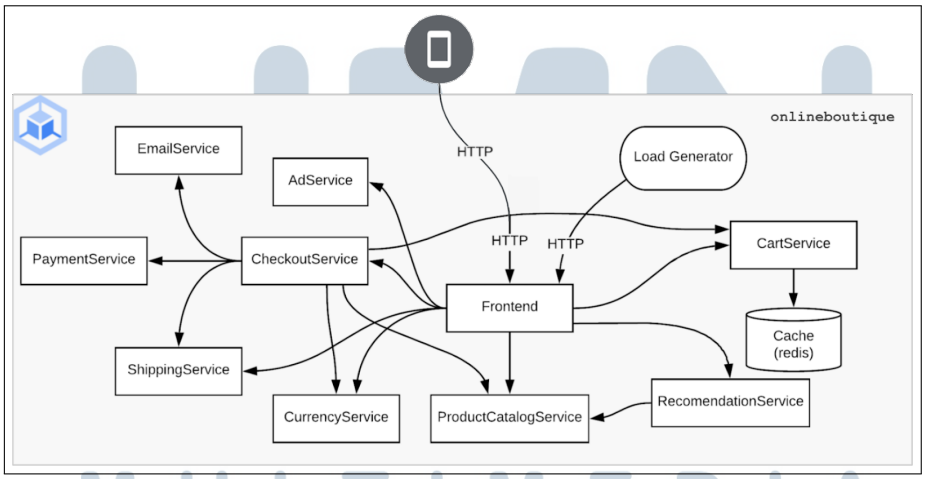
BAB 1
PENDAHULUAN

1.1 Latar Belakang Masalah

Sistem perangkat lunak modern saat ini dituntut untuk memenuhi standar operasi yang andal dengan ketersediaan yang tinggi. Tuntuan ini mendorong industri perangkat lunak untuk beralih dari arsitektur monolitik, yaitu aplikasi yang dibangun sebagai satu program besar dan utuh, menjadi arsitektur terdistribusi seperti *microservices*, yaitu aplikasi yang dipecah menjadi layanan-layanan kecil yang independen dan saling terhubung. Peralihan arsitektur ini memberikan kemudahan untuk dikembangkan, diluncurkan, dan diskalakan sehingga meningkatkan fleksibilitas dan skalabilitas operasional [1]. Namun, arsitektur ini tidak menghilangkan kompleksitas sistem, melainkan hanya memindahkan letak kompleksitas tersebut. Pada arsitektur monolitik, hubungan antarbagian program ditangani secara otomatis dalam satu proses yang sama. Pada arsitektur *microservices*, hubungan tersebut berubah menjadi komunikasi antarlayanan melalui jaringan, sehingga pengelolaan sistem dan pemantauan hubungan antarlayanan menjadi tantangan operasional.



Gambar 1.1. Ilustrasi arsitektur Google Online Boutique Shop
Sumber: [2]

Kompleksitas arsitektur *microservices* yang tinggi menciptakan tanggung jawab yang besar bagi tim *Site Reliability Engineering* (SRE), yaitu tim yang berperan menjaga sistem tetap berjalan secara stabil dan andal. Tim ini bertugas memastikan layanan tetap dapat diakses, responsif, dan berjalan sesuai standar yang

telah ditentukan [3]. Dalam ekosistem terdistribusi, banyak layanan yang saling terhubung dan bergantung satu sama lain, gangguan kecil pada satu layanan dapat memicu efek berantai yang merambat dengan cepat dan berdampak luas sebelum terdeteksi [4]. Kondisi ini pernah terekam pada pengujian injeksi kegagalan terhadap Google Online Boutique Shop yang arsitekturnya dapat dilihat pada gambar 1.1.

Kasus nyata dari dataset evaluasi RCAEval pada aplikasi Online Boutique Shop menggambarkan efek berantai tersebut. Sebuah gangguan disuntikkan pada layanan CurrencyService hingga menyebabkan penurunan performa *disk* secara signifikan. Gangguan ini pertama kali muncul pada metrik internal CurrencyService itu sendiri, yaitu penggunaan CPU dan latensinya. Dampaknya merambat ke latensi CheckoutService dan CartService yang bergantung padanya, bahkan dampak terlihat lebih besar dibanding gangguan aslinya [5]. Kejadian ini menggambarkan bagaimana gejala yang paling terlihat dapat menyesatkan diagnosis, sementara akar masalah sebenarnya tersembunyi jauh di dalam arsitektur sistem. Untuk mengatasi resiko ini, tim SRE mengandalkan *Root Cause Analysis* atau RCA, yaitu proses menganalisis data operasional sistem untuk mengidentifikasi penyebab utama suatu gangguan, sehingga pemulihan dapat dilakukan sebelum kerusakan sistem meluas [6].

