BAB 1
PENDAHULUAN

1.1 Latar Belakang Masalah

Dalam era perkembangan teknologi yang sangat pesat, aplikasi pendeteksi penyakit mata pada hewan ternak menjadi krusial untuk menjaga kesehatan hewan ternak. Contoh nyata adalah aplikasi Ternaku, sebuah aplikasi *mobile* pendeteksi penyakit mata *pink eye* pada hewan ternak yang menggunakan *Python* sebagai *backend* dalam aplikasinya [1]. Meskipun demikian, seiring dengan pertumbuhan kompleksitas dan kebutuhan kinerja yang lebih baik, penulis mengidentifikasi beberapa kendala dalam *backend* pada aplikasi berbasis *Python*. Salah satu keterbatasan yang perlu diperhatikan adalah kinerjanya yang relatif lambat karena sifat *interpretatif Python* [2], khususnya dalam aplikasi yang menuntut kinerja tinggi atau pemrosesan *real-time*. Kelemahan ini mendorong penulis untuk melakukan evaluasi teknologi dan mempertimbangkan implementasi *Node.js* sebagai alternatif yang lebih responif sebagai *backend*.

Node.js seringkali lebih direkomendasikan untuk pengembangan *Api backend* dibandingkan dengan *Python* karena beberapa alasan. Pertama, *Node.js* terkenal karena kinerjanya yang tinggi dan kemampuannya dalam hal skalabilitas [3]. Dibangun di atas mesin *JavaScript V8, Node.js* memungkinkan eksekusi kode yang efisien dan cepat. Karakteristik ini menjadikannya *platform* yang sangat sesuai untuk menangani jumlah permintaan yang besar dan melaksanakan tugas-tugas dengan tingkat pemrosesan data yang tinggi. Selanjutnya, dengan kekuatan *non-blocking I/O* dan pendekatan *event-driven*, menawarkan potensi peningkatan kinerja yang signifikan dalam situasi di mana responsivitas waktu nyata diperlukan [4] [5] [6]. Pilihan ini memungkinkan pengoptimalan proses *backend* untuk meningkatkan efisiensi dan kecepatan dalam pendeteksian penyakit mata. Dengan demikian, penulis mengambil langkah untuk mengintegrasikan *Node.js* sebagai bagian terpenting dari *backend* dalam aplikasi, mengantisipasi manfaat positif yang dapat diakses melalui teknologi ini.

Disamping perubahasan *backend* dari *Node.js*, Adopsi arsitektur *REST* (*Representational State Transfer*) juga menjadi bagian dari implementasi ini. Arsitektur *REST* (*Representational State Transfer*) yang merupakan gaya arsitektur perangkat lunak dengan sistem terdistribusi [7], bertujuan untuk memperbaiki struktur aplikasi secara keseluruhan. Keputusan ini didasarkan pada keinginan untuk meningkatkan interoperabilitas [8], mempermudah pengembangan dan pemeliharaan, serta memberikan pemisahan antarmuka pengguna dan logika server [9]. Arsitektur *RESTful* ini diharapkan dapat memberikan fleksibilitas yang diperlukan untuk mengelola berbagai permintaan yang mungkin muncul dalam konteks aplikasi deteksi penyakit mata pada hewan ternak [10].

Meskipun proses implementasi ini tidak datang tanpa tantangan, seperti perubahan paradigma pemrograman dan integrasi dengan komponen eksisting, penulis yakin bahwa keuntungan jangka panjang dari penggunaan *Node.js* dan arsitektur *RESTful* akan melampaui tantangan tersebut. Harapannya, implementasi *Node.js* dengan arsitektur *REST* ini akan menghasilkan aplikasi baru Ternakami yang memberikan peningkatan signifikan dalam kinerja aplikasi, responivitas waktu nyata, serta memfasilitasi pemeliharaan dan pengembangan masa depan. Keputusan ini tidak hanya terkait dengan perkembangan teknologi, tetapi juga terkait dengan dampak positif yang dapat diberikan kepada sektor kesehatan hewan ternak secara keseluruhan, terutama dalam penanganan masalah *pink eye*.

