BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Penelitian ini menggunakan beberapa sumber pustaka yang berhubungan dengan kasus atau metode yang akan diteliti, Diantaranya yaitu :

Frederica, G. N. A., Pandelaki, S., & Kumenap, V. D. (2023), telah melakukan penelitian tentang "Implementasi Arsitektur Microservice pada Aplikasi Pendeteksi Kepiting Soka" menggunakan teknologi microservices, cloud computing (Firebase Authentication, Cloud Firestore, Cloud Storage Bucket), dan TensorFlow Lite untuk mendeteksi kepiting soka secara efisien. Masalah utama yang dihadapi adalah kurangnya efisiensi dalam pendekatan monolitik, sehingga aplikasi sulit dikelola dan diskalakan. Hasilnya menunjukkan bahwa implementasi arsitektur microservices berhasil meningkatkan efisiensi aplikasi dengan memanfaatkan layanan cloud, memungkinkan pengelolaan data yang lebih baik, pengujian model machine learning menggunakan Flask API, dan integrasi yang fleksibel antara model machine learning dan aplikasi Android.

Chandra, C., Wijaya, F., Gunawan, J. A., Lee, J. R., & Maulana, A. (2024), telah melakukan penelitian tentang mengembangkan RESTful API berbasis Node.js dan Express.js untuk aplikasi edukasi EksFlorasi, yang mendorong anak-anak mengenal alam. API diimplementasikan dengan Google Cloud Platform (Cloud Run, Cloud SQL,

dan Cloud Storage), mendukung hingga 50 pengguna simultan dengan performa optimal dan latensi rendah.

Gurohman, D. T., Susanto, B. M., Hariyanto, A., Atmadji, E. S. J., Gumilang, M. A., Antika, E., & Mukhlisoh, N. A. (2024), telah melakukan penelitian tentang menerapkan Horizontal Pod Autoscaler dan Redis Cluster berbasis Kubernetes untuk meningkatkan performa website elearning Moodle. Masalah yang dihadapi adalah kebutuhan sumber daya komputasi besar untuk mendukung akses simultan pengguna. Hasil penelitian menunjukkan bahwa pendekatan ini mampu meningkatkan performa website elearning sebesar 4,3%, lebih baik dibandingkan dengan pendekatan monolitik.

Alchuluq, L. M., & Nurzaman, F. (2021), telah melakukan penelitian tentang analisis penggunaan Arsitektur Microservice dibandingkan dengan Arsitektur Monolitik untuk layanan bisnis toko online. Masalah yang dihadapi adalah kebutuhan aplikasi yang stabil untuk menangani banyak layanan dan trafik besar. Hasil penelitian menunjukkan bahwa Arsitektur Microservice lebih cepat dalam memproses data (888,441 data/menit) dan lebih cocok diterapkan pada aplikasi yang berkembang besar dengan banyak fitur dan trafik tinggi.

Kusuma, I. G. N. A. (2022), telah melakukan penelitian tentang implementasi autentikasi dan otorisasi stateless sederhana dalam aplikasi pencatatan inventori berbasis website menggunakan teknologi Microservice dan REST API. Permasalahan autentikasi pada transisi dari arsitektur monolitik ke microservice diselesaikan dengan membangun authority server dan memanfaatkan protokol HTTP stateless. Hasilnya

menunjukkan peningkatan jumlah data terkirim dengan tambahan waktu pemrosesan kurang dari 35ms yang tidak signifikan.

Pembahasan yang dibuat kali ini tentang “Implementasi Arsitektur Microservice Pada Aplikasi Inventory Barang Berbasis Paas Cloud Computing”. Penelitian ini bertujuan untuk menyelesaikan masalah melalui aplikasi yang penulis buat, berupa aplikasi web Dibuat dengan menerapkan arsitektur Microservices menggunakan RestfulAPI berbasis PaaS.

Tabel tinjauan pustaka merupakan tabel yang dibuat untuk mendefinisikan penelitian yang sebelumnya hampir sama dilakukan dengan penelitian yang diajukan saat ini, adapun perbandingan yang menjadi tabel tinjauan pustaka penelitian yakni :