Kebutuhan akan RCA yang cepat dan akurat pada kenyataannya sering terhambat oleh sistem pemantauan yang tidak terintegrasi. Ketika satu layanan mengalami gangguan, dampaknya dapat merambat dengan cepat ke layanan-layanan lain yang saling bergantung, sehingga gejala kegagalan yang paling terlihat sering muncul jauh dari sumber masalah sebenarnya [7]. Kondisi ini memaksa praktisi SRE untuk menelusuri data secara manual dari berbagai sumber pemantauan terpisah, sebuah proses yang tanpa bantuan alat otomatis dapat memakan waktu berjam-jam untuk satu insiden [8]. Permasalahan ini semakin signifikan pada sistem terdistribusi berskala industri dengan ribuan layanan mikro, di mana volume data telemetri yang dihasilkan dapat melampaui sepuluh *terabyte* per hari [9]. Oleh karena itu, diperlukan korelasi antar sumber data yang berbeda untuk melengkapi kekurangan dari masing-masing instrumen pemantauan.

Mengorelasikan data telemetri dari berbagai sumber yang berbeda umumnya dilakukan dengan bantuan model *machine learning* (pembelajaran mesin), yaitu sistem yang dilatih mengenali pola dari data yang telah dipelajari untuk memprediksi penyebab suatu gangguan. Survei terkini menunjukkan bahwa pendekatan *machine learning* tersebut sangat bergantung pada data berskala

besar, sehingga merancang model yang efektif menjadi tantangan tersendiri [10]. Namun, satu data telemetri biasanya membawa banyak ukuran sekaligus, seperti penggunaan CPU, memori, dan waktu respons yang semuanya tercatat bersamaan. Jika seluruh variabel tersebut dimasukkan langsung ke dalam model tanpa seleksi yang tepat, tumpukan informasi yang berlebih berpotensi membebani komputasi dan mengaburkan hasil diagnosis. Tanpa proses memilah variabel mana yang paling berpengaruh terhadap hasil, model menjadi sulit dipahami cara kerjanya, sehingga operator atau praktisi SRE kehilangan panduan yang jelas dalam mengidentifikasi akar masalah [11].

Berbagai penelitian terbaru sudah mencoba menjawab kebutuhan interpretasi ini dengan menggabungkan beberapa jenis data telemetri sekaligus, namun beberapa kelemahan mendasar masih belum terpecahkan. Mayoritas penelitian hanya mengevaluasi kontribusi data berdasarkan akurasi keseluruhan, tanpa menganalisis hingga ke tingkat fitur [12, 13]. Kerangka kerja terbaru juga cenderung mengandalkan arsitektur kecerdasan buatan yang kompleks untuk memaksimalkan akurasi prediksi [9, 14]. Sifat *blackbox* pada arsitektur ini menutupi cara kerja internalnya, sehingga mengorbankan transparansi pada tingkat fitur. Walaupun beberapa model menjelaskan hubungan sebab akibat antarlayanan, pendekatan ini belum mampu mengungkap fitur spesifik penentu akar masalah.

Berdasarkan celah tersebut, penelitian ini memanfaatkan data telemetri untuk mengidentifikasi layanan yang menjadi akar masalah gangguan pada sistem *microservices*. Data telemetri yang digunakan meliputi metrik berupa ukuran numerik sistem seperti CPU dan memori, serta *traces* yang memetakan alur permintaan antarlayanan. Data ini direpresentasikan dalam bentuk data tabular yang digunakan sebagai *input* model klasifikasi XGBoost, dengan layanan sumber masalah (*root cause service*) sebagai kelas target. Penggunaan XGBoost didukung oleh temuan dari berbagai penelitian yang menunjukkan kinerjanya pada data tabular yang melampaui model *deep learning* yang lebih kompleks [15, 16]. Penelitian ini mengintegrasikan SHAP untuk menjawab kebutuhan interpretabilitas tingkat fitur tersebut. Melalui SHAP, kontribusi setiap fitur terhadap hasil prediksi dapat diukur sehingga metrik maupun karakteristik *traces* yang memengaruhi proses identifikasi layanan *root cause* dapat dianalisis lebih jelas [17]. Pada model klasifikasi berbasis pohon seperti XGBoost, nilai SHAP dapat dihitung secara efisien melalui integrasi algoritma TreeSHAP [18].

