MITIGASI *RACE CONDITION* MENGGUNAKAN *MESSAGE BROKER* PADA *BACKEND* SISTEM PENJUALAN TIKET BERBASIS GOLANG

SKRIPSI

Diajukan untuk memenuhi salah satu syarat mencapai derajat Sarjana

Program Studi Informatika

disusun oleh

MIFTAHUDIN FAIZ

20.11.3675

Kepada

FAKULTAS ILMU KOMPUTER

UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

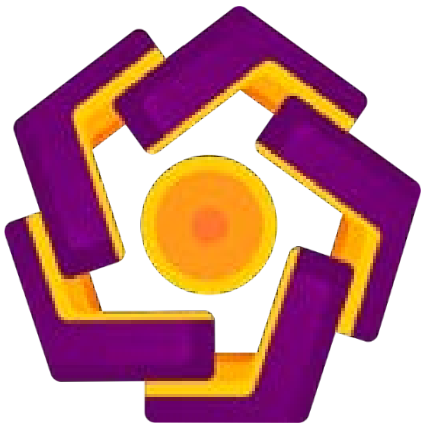
2025

MITIGASI *RACE CONDITION* MENGGUNAKAN *MESSAGE BROKER* PADA *BACKEND* SISTEM PENJUALAN TIKET BERBASIS GOLANG

SKRIPSI

untuk memenuhi salah satu syarat mencapai derajat Sarjana

Program Studi Informatika



disusun oleh

MIFTAHUDIN FAIZ

20.11.3675

Kepada

FAKULTAS ILMU KOMPUTER

UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2025

HALAMAN PERSETUJUAN

SKRIPSI

MITIGASI *RACE CONDITION* MENGGUNAKAN *MESSAGE BROKER* PADA *BACKEND* SISTEM PENJUALAN TIKET BERBASIS GOLANG

yang disusun dan diajukan oleh

Miftahudin Faiz

20.11.3675

telah disetujui oleh Dosen Pembimbing Skripsi
pada tanggal 30 Januari 2025

Dosen Pembimbing,

Ali Mustopa, M.Kom.
NIK. 190302192

HALAMAN PENGESAHAN

SKRIPSI

MITIGASI *RACE CONDITION* MENGGUNAKAN *MESSAGE BROKER* PADA *BACKEND* SISTEM PENJUALAN TIKET BERBASIS GOLANG

yang disusun dan diajukan oleh

Miftahudin Faiz

20.11.3675

Telah dipertahankan di depan Dewan Penguji
pada tanggal 30 Januari 2025

Susunan Dewan Penguji

Nama Penguji	Tanda Tangan
Sudarmawan, M. T. **NIK. 190302035**	
Novi Prisma Yunita, M. Kom. **NIK. 190302526**	
Ali Mustopa, M. Kom. **NIK. 190302192**	

Skripsi ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Sarjana Komputer
Tanggal 30 Januari 2025

DEKAN FAKULTAS ILMU KOMPUTER

Hanif Al Fatta,S.Kom., M.Kom., Ph.D.
NIK. 190302096

HALAMAN PERNYATAAN KEASLIAN SKRIPSI

Yang bertandatangan di bawah ini,

Nama mahasiswa : Miftahudin Faiz
NIM : 20.11.3675

Menyatakan bahwa Skripsi dengan judul berikut:

MITIGASI *RACE CONDITION* MENGGUNAKAN *MESSAGE BROKER* PADA *BACKEND* SISTEM PENJUALAN TIKET BERBASIS GOLANG

Dosen Pembimbing: Ali Mustopa, M. Kom

1. Karya tulis ini adalah benar-benar ASLI dan BELUM PERNAH diajukan untuk mendapatkan gelar akademik, baik di Universitas AMIKOM Yogyakarta maupun di Perguruan Tinggi lainnya.
2. Karya tulis ini merupakan gagasan, rumusan dan penelitian SAYA sendiri, tanpa bantuan pihak lain kecuali arahan dari Dosen Pembimbing.
3. Dalam karya tulis ini tidak terdapat karya atau pendapat orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan disebutkan dalam Daftar Pustaka pada karya tulis ini.
4. Perangkat lunak yang digunakan dalam penelitian ini sepenuhnya menjadi tanggung jawab SAYA, bukan tanggung jawab Universitas AMIKOM Yogyakarta.
5. Pernyataan ini SAYA buat dengan sesungguhnya, apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka SAYA bersedia menerima SANKSI AKADEMIK dengan pencabutan gelar yang sudah diperoleh, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi.