Dalam konteks penelitian ini, tujuan utama adalah memberikan solusi yang lebih responsif dan efisien dalam deteksi penyakit mata pada hewan ternak, khususnya penyakit *pink eye*, melalui implementasi *backend Node.js* dan adopsi arsitektur *RESTful*. Tinjauan pustaka telah mengidentifikasi kelemahan kinerja dalam *backend* berbasis *Python*, terutama dalam menghadapi tuntutan aplikasi yang memerlukan kinerja tinggi dan pemrosesan *real-time*. Dengan memilih *Node.js*, penelitian ini bertujuan untuk meningkatkan kecepatan eksekusi dan responsivitas waktu nyata yang menjadi kunci dalam deteksi dini penyakit mata pada hewan ternak.

Selain itu, adopsi arsitektur *RESTful* diarahkan untuk meningkatkan struktur keseluruhan aplikasi, memudahkan pengembangan, pemeliharaan, dan memastikan interoperabilitas yang lebih baik. Dengan merinci tujuan ini berdasarkan tinjauan pustaka, penelitian ini diarahkan untuk memberikan kontribusi positif dalam peningkatan efisiensi serta kemudahan pengembangan dan pemeliharaan aplikasi deteksi penyakit mata. Harapannya, peningkatan ini tidak hanya akan mencakup aspek teknis semata, tetapi juga akan memberikan dampak positif pada sektor kesehatan hewan ternak secara menyeluruh, khususnya dalam penanganan permasalahan kesehatan mata yang signifikan seperti *pink eye*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disusun, penulis merumuskan pernyataan masalah sebagai berikut:

1. Menentukan kinerja aplikasi Ternakami setelah implementasi backend Node.js, dengan fokus pada analisis kecepatan respon, throughput, dan konsistensi kinerja.
2. Mengidentifikasi metode pengujian performa dengan JMeter dan menganalisis pengaruhnya terhadap stabilitas dan kehandalan aplikasi Ternakami, serta implikasinya terhadap pengembangan aplikasi deteksi penyakit mata pada hewan ternak.

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah didefinisikan, maka tujuan dari penelitian ini adalah sebagai berikut:

1. Menentukan kinerja aplikasi Ternakami setelah implementasi backend menggunakan Node.js dengan arsitektur REST, termasuk analisis terhadap kecepatan respon, throughput, dan konsistensi kinerja.
2. Mengidentifikasi dan mengevaluasi metode pengujian performa menggunakan JMeter, serta menganalisis pengaruhnya terhadap stabilitas dan kehandalan aplikasi Ternakami, khususnya dalam konteks pengembangan aplikasi deteksi penyakit mata pada hewan ternak.