Penelitian ini merancang tiga model klasifikasi XGBoost, yaitu dua model unimodal yang masing-masing menggunakan data metrik dan *traces*

secara terpisah, serta satu model multimodal yang menggabungkan keduanya, untuk membandingkan pengaruh tiap jenis data terhadap kinerja identifikasi akar masalah. Analisis SHAP kemudian diterapkan pada model multimodal untuk mengidentifikasi fitur-fitur paling berpengaruh serta membandingkan kontribusi relatif fitur metrik dan traces pada berbagai skenario kegagalan. Model multimodal selanjutnya dilatih ulang menggunakan subset fitur dengan kontribusi SHAP tertinggi, untuk mengevaluasi sejauh mana reduksi jumlah fitur memengaruhi performa klasifikasi pada masing-masing arsitektur. Hasil ini diharapkan memberikan gambaran yang lebih jelas mengenai peran tiap jenis data telemetri dalam diagnosis layanan yang mengalami gangguan, serta sejauh mana seleksi fitur berbasis SHAP dapat menyederhanakan ruang fitur tanpa mengorbankan akurasi.

1.2 Rumusan Masalah

Berdasarkan pemaparan celah penelitian pada latar belakang masalah di atas, rumusan masalah dalam skripsi ini difokuskan pada dua pertanyaan berikut.

1. Bagaimana merancang model klasifikasi XGBoost yang memanfaatkan data metrik dan *distributed traces* sebagai masukan untuk identifikasi layanan *root cause* pada sistem *microservices*?
2. Bagaimana perbandingan tingkat akurasi pelokalan layanan *root cause* pada model XGBoost ketika dilatih menggunakan data metrik secara terisolasi, data *traces* secara terisolasi, dan gabungan telemetri multimodal?
3. Bagaimana perbandingan proporsi kontribusi fitur metrik dan *traces* dalam mengidentifikasi layanan *root cause* berdasarkan analisis atribusi fitur menggunakan SHAP pada arsitektur *microservices*?
4. Bagaimana dampak reduksi dimensi fitur berdasarkan tipe sinyal dengan kontribusi SHAP tertinggi terhadap performa klasifikasi model XGBoost pada masing-masing arsitektur *microservices*?

1.3 Batasan Permasalahan

Agar ruang lingkup penelitian tetap terfokus dan selaras dengan latar belakang serta rumusan masalah yang telah ditetapkan, batasan masalah yang akan digunakan dalam penelitian ini diuraikan sebagai berikut.

1. Penelitian ini dibatasi pada dataset RCAEval RE2 yang mencakup dua arsitektur *microservices*, yaitu Google Online Boutique Shop dan Train Ticket, masing-masing terdiri dari 30 skenario kegagalan dengan 3 repetisi per skenario. Dataset ini merupakan hasil simulasi injeksi kegagalan terkontrol dan tidak mencakup data telemetri dari lingkungan produksi nyata.

2. Modalitas data telemetri yang digunakan dibatasi pada metrik sistem dan *distributed traces*. Data log tidak diikutsertakan sebagai sumber data dalam penelitian ini.

3. Target identifikasi dibatasi pada penentuan layanan yang menjadi *root cause* dari gangguan sistem (*faulty service localization*). Klasifikasi tipe kegagalan (*fault type classification*) tidak termasuk dalam ruang lingkup penelitian ini.

4. Evaluasi model dibatasi pada pengujian *offline* dengan skenario kegagalan tunggal (*single fault*). Skenario dengan kegagalan simultan pada lebih dari satu layanan tidak tercakup dalam penelitian ini.

5. Ekstraksi fitur telemetri dibatasi pada jendela waktu tetap sebelum dan sesudah waktu injeksi kegagalan (*inject_time*). Sensitivitas model terhadap variasi ukuran jendela waktu tidak dievaluasi dalam penelitian ini.

6. Penelitian ini tidak mencakup inferensi *real-time* maupun penerapan sistem diagnosis pada lingkungan produksi aktif.

7. Pengujian reduksi fitur tidak mencakup penyetelan ulang hiperparameter, sehingga hasil mencerminkan dampak murni reduksi fitur, bukan performa optimal model dengan ruang fitur tereduksi.

1.4 Tujuan Penelitian

Sejalan dengan rumusan masalah yang telah ditetapkan, tujuan dari penelitian ini adalah sebagai berikut.

1. Merancang model klasifikasi XGBoost yang memanfaatkan data metrik dan *distributed traces* sebagai masukan untuk identifikasi layanan sumber masalah pada sistem *microservices*.

2. Mengevaluasi dan membandingkan secara kuantitatif tingkat akurasi pelokalan layanan *root cause* dari model XGBoost yang dilatih menggunakan

data metrik secara terisolasi, data *traces* secara terisolasi, dan gabungan telemetri multimodal.

3. Menganalisis proporsi kontribusi fitur metrik dan *traces* dalam mengidentifikasi layanan *root cause* menggunakan metode atribusi fitur SHAP pada masing-masing arsitektur *microservices*.

4. Mengevaluasi dampak reduksi dimensi fitur berdasarkan tipe sinyal dengan kontribusi SHAP tertinggi terhadap performa klasifikasi model XGBoost pada masing-masing arsitektur *microservices*.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan kontribusi dan manfaat nyata, baik dari sudut pandang teoritis akademis maupun aplikasi praktis di industri, yang dijabarkan pada poin-poin berikut.

1. **Manfaat Akademis**
   Penelitian ini memberikan bukti empiris mengenai pengaruh penggabungan data metrik dan *traces* terhadap performa model klasifikasi berbasis pohon, sekaligus memperkaya literatur *Root Cause Analysis* (RCA) pada sistem *microservices* dengan pendekatan yang mampu mengukur kontribusi tiap jenis data secara kuantitatif. Penerapan SHAP *TreeExplainer* pada kasus ini juga dapat menjadi referensi metodologis bagi penelitian serupa di masa mendatang.

2. **Manfaat Praktis**
   Penelitian ini menghasilkan pendekatan klasifikasi akar masalah yang ringan dan efisien secara komputasi, sehingga berpotensi mudah diadopsi oleh tim *DevOps* maupun *Site Reliability Engineering* (SRE). Penjelasan kontribusi fitur dari SHAP berpotensi membantu teknisi menelusuri sumber masalah secara lebih terarah, yang pada akhirnya dapat mendukung penurunan waktu perbaikan (*Mean Time to Resolution*/MTTR) dan stabilitas layanan *microservices*.

1.6 Sistematika Penulisan

Sistematika penulisan laporan adalah sebagai berikut:

- Bab 1 PENDAHULUAN

  Bab ini menguraikan latar belakang masalah terkait tantangan melokalisasi *root cause* pada arsitektur *microservices*, rumusan masalah, batasan permasalahan, tujuan penelitian, manfaat penelitian secara akademis dan praktis, serta sistematika penulisan laporan.

- Bab 2 LANDASAN TEORI

  Bab ini berisi penjelasan mengenai konsep dan teori yang mendasari penelitian, meliputi arsitektur *microservices* dan observabilitas (metrik serta *traces*), rekayasa fitur data telemetri, algoritma ansambel berbasis pohon khususnya XGBoost, metode optimasi model (Optuna dan *Cross Validation*), metrik evaluasi, serta konsep *Explainable AI* menggunakan SHAP.

- Bab 3 METODOLOGI PENELITIAN

  Bab ini menjelaskan tahapan sistematis penelitian, mulai dari pengumpulan dataset RCAEval, pra-pemrosesan data (reduksi *noise* dan penyaringan varians nol), ekstraksi fitur delta, perancangan kondisi eksperimen (unimodal metrik, unimodal *traces*, dan multimodal), proses optimasi dan pelatihan model XGBoost, hingga rancangan analisis interpretasi menggunakan metode SHAP.

- Bab 4 HASIL DAN DISKUSI

  Bab ini memaparkan hasil evaluasi dari implementasi yang telah dilakukan, mencakup perbandingan tingkat akurasi dan *Macro F1-Score* model XGBoost pada ketiga kondisi eksperimen. Bab ini juga memuat diskusi mendalam terkait analisis *feature importance* berbasis bobot serta interpretasi SHAP untuk mengukur proporsi kontribusi modalitas dan tipe sinyal terhadap prediksi *root cause*.

- Bab 5 SIMPULAN DAN SARAN

  Bab ini berisi simpulan akhir yang menjawab rumusan masalah berdasarkan hasil pengujian performa model dan analisis interpretasi yang telah dipaparkan, serta saran-saran konstruktif untuk pengembangan riset terkait *Root Cause Analysis* (RCA) di masa mendatang.