif.*, *9*(1).

Su, L. (2020). Nanophotonic inverse design with SPINS: Software architecture and practical considerations. *Applied Physics Reviews*, *7*(1). https://doi.org/10.1063/1.5131263

Suryotrisongko, H. (2017). Arsitektur Microservice untuk Resiliensi Sistem Informasi. *Jurnal Sisfo.*, *06*(02), 235–250.

Varadharajan, V. (2019). A policy-based security architecture for software-defined networks. *IEEE Transactions on Information Forensics and Security*, *14*(4), 897–912. https://doi.org/10.1109/TIFS.2018.2868220

Xu, H., Wang, H., & Zhang, S. (2019). *Application Programming Interface (API) Service Apparatus and Application Programming Interface (API) Service System* (Issue 16/181,927).

Yang, C. (2020). Big data driven edge-cloud collaboration architecture for cloud manufacturing: A software defined perspective. *IEEE Access*, *8*, 45938–45950. https://doi.org/10.1109/ACCESS.2020.2977846

Yang, Z. (2019). Software-defined wide area network (SD-WAN): architecture, advances and opportunities. *Proceedings - International Conference on Computer Communications and Networks, ICCCN*, *2019*. https://doi.org/10.1109/ICCCN.2019.8847124

Zhao, S. (2019). Sectee: A software-based approach to secure enclave architecture using TEE. *Proceedings of the ACM Conference on Computer and Communications Security*, 1723–1740. https://doi.org/10.1145/3319535.3363205