Tabel 2. 1 Tinjauan Pustaka

Sumber	Judul	Teknologi	Masalah	Hasil
Frederica, G. N. A., Pandelaki, S., & Kumenap, V. D. (2023).	Implementasi Arsitektur Microservice pada Aplikasi Pendeteksi Kepiting Soka	Microservices, Cloud Computing, Machine Learning	Kendala pada budidaya kepiting soka, seperti waktu molting tidak seragam, dan tantangan skalabilitas aplikasi monolitik	Arsitektur microservice berhasil diterapkan, meningkatkan efisiensi dan skalabilitas, serta mempermudah pemeliharaan.
Chandra, C., Wijaya, F., Gunawan, J. A., Lee, J. R., & Maulana, A. (2024).	Perancangan dan Implementasi RESTful API untuk Aplikasi Mobile Pembelajaran Flora dan	Node.js, Express.js, RESTful API, Google Cloud Platform (Cloud Run, Cloud SQL, Cloud Storage)	Anak-anak lebih banyak menghabiskan waktu dengan smartphone di rumah setelah pandemi, sehingga kurang	RESTful API berhasil dikembangkan dan di-deploy menggunakan layanan Google Cloud Platform, memberikan

	Fauna pada Google Cloud Platform		eksplorasi alam yang mendukung perkembangan sosial dan kognitif.	performa optimal hingga 50 pengguna secara bersamaan.
Gurohman, D. T., Susanto, B. M., Hariyanto, A., Atmadji, E. S. J., Gumilang, M. A., Antika, E., & Mukhlisoh, N. A. (2024).	Penerapan Horizontal Pod Autoscaler dan Redis Cluster Berbasis Kubernetes untuk Meningkatkan Performa Website Elearning	Kubernetes, Redis Cluster, Moodle	Server elearning membutuhkan sumber daya komputasi besar untuk mendukung akses pengguna secara simultan.	Penerapan horizontal pod autoscaler dan redis cluster meningkatkan performa elearning Moodle sebesar 4,3%, lebih baik dibandingkan pendekatan monolitik.
Alchuluq, L. M., & Nurzaman, F. (2021).	Analisis pada Arsitektur Microservice untuk Layanan Bisnis Toko Online	Arsitektur Microservice, Arsitektur Monolitik	Toko online membutuhkan aplikasi stabil untuk menangani layanan dan trafik yang besar.	Arsitektur Microservice dapat memproses data lebih cepat (888,441 data/menit) dibandingkan Arsitektur Monolitik, dan cocok untuk aplikasi dengan banyak fitur dan trafik besar.
Kusuma, I. G. N. A. (2022).	Aplikasi Pencatatan Inventori Berbasis Website dengan Skema Autentikasi dan Otorisasi	Microservice, REST API, Stateless HTTP	Permasalahan pada proses autentikasi dan otorisasi saat beralih dari konsep monolitik ke microservice	Konsep stateless sederhana dapat diimplementasikan dengan peningkatan jumlah data terkirim, namun tidak signifikan (waktu

	Stateless Sederhana			pemrosesan < 35ms)
I Gst Made Adi Surya Darma (2024)	Implementasi Arsitektur Microservice Pada Aplikasi Inventory Barang Berbasis Paas Cloud Computing	Microservices, Laravel lumen, Vue.Js, PostgreSQL, RestfulAPI, Heroku	Membangun Sistem Inventory Barang dengan Memanfaatkan Arsitektur Microservices dan RestfulAPI	Akan Dibuat Sistem Inventory Gudang dengan penyimpanan data terpusat Memanfaatkan Arsitektur Microservices dan RestfulAPI

2.2 Dasar Teori

2.2.1 Fremework

Framework adalah sebuah kerangka kerja yang digunakan untuk mempermudah *developer software* dalam membuat dan mengembangkan aplikasi. *Framework* berisikan fungsi dasar dan perintah yang dipakai untuk membuat dan mengembangkan sebuah aplikasi dengan harapan aplikasi yang dibuat dapat dibangun secara lebih terstruktur, lebih cepat serta tersusun dengan rapi. Diantaranya fungsi *framework* sebagai berikut:

1. Mempercepat proses pembuatan aplikasi
2. Membantu developer dalam perencanaan, pembuatan, dan pemeliharaan aplikasi.
3. Memiliki tingkat keamanan yang lebih baik.
4. Dapat menghemat waktu dan biaya.
5. Memudahkan dalam mencari *bugs* didalam kode program.

2.2.2 Laraval lumen

Lumen Laravel adalah versi ringan dari Laravel, yang dikembangkan oleh Taylor Otwell untuk membangun aplikasi web API yang cepat dan efisien, dengan fitur seperti routing, database migration, dan middleware untuk mempermudah pengembangan. Sementara Laravel, juga dikembangkan oleh Taylor Otwell, adalah framework open-source yang mengikuti struktur MVC, menggabungkan fitur terbaik dari berbagai framework seperti CodeIgniter dan Ruby on Rails, serta menawarkan kemudahan dan kecepatan dalam pembuatan aplikasi web berkat fitur seperti migrasi dan artisan CLI. Berikut ini beberapa fitur yang dimiliki oleh *framework* laravel:

1. *Bundles* yaitu sebuah fitur dengan sistem pengemasan modular dan berbagai *bundles* telah tersedia untuk digunakan dalam aplikasi.
2. *Eloquent* ORM merupakan penerapan PHP lanjutan dari pola "*active record"* menyediakan metode internal untuk mengatasi kendala hubungan antara objek *database.* Pembangunan *query Laravel Fluent* didukung *Eloquent.*
3. *Application Logic* merupakan bagian dari aplikasi yang dikembangkan, baik menggunakan *Controllers* maupun sebagai bagian dari deklarasi *Route*.
4. *Authentication* adalah bagian internal dari aplikasi web modern, Laravel menyediakan otentikasi diluar kotak, dengan menjalankan perintah sederhana.
5. *Modularity* adalah sejauh mana komponen aplikasi web dapat dipisahkan dan digabungkan kembali. Modularitas dapat membantu untuk mempermudah proses *update.*

6. *Caching* adalah sebuah teknik untuk menyimpan data di lokasi penyimpanan sementara dan dapat diambil dengan cepat saat dibutuhkan.

2.2.3 Pengertian Inventory

Pengertian dari inventory adalah barang yang dikelola oleh perusahaan dengan tujuan untuk dijual. Inventory dapat berupa bahan mentah yang dibeli dan diubah menjadi sesuatu yang sama sekali baru. Selain itu bisa juga berupa produk massal yang diuraikan menjadi bagian-bagian penyusunnya dan jual secara terpisah. Bahkan bisa menjadi sesuatu yang sama sekali tidak berwujud. Berikut adalah beberapa jenis dari inventory itu sendiri:

a. Jenis-jenis Inventory
    1) Barang jadi / barang untuk dijual, adalah produk yang dijual ke pelanggan
    2) Bahan baku, adalah persediaan yang digunakan untuk membuat barang jadi
    3) Work in Process, yang artinya barang yang belum jadi – persediaan yang merupakan bagian dari proses manufaktur
    4) Barang MRO, MRO adalah singkatan dari pemeliharaan, perbaikan dan pengoperasian. Barang ini adalah inventory yang digunakan untuk mendukung proses produksi
    5) Stok pengaman, adalah persediaan tambahan yang disimpan untuk mengatasi kekurangan pemasok atau lonjakan permintaan.

b. Manfaat Sistem Inventory

Sistem inventory berguna untuk menentukan jumlah persediaan yang optimal dengan biaya total yang minimal. Persediaan atau inventory meliputi bahan mentah atau bahan baku, bahan pembantu, bahan dalam proses atau work in process, suku cadang, dan barang jadi atau finished good, karena timbulnya ketidakpastian permintaan, ketidakpastian pasokan supplier, dan ketidakpastian waktu pemesanan.

2.2.4 PHP

PHP atau *Hypertext Preprocessor* adalah bahasa pemrograman *script server side* yang sengaja dirancang lebih cenderung untuk membuat dan mengembangkan web. Bahasa pemrograman ini dirancang untuk pengembang web agar dapat menciptakan suatu halam web yang bersifat dinamis.

PHP diciptakan oleh Rasmus Lerdorf seorang pemrogram C, dan digunakan untuk mencatat jumlah pengunjung pada *homepage.* Pada awal tahun 1995 dirilis PHP/FI (*Form Interperter*) yang memiliki kemampuan dasar membangun aplikasi web, memproses form dan mendukung data MySQL.

2.2.5 PostgreSQL

PostgreSQL atau disebut Postgres merupakan salah satu dari sekian banyak database besar yang ada, yang menawarkan skalabilitas, keluwesan, dan kinerja yang tinggi bagi user

Postgres pertama kali muncul pada tahun 1996. PostgresSQL merupakan database server untuk mengolah data yang bersifat open source, dan memiliki lisensi GPL (General Public License) serta merupakan salah satu dari sekian banyak database server yang ada.

Postgres adalah ORDBMS (Object Relational Database Management System) yang bersifat open. Beberapa fitur PostgreSQL adalah sebagai berikut:

1) Inheritance, dimana satu table dapat diturunkan model dan beberapa karakteristik dari table lainnya
2) Multi Version Concurrency Control (MVCC), dimana user diberi data snapshot ketika suatu perubahan dilakukan sampai commit.
3) Rules, dimana suatu query DML yang dikirimkan ke serverakan mengalami penulisan ulang (rewrite). Ini terjadi sebelum diproses lebih lanjut oleh query planner.

2.2.6 Vue.js

Vue.js adalah sebuah progresif framework untuk membangun user interface tidak seperti beberapa framework monolitik lainnya, vue dirancang dari bagian paling dasar agar dapat diadopsi secara bertahap. Inti pustakanya difokuskan pada layer tampilan saja dan sangat mudah untuk diintegrasikan dengan library lain atau proyek yang sudah ada. di sisi lain vue mampu mendukung single page aplication yang canggih saat dikombinasikan dengan modern tooling dan library support.

2.2.7 Platfom As A Service (PAAS)

Platform-as-a-Service (PaaS) adalah model cloud computing yang menyediakan lingkungan untuk pengembangan dan penyebaran aplikasi melalui internet. Dalam PaaS, penyedia layanan cloud menyediakan infrastruktur, middleware, alat pengembangan, dan sistem manajemen basis data, sehingga pengguna dapat fokus pada pengembangan aplikasi tanpa perlu mengelola infrastruktur yang kompleks. PaaS memungkinkan efisiensi dan fleksibilitas lebih tinggi, mempercepat waktu pemasaran aplikasi, serta mengurangi biaya dan kompleksitas infrastruktur (Silitonga & Ni, 2024).

PaaS juga menawarkan kemudahan dalam pengelolaan skalabilitas, keamanan, dan pemeliharaan sistem, yang sepenuhnya ditangani oleh penyedia layanan. Layanan ini mencakup framework aplikasi dan bahasa pemrograman, mempermudah pengguna dalam membuat, menguji, dan menyebarkan aplikasi web tanpa kendali langsung atas perangkat keras, sistem operasi, atau jaringan. Dengan demikian, pengguna tidak perlu khawatir tentang instalasi atau pemeliharaan server, karena semuanya ditangani oleh penyedia layanan (Shiddiq, 2021).

2.2.8 Heroku

Heroku adalah platform cloud Platform-as-a-Service (PaaS) yang memungkinkan pengguna untuk membangun, menjalankan, dan mengoperasikan aplikasi secara keseluruhan. Sebagai layanan PaaS, Heroku memberikan kemudahan bagi pengguna dengan menangani infrastruktur, server, dan hardware sehingga pengguna dapat fokus pada pengembangan aplikasi. Proses deploy aplikasi pada

Heroku cukup sederhana, yaitu melalui konfigurasi dengan platform yang didukung (Syaputro & Widya, 2022).

Heroku juga menyediakan layanan web yang memungkinkan pengembang untuk menempatkan aplikasi mereka di ruang publik agar dapat diakses oleh semua orang. Dengan kemudahan ini, Heroku menjadi solusi yang ideal untuk pengembang yang ingin memanfaatkan cloud computing untuk menjalankan aplikasi mereka tanpa harus mengelola infrastruktur secara langsung (Wijaya & Arfandy, 2019).

2.2.9 Microsevices

Microservice adalah kumpulan beberapa proses yang berkomunikasi antar service lain untuk membentuk sebuah aplikasi yang kompleks terhadap bahasa API apapun. Microservice merupakan pengembangan lanjutan dari Service-oriented Architecture karena microservice merupakan sistem yang terdiri dari komponen servis-servis blok, kecil, terpisah dan fokus pada tugas-tugasnya atau bekerja secara metode modular, autonomous untuk tujuan masing-masing namun terkoneksi satu sama lain secara beraturan. Dengan arsitektur ini maka tercapai satu tujuan utama tertentu dalam pengembangan software (Putra, R. A., 2019). Terdapat beberapa kelebihan dari arsitektur microservice, yaitu:

1) Arsitektur microservice membuat kode aplikasi lebih sedikit dan bersifat independen sehingga dapat dilakukan pengujian aplikasi secara independent.
2) Memudahkan pemeliharaan perangkat lunak.
3) Dapat melakukan proses distrubusi perangkat lunak secara independen.

4) Mudahnya dalam melakukan scalability.
5) Developer dapat bebas dalam mengembangkan aplikasi dengan berbagai bahasa pemrograman dan framework.

2.2.10 RESTfull API

Representational State Transfer (REST) merupakan arsitektur perangkat lunak yang biasanya digunakan pada pembuatan web service. RESTful API adalah sebuah antarmuka yang dibuat menerapkan prinsip REST. RESTful API menggunakan protokol HTTP untuk melakukan komunikasi antara sistem server dan klien. Alur komunikasi antara klien dan server pada RESTful API diawali dengan klien melakukan HTTP request ke server RESTful, server RESTful lalu akan memproses request tersebut dan memberi resource dalam bentuk JavaScript Object Notation (JSON) (Yudha, A. M., & Cahyono, A. B., 2022). Restful API dapat diakses dari berbagai perangkat dan mudah untuk dikembangkan (Paramitha, I. A. K. P., Wiharta, D. M., & Arsa, I. M., 2022).