1.4 Batasan Masalah

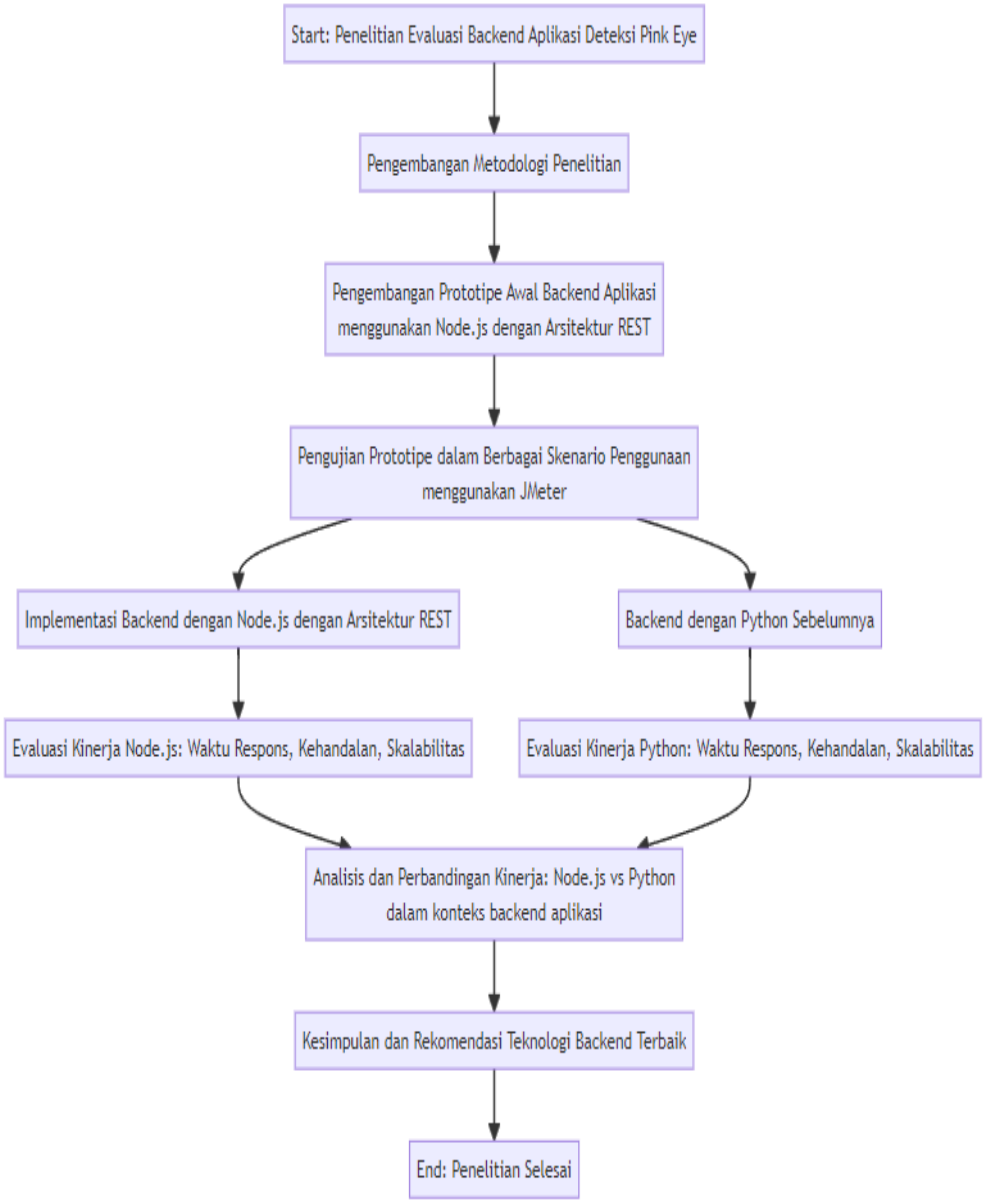
Berikut ini adalah batasan masalah dari penelitian yang dilakukan oleh penulis:

1. Lingkup Aplikasi:
- Penelitian terfokus pada *backend* aplikasi Ternakami untuk mendeteksi penyakit mata *Pink Eye* pada hewan ternak.
- Evaluasi membandingkan kinerja *backend* menggunakan *Node.js* (dengan arsitektur *REST*) dan *backend Python*.
2. Perbandingan Performa:
- Analisis kinerja membandingkan *backend Node.js* dan *Python* dalam menangani permintaan deteksi penyakit mata pada hewan ternak.
- Parameter kinerja dievaluasi melibatkan waktu respon, kehandalan, dan skalabilitas.
3. Responsivitas Waktu Nyata:
- Evaluasi memfokuskan responivitas waktu nyata *backend Node.js* dalam pendeteksian penyakit mata.
4. Skalabilitas Aplikasi:
- Analisis skalabilitas mengevaluasi kemampuan *backend Node.js* dalam mendukung pertumbuhan kompleksitas aplikasi deteksi penyakit mata pada hewan ternak.
- Evaluasi kemampuan sistem menangani beban kerja yang lebih besar.
5. Integrasi dengan Arsitektur *REST*:
- Penerapan arsitektur *REST* menjadi fokus utama evaluasi, dengan peningkatan interoperabilitas, pemeliharaan, dan pengembangan aplikasi.

1.5 Metode Penelitian

Penulis menerapkan metodologi yang menggabungkan teknik kuantitatif, eksperimental, dan *prototyping internal*. Pendekatan ini memungkinkan penulis untuk mengukur secara objektif berbagai parameter kinerja, termasuk waktu respon, kehandalan, dan skalabilitas. Fokus utama penelitian ini adalah mengimplementasi *backend Node.js* dengan arsitektur *RESTful*, dan mengevaluasi dengan *backend Python* yang ada sebelumnya.

Penelitian ini diinisiasi dengan fokus pada evaluasi kinerja *backend* aplikasi deteksi *Pink Eye*. Penelitian diawali dengan pengembangan metodologi penelitian yang akan menjadi dasar evaluasi tersebut. Tahap berikutnya adalah pembuatan *prototype* awal *backend* menggunakan *Node.js*, yang implementasikan sebagai *backend* untuk Ternakami. Untuk menilai efektivitas *prototype Node.js*, dilakukan serangkaian pengujian dalam berbagai skenario menggunakan alat seperti JMeter, untuk mengukur aspek-aspek seperti waktu respon, kehandalan, dan skalabilitas. Implementasi *Node.js* ini dibangun dengan arsitektur *RESTful*, memungkinkan sistem yang terstruktur untuk dievaluasi secara menyeluruh. Setelah pengujian, dilakukan analisis komparatif terhadap kinerja antara hasil implementasi *backend Node.js* dengan *Python* untuk menentukan mana yang lebih unggul dalam konteks aplikasi ini. Dari hasil analisis ini, penulis dapat menyimpulkan dan merekomendasikan teknologi *backend* yang terbaik, dengan pertimbangan apakah *Node.js* menyediakan keunggulan yang signifikan atas *Python* dalam pengembangan *backend* aplikasi deteksi *Pink Eye*. Penelitian ini ditutup dengan pembuatan kesimpulan dan rekomendasi yang berdasar pada evaluasi terperinci dan perbandingan kinerja antara kedua teknologi tersebut. Alur dari penelitian yang dilakukan oleh penulis dapat digambarkan seperti pada **Gambar 1.1** berikut ini.



Gambar 1.1 Alur Peneltian

1.6 Sistematika Penulisan

Tabel 1. 1 Tabel Sistematika Penulisan

BAB I	Menjabarkan tentang latar belakang penelitian, rumusan masalah, tujuan penelitian, batasan masalah, metodologi penelitian, sistematika penulisan, serta jadwal pelaksanaan berdasarkan penelitian ini.
BAB II	Bagian ini memuat menjabarkan teori pendukung serta teori yang melandasi penyusunan tugas akhir ini
BAB III	Pada bagian ini memuat mengenai perancangan secara keseluruhan mulai dari desain arsitektur sistem, pengkodean, dan *deployment*
BAB IV	Bagian ini mengulas mengenai hasil dari penelitian seperti hasil respon dan analisa pengujian *backend Api*
BAB V	Berisi kesimpulan dan saran dari semua proses pengerjaan selama proses penelitian tugas akhir

1.7 Jadwal Pelaksanaan

Tabel 1. 2 Jadwal Pelaksanaan Penelitian

No.	**Deskripsi Tahapan**	**Tahun 2024**						
		Jan	Feb	Mar	Apr	Mei	Jun	Jul
1	Tahapan Persiapan Penelitian							
2	Tahapan Pengkodean *Backend Node.js*							
3	Uji Coba *Backend Node.js*							
4	Integrasi Kedua *Backend* ke *Google Cloud Platform*							
5	Pengujian *API* Kedua *Backend* pada Aplikasi JMeter							
6	Penyusunan Laporan							