BAB 1
USULAN GAGASAN

1.1 Deskripsi Umum Masalah

Seiring dengan perkembangan teknologi informasi, kebutuhan akan pengelolaan jaringan yang efisien dan fleksibel semakin meningkat. Di tengah tantangan tersebut, arsitektur jaringan monolitik telah muncul sebagai solusi yang sangat potensial. Arsitektur monolitik memungkinkan penempatan semua fungsi dan fitur secara terpusat dalam satu aplikasi tunggal yang besar. Penggunaan arsitektur monolitik umumnya digunakan sebagai otak dari seluruh jaringan. Namun, pendekatan ini memiliki beberapa kelemahan yang signifikan. Salah satunya adalah ketergantungan pada satu titik kegagalan tunggal yang dapat menyebabkan kerentanan kinerja yang buruk dalam skala yang besar. Selain itu, perubahan dalam konfigurasi atau kebijakan jaringan sering kali memerlukan pemeliharaan atau peningkatan pada keseluruhan sistem, yang dapat menjadi proses yang rumit dan berisiko [1].

Untuk mengatasi kelemahan tersebut, pendekatan baru telah muncul dengan menggunakan model arsitektur mikroservis. Dengan model ini, fungsi layanan yang sebelumnya terkonsentrasi dalam satu kontrol layanan yang besar, dipisahkan menjadi sejumlah layanan kecil yang independen [2]. Setiap mikroservis bertanggung jawab atas tugas-tugas spesifik dalam pengelolaan jaringan, seperti manajemen layanan atau aliran data. Penggunaan arsitektur mikroservis membawa beberapa manfaat yang signifikan. Pertama, memecah fungsi kontrol menjadi layanan kecil yang dapat mengurangi dampak dari kegagalan tunggal, meningkatkan keandalan dan ketahanan jaringan secara keseluruhan. Kedua, memungkinkan perubahan dan peningkatan pada satu bagian dari sistem tanpa mempengaruhi keseluruhan jaringan, mempercepat waktu penerapan dan mengurangi resiko kesalahan. Selain itu, dengan skala yang lebih kecil dan independen, mikroservis memfasilitasi penyesuaian dan skalabilitas yang lebih baik dalam lingkungan jaringan yang berubah dengan cepat [3].

Dengan demikian, transisi dari arsitektur monolitik ke model arsitektur mikroservis menawarkan solusi yang lebih efisien, handal, dan fleksibel untuk pengelolaan jaringan modern. Dengan mengadopsi arsitektur mikroservis penelitian ini melakukan uji coba dengan penerapan arsitektur mikroservis dalam jaringan *internet of things* monitoring kolam budidaya ikan, sehingga jaringan dapat lebih responsif terhadap perubahan lingkungan kolam ikan, lebih mudah diatur, dan lebih andal dalam menghadapi tantangan yang semakin kompleks dalam dunia jaringan yang terus berkembang. Penelitian ini bertujuan untuk mengembangkan sistem menggunakan arsitektur

mikroservis. Penelitian ini akan dilakukan di PT Helmi Farm Mandiri, sebuah kawasan pengembangan budidaya ikan di Universitas Telkom Bandung, Jawa Barat. Langkah ini tidak hanya memperbaiki kinerja jaringan, tetapi juga meningkatkan skalabilitas dan efisiensi operasional secara keseluruhan, menghadirkan infrastruktur jaringan yang siap menghadapi tantangan masa depan [4].

1.2 Analisa Masalah

Arsitektur monolitik telah menjadi perbincangan hangat dalam industri teknologi informasi. Teknologi ini memungkinkan kita untuk mengelola jaringan dengan lebih efisien dan efektif, namun juga menimbulkan masalah di beberapa aspek seperti manufakturabilitas, keberlanjutan, dan reliabilitas. Oleh karena itu, kami akan membahas analisa masalah dalam empat aspek tersebut, dan bagaimana arsitektur mikroservis dapat menjadi solusi dari masalah tersebut.

1.2.1 Aspek Manufakturabilitas

Dalam arsitektur monolitik, satu komponen besar menangani semua fungsi dan logika pengelolaan jaringan. Jika terjadi perubahan atau perbaikan pada komponen tersebut, seperti pembaruan perangkat lunak atau konfigurasi, hal tersebut dapat mempengaruhi keseluruhan sistem. Risiko kesalahan atau *downtime* produksi meningkat karena seluruh sistem harus dihentikan sementara selama proses perubahan atau perbaikan berlangsung. Dalam konteks budidaya ikan, *downtime* produksi dapat menyebabkan gangguan signifikan pada proses manufakturing atau pengolahan ikan. Jika sistem monitoring tidak berfungsi selama *downtime*, informasi penting seperti kondisi lingkungan dapat hilang atau tidak akurat. Hal ini dapat mengakibatkan kesalahan dalam pengambilan keputusan terkait pengelolaan budidaya ikan, yang pada akhirnya dapat berdampak pada kualitas dan kuantitas hasil panen.

1.2.2 Aspek Keberlanjutan *(Sustainability)*

Ketika jumlah perangkat IoT yang terhubung untuk monitoring meningkat, penggunaan arsitektur monolitik dapat mengalami kinerja yang buruk atau bahkan kegagalan karena tidak mampu menangani skala yang lebih besar. Arsitektur monolitik memiliki keterbatasan dalam skalabilitas dan kemampuan untuk mengelola peningkatan jumlah perangkat IoT secara berkelanjutan. Masalah ini dapat berdampak signifikan pada keberlanjutan budidaya ikan. Ketika fungsi layanan tidak dapat menangani peningkatan jumlah perangkat IoT dengan baik, kualitas

pemantauan dan pengumpulan data akan terganggu. Informasi yang tidak akurat atau tidak lengkap dapat menyebabkan kesalahan dalam pengambilan keputusan, yang pada akhirnya dapat mengakibatkan praktik pengelolaan sumber daya yang tidak berkelanjutan.

1.2.3 Aspek Reliabilitas

Dalam arsitektur monolitik, semua fungsi dan logika pengelolaan jaringan terpusat pada satu komponen layanan. Reliabilitas sistem ini cenderung tinggi karena tidak melibatkan banyak komponen terpisah yang harus berkomunikasi satu sama lain, yang dapat mengurangi potensi kegagalan komunikasi antar komponen. Namun, kelemahan utamanya adalah ketergantungan pada satu titik kegagalan. Jika komponen ini mengalami kegagalan, seluruh sistem akan terhenti, yang berdampak signifikan terutama dalam operasi budidaya ikan. Kerusakan atau gangguan pada sistem monitoring dapat menyebabkan hilangnya data penting seperti kondisi lingkungan. Hal ini dapat mengakibatkan pengambilan keputusan yang salah dalam pengelolaan budidaya ikan.

1.3 Analisa Solusi yang Ada

Dalam melakukan analisis terhadap solusi yang sudah tersedia, penelitian ini melakukan studi literatur dari beberapa jurnal yang telah menginvestigasi penerapan solusi terhadap masalah yang timbul dalam arsitektur monolitik. Penggunaan multi kontroler, *network slicing*, dan mikroservis memiliki potensi besar untuk menyediakan solusi yang efektif terhadap masalah yang terjadi dalam arsitektur monolitik.

1.3.1 Multi Kontroler

Pendekatan multi kontroler dalam pengelolaan sistem menawarkan beberapa kelebihan, termasuk kemampuan untuk mendistribusikan beban kerja secara lebih merata dan meningkatkan redundansi serta ketahanan terhadap kegagalan. Dengan memiliki beberapa kontroler yang bekerja secara bersamaan, sistem dapat lebih responsif dan adaptif terhadap perubahan kondisi operasional. Namun, multikontroler juga memiliki kekurangan, seperti kompleksitas yang lebih tinggi dalam hal sinkronisasi dan komunikasi antar kontroler, yang bisa menyebabkan peningkatan latensi dan potensi konflik data. Selain itu, biaya implementasi dan pemeliharaan yang lebih besar menjadi tantangan signifikan.

1.3.2 *Network Slicing*

Penggunaan *network slicing* memungkinkan pembagian jaringan fisik menjadi beberapa jaringan virtual yang dapat dioptimalkan untuk berbagai aplikasi, memberikan fleksibilitas dan efisiensi dalam penggunaan sumber daya jaringan. Ini sangat berguna untuk aplikasi yang membutuhkan performa tinggi dan ketersediaan yang konsisten. Namun, tantangan utama dari *network slicing* adalah kompleksitas dalam manajemen dan keamanan, serta biaya implementasi yang tinggi. Di sisi lain, pendekatan *network slicing* memungkinkan skalabilitas dan agilitas yang lebih tinggi. Pendekatan ini mempermudah proses pembaruan, penambahan fitur, dan perbaikan tanpa mengganggu keseluruhan sistem, serta memungkinkan pengembang untuk merespons perubahan kebutuhan dengan lebih cepat dan efisien.

1.3.3 Mikroservis

Penggunaan mikroservis memungkinkan aplikasi dan layanan dalam jaringan untuk dikembangkan dan dikelola secara terpisah, memfasilitasi fleksibilitas, skalabilitas, dan efisiensi dalam pengembangan dan pengelolaan layanan [5]. Pendekatan arsitektur mikroservis memiliki beberapa kelebihan yang signifikan. Skalabilitasnya memungkinkan setiap layanan ditingkatkan atau diturunkan sesuai kebutuhan tanpa mempengaruhi keseluruhan sistem, dan fleksibilitasnya memudahkan pengembang untuk memperbarui atau memperbaiki bagian tertentu tanpa gangguan besar. Selain itu, pendekatan ini memungkinkan pengembangan dan pengujian yang lebih cepat, serta meningkatkan ketahanan terhadap kegagalan karena masalah dalam satu layanan tidak langsung merusak seluruh aplikasi. Kekurangannya termasuk kompleksitas manajemen layanan yang lebih tinggi dan kebutuhan akan alat yang baik.