Yogyakarta, 30 Januari 2025

Yang Menyatakan,

MIFTAHUDIN FAIZ

HALAMAN PERSEMBAHAN

Dengan penuh rasa syukur, skripsi ini saya persembahkan kepada keluarga tercinta, yang selalu memberikan doa, dukungan, dan kasih sayang tanpa batas. Terima kasih atas kesabaran, semangat, serta pengorbanan yang telah diberikan selama ini. Kepada teman-teman yang selalu ada di saat suka maupun duka, terima kasih atas dukungan, motivasi, serta kebersamaan yang membuat perjalanan ini lebih bermakna. Kehadiran kalian memberikan semangat dalam menghadapi setiap tantangan. Semoga karya ini menjadi langkah awal menuju masa depan yang lebih baik dan dapat memberikan manfaat bagi banyak orang.

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Allah ﷻ atas limpahan rahmat dan berkah-Nya yang telah mempermudah serta melancarkan segala aktivitas, termasuk dalam proses pengerjaan skripsi ini. Penulis mengucapkan terima kasih yang sebesar-besarnya kepada kedua orang tua, kakak, dosen pembimbing, serta rekan-rekan yang telah memberikan dukungan selama penelitian ini berlangsung.

Penelitian ini disusun sebagai bagian dari pemenuhan syarat kelulusan mata kuliah Skripsi pada Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Amikom Yogyakarta. Proses penyelesaian penelitian ini tentunya tidak lepas dari berbagai tantangan dan hambatan, yang memerlukan usaha keras dari penulis untuk menyelesaikannya. Penulis juga menyadari bahwa penyelesaian skripsi ini tidak terlepas dari bantuan berbagai pihak, baik dalam bentuk materi maupun non-materi, sehingga penelitian ini dapat diselesaikan meskipun masih terdapat kekurangan di dalamnya.

