BAB 1
PENDAHULUAN

1.1 Latar Belakang

Pada era digital yang terus berkembang, teknologi Internet of Things (IoT) telah menjadi tulang punggung dalam menciptakan sistem yang cerdas, efisien, dan responsif. Salah satu penerapan IoT yang signifikan adalah pengembangan dashboard pemantauan yang menyajikan data secara *real-time* untuk memudahkan pengguna dalam mengidentifikasi masalah, memantau kinerja sistem, dan mengambil keputusan yang lebih baik guna meningkatkan efisiensi operasional. Namun, pengembangan sistem pemantauan seperti ini menghadapi tantangan fundamental dalam hal infrastruktur jaringan, terutama ketika *server* harus menangani banyak koneksi simultan dan operasi I/O yang intensif seperti kueri basis data dengan volume data besar.

Arsitektur jaringan sinkron konvensional yang umum digunakan dalam pengembangan aplikasi web mengakibatkan *server* menjadi tidak responsif (blocking) ketika mengeksekusi operasi I/O yang memerlukan waktu lama. Menurut (Zhang et al., 2020) dalam penelitiannya yang berjudul "The Impact of Event Processing Flow on Asynchronous *Server* Efficiency", *server* dengan arsitektur sinkron mengalami degradasi kinerja signifikan pada beban konkurensi tinggi karena setiap thread harus menunggu operasi I/O selesai sebelum dapat melayani permintaan lainnya. Penelitian tersebut menunjukkan bahwa context switch yang berlebihan pada arsitektur multi-threaded dapat menyebabkan overhead CPU hingga 24% pada kondisi beban tertentu.

Di sisi lain, arsitektur jaringan asinkron berbasis event-driven menawarkan solusi untuk mengatasi masalah blocking tersebut. Twisted Python, sebagai salah satu framework event-driven yang matang, menyediakan mekanisme non-blocking I/O melalui konsep reactor pattern dan *Deferred* untuk menangani operasi yang memerlukan waktu tanpa menghentikan eksekusi program utama. Menurut dokumentasi arsitektur Twisted (Jessica McKellar, 2011), framework ini mampu menangani ribuan koneksi simultan dengan penggunaan sumber daya yang minimal dibandingkan pendekatan thread-based konvensional.

Dalam konteks IoT, protokol MQTT (Message Queuing Telemetry Transport) telah menjadi standar de facto untuk komunikasi perangkat IoT karena karakteristiknya yang lightweight dan efisien. Penelitian oleh (Saha et al., 2024) yang berjudul "Performance evaluation framework of MQTT client libraries for IoT applications in manufacturing" menunjukkan bahwa MQTT menawarkan komunikasi yang lightweight, efisien, dan reliable untuk aplikasi IoT, dengan kebutuhan akan latensi rendah dan skalabilitas tinggi.

(Aziz Fikhri et al., 2025)dalam "Perbandingan Kinerja Protokol MQTT dan HTTP Dalam Komunikasi Data Internet of Things Artikel Penelitian" juga menunjukkan bahwa MQTT unggul dalam pertukaran pesan terdistribusi pada lingkungan IoT.

Permasalahan kritis yang dihadapi adalah ketika operasi basis data yang berat dieksekusi pada main thread (reactor thread) dalam arsitektur event-driven. Operasi blocking seperti kueri SQL dengan dataset besar akan menghentikan seluruh event loop, menyebabkan *server* tidak dapat melayani permintaan lain hingga operasi selesai. Mekanisme deferToThread pada Twisted Python menyediakan solusi dengan mendelegasikan operasi blocking ke background thread pool, sehingga reactor thread tetap bebas menangani event lainnya.

Berdasarkan latar belakang tersebut, penelitian ini menjadi relevan untuk mengimplementasikan dan menguji arsitektur jaringan asinkron pada dashboard IoT *real-time* menggunakan Twisted Python, dengan fokus pada mekanisme deferToThread untuk menangani operasi basis data yang berat, serta membandingkan kinerjanya dengan pendekatan sinkron konvensional.

1.2 Perumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana cara mengimplementasikan komunikasi jaringan asinkron menggunakan *Twisted Python*?.
2. Bagaimana Menangani koneksi simultan dalam jumlah besar menggunakan protokol TCP/IP dan HTTP?.
3. Bagaimana komunikasi *non-blocking* dapat mengurangi latensi dan meningkatkan efisiensi komunikasi?

1.3 Batasan Masalah

Agar ruang lingkup penelitian tidak terlalu luas, batasan masalah yang diterapkan adalah sebagai berikut:

1. Fokus pada pemrograman jaringan asinkron menggunakan *framework Twisted Python*.
2. *Twisted Python* hanya mengimplementasikan protokol TCP/IP dan HTTP untuk komunikasi IoT.
3. Pengujian dilakukan pada simulasi perangkat IoT dalam skala kecil hingga menengah.
4. Penelitian lebih menitikberatkan pada aspek pemrograman jaringan dan komunikasi asinkron.
5. Pengembangan antarmuka atau dashboard visualisasi tidak dibahas secara mendalam.

1.4 Tujuan dan Manfaat

1.4.1 Tujuan

Penelitian ini bertujuan untuk mengimplementasikan jaringan asinkron menggunakan *framework* Twisted Python guna mendukung komunikasi IoT secara *real-time*, menganalisis dampak penggunaan komunikasi *non-blocking* dalam mengurangi latensi dan meningkatkan efisiensi jaringan, serta membandingkan kinerja *dashboard* pada implementasi komunikasi jaringan asinkron dan sinkron.

1.4.2 Manfaat

Hasil dari penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Menyediakan referensi tentang implementasi jaringan asinkron menggunakan *Twisted Python* dalam konteks IoT.
2. Memberikan wawasan tentang teknik komunikasi *non-blocking* yang dapat meningkatkan efisiensi sistem IoT.
3. Menyediakan panduan untuk mengembangkan aplikasi IoT berskala besar dengan koneksi simultan yang stabil dan responsif.
4. Mendukung terciptanya sistem IoT yang andal dan efisien dengan latensi rendah serta skalabilitas tinggi.