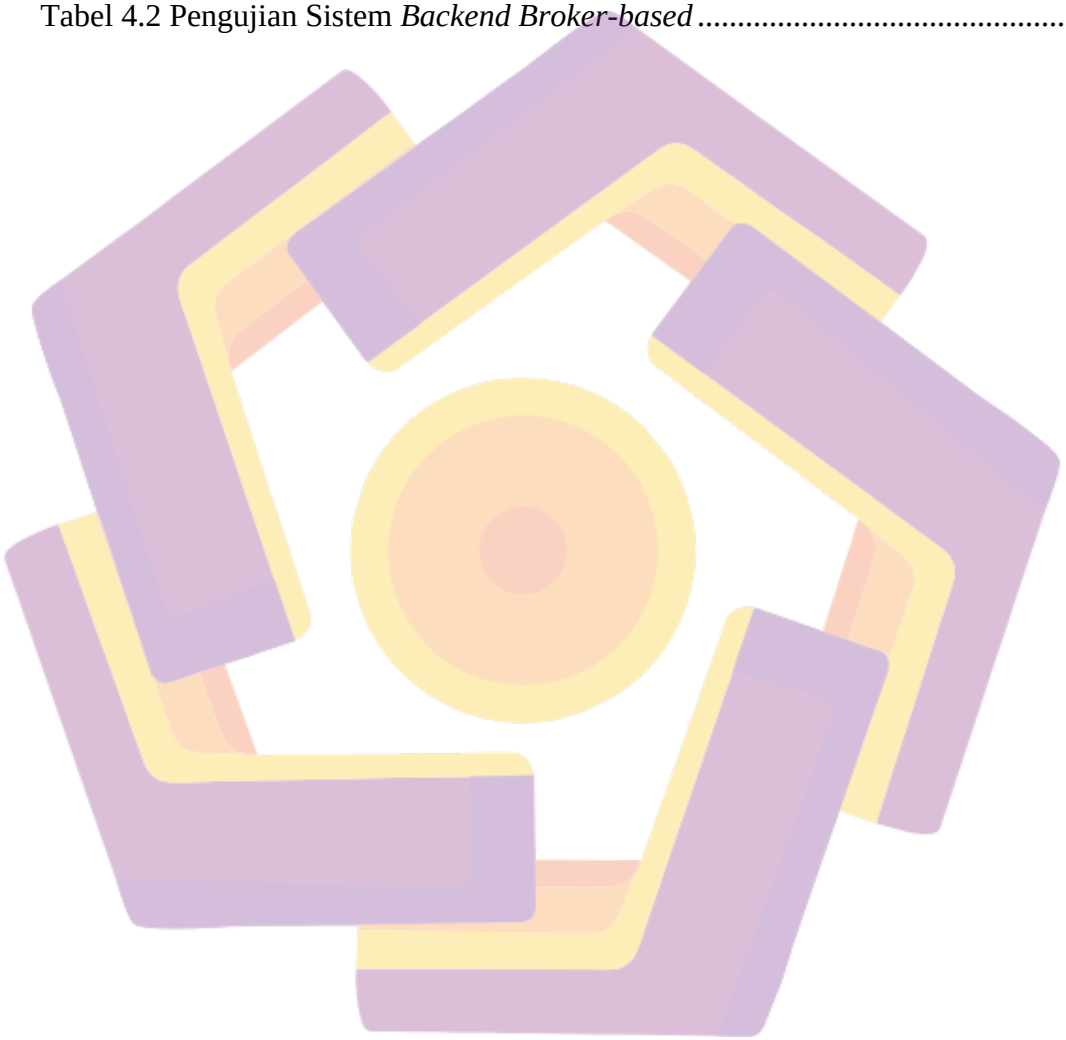
Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Oleh karena itu, penulis dengan senang hati menerima kritik dan saran yang membangun untuk perbaikan di masa mendatang. Harapan penulis, semoga skripsi ini dapat bermanfaat bagi pembaca serta memberikan kontribusi bagi perkembangan ilmu pengetahuan, khususnya dalam bidang teknologi backend dan arsitektur sistem terdistribusi.

Yogyakarta, 30 Januari 2025

Penulis

DAFTAR ISI

DAFTAR TABEL

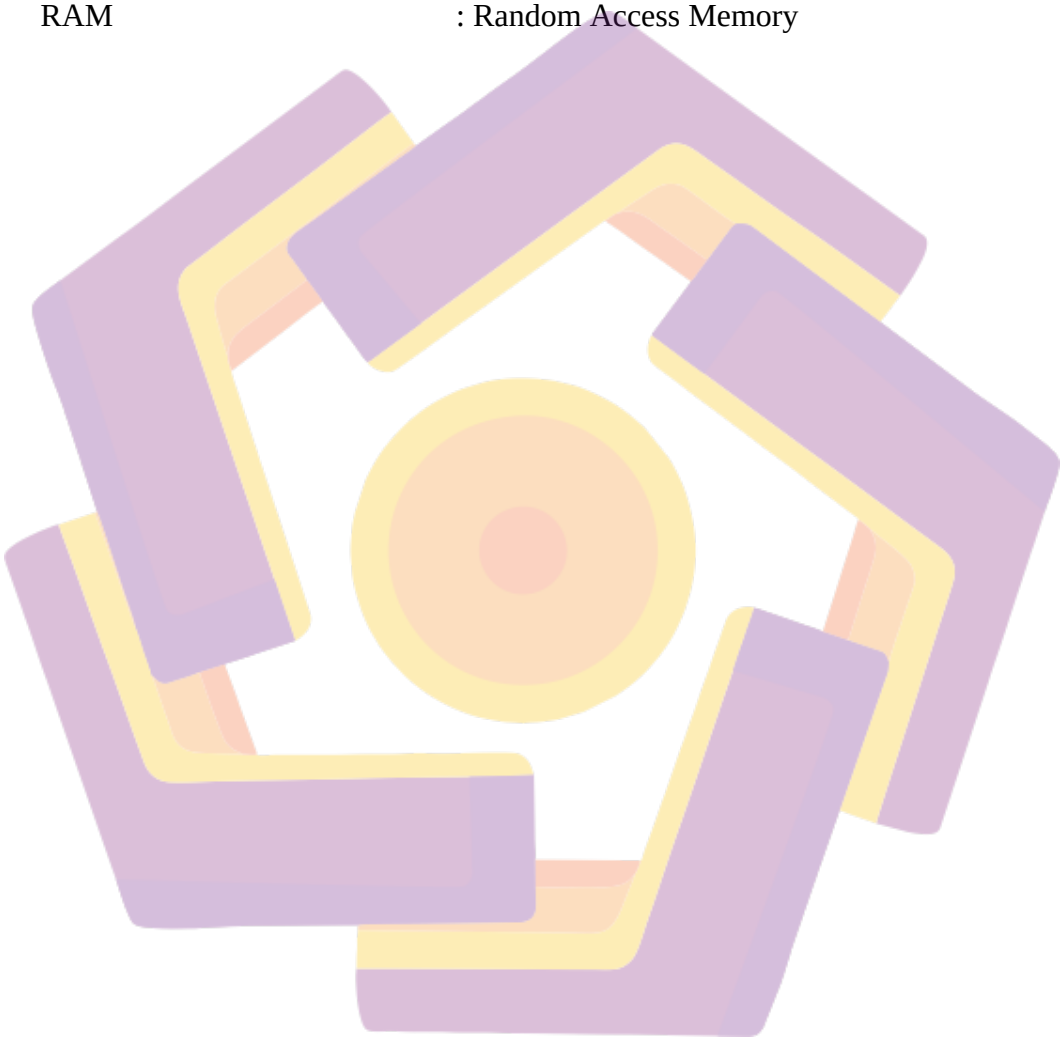


DAFTAR GAMBAR

DAFTAR LAMPIRAN

DAFTAR LAMBANG DAN SINGKATAN

API	: Application Programming Interface
AMQP	: Advanced Message Queuing Protocol
ORM	: Object Relational Mapper
CPU	: Central Processing Unit
RAM	: Random Access Memory



DAFTAR ISTILAH

Platform	: Kerangka atau lingkungan yang menyediakan alat dan layanan untuk menjalankan atau mengembangkan aplikasi.
Backend	: Bagian dari aplikasi yang berjalan di *server*, bertugas mengelola logika aplikasi, basis data, dan komunikasi dengan *frontend*.
Request	: Permintaan yang dikirim oleh klien ke *server* untuk memproses data atau mendapatkan informasi.
Race Condition	: Situasi di mana dua atau lebih *thread* mencoba mengakses atau memodifikasi data secara bersamaan, sehingga menyebabkan hasil yang tidak diharapkan.
Thread	: Unit terkecil dalam proses komputasi yang menjalankan instruksi secara independen.
Resource	: Sumber daya, seperti memori, waktu CPU, atau koneksi jaringan, yang digunakan oleh aplikasi untuk menjalankan tugasnya.
Overselling	: Menjual lebih banyak kapasitas sumber daya daripada yang sebenarnya tersedia.
Concurrency	: Kemampuan untuk menjalankan beberapa tugas secara bersamaan dalam suatu sistem.
Goroutine	: Unit ringan seperti *thread* pada bahasa pemrograman Go yang memungkinkan menjalankan fungsi secara bersamaan.
Channel	: Mekanisme komunikasi antar-*goroutine* untuk mengirim atau menerima data secara aman.
Mutex	: Mekanisme pengunci untuk mencegah beberapa *thread* mengakses sumber daya yang sama secara bersamaan.
Local Memory	: Memori yang hanya dapat diakses oleh satu proses atau *thread* tertentu.
Database	: Tempat penyimpanan data terstruktur yang dapat diakses dan diolah oleh aplikasi.
Distributed Database	: Basis data yang tersebar di beberapa lokasi fisik atau *server*, tetapi terlihat seperti satu sistem terpadu.

Message Broker	: Sistem perantara yang mengelola pengiriman pesan antar aplikasi.
Message Queue	: Antrian yang digunakan untuk menyimpan dan mengelola pesan sebelum diteruskan ke penerima.
Database Transactions	: Sekumpulan operasi pada database yang dieksekusi sebagai satu unit logis untuk menjaga konsistensi data.
Multi-Core	: Prosesor dengan beberapa inti (*core*) yang memungkinkan menjalankan banyak tugas secara bersamaan.
Synchronization Primitives	: Mekanisme dasar seperti mutex atau semaphore yang digunakan untuk mengelola akses bersamaan ke sumber daya.
Database Management System	: Perangkat lunak yang digunakan untuk membuat, mengelola, dan memelihara basis data.
Database Engine	: Komponen inti dari DBMS yang menangani penyimpanan, pengambilan, dan manipulasi data.
Overhead	: Beban tambahan pada sistem, seperti waktu atau memori, yang diperlukan untuk menjalankan suatu proses.
Instance	: Salinan spesifik dari sebuah program atau layanan yang sedang berjalan.
Asynchronous	: Pemrosesan yang berjalan di latar belakang tanpa menunggu hasil langsung, memungkinkan tugas lain tetap berjalan.
Synchronous	: Pemrosesan yang berjalan secara berurutan, di mana setiap tugas menunggu tugas sebelumnya selesai.
Event-Driven App	: Aplikasi yang merespons dan memproses peristiwa, seperti input pengguna atau pesan sistem.
Data Races	: Situasi di mana beberapa *thread* mengakses data secara bersamaan tanpa sinkronisasi yang benar, menyebabkan inkonsistensi.
Variable	: Entitas dalam program yang menyimpan nilai sementara untuk digunakan selama eksekusi.
File	: Berkas atau dokumen yang menyimpan data dalam bentuk tertentu di sistem penyimpanan.

Microservices	: Pendekatan arsitektur di mana aplikasi dipecah menjadi layanan-layanan kecil yang dapat dikelola secara independen.
Server	: Komputer atau sistem yang menyediakan layanan kepada klien melalui jaringan.
Processor	: Unit pemroses dalam komputer yang menjalankan instruksi dari program.
Storage	: Media atau tempat untuk menyimpan data.
Connection	: Hubungan atau saluran komunikasi antara klien dan *server*.
Local Connection	: Koneksi antara aplikasi atau layanan yang berjalan di mesin yang sama.
Programming Language	: Bahasa yang digunakan untuk menulis program atau aplikasi.
Framework	: Kerangka kerja yang menyediakan struktur dasar dan alat bantu untuk pengembangan aplikasi.
Load Tester	: Alat yang digunakan untuk menguji kinerja aplikasi di bawah beban tertentu.
Custom	: Dibuat atau disesuaikan sesuai dengan kebutuhan khusus.
Version Control	: Sistem untuk mengelola perubahan kode atau dokumen dalam pengembangan perangkat lunak.
Operating System	: Perangkat lunak yang mengelola perangkat keras dan menyediakan layanan dasar untuk program lain.
Text Editor	: Perangkat lunak yang digunakan untuk menulis atau mengedit kode atau teks.
Response Time	: Waktu yang dibutuhkan oleh sistem untuk merespons permintaan dari klien.
Throughput	: Jumlah tugas atau permintaan yang dapat diproses oleh sistem dalam waktu tertentu.
Bottleneck	: Bagian dari sistem yang membatasi kinerja keseluruhan karena keterbatasannya.
Latency	: Waktu tunda antara permintaan dan respons dalam sistem.

Baseline	: Titik referensi awal untuk membandingkan hasil pengujian.
Broker-Based	: Sistem yang menggunakan perantara (broker) untuk mengatur komunikasi antara komponen.
Client	: Aplikasi atau perangkat yang meminta layanan dari *server*.
Endpoint	: Titik akhir dalam API yang menerima dan memproses permintaan dari klien.
Worker	: Proses atau *thread* yang menjalankan tugas tertentu di latar belakang.
Library	: Kumpulan fungsi atau kode yang dapat digunakan kembali dalam pengembangan aplikasi.

INTISARI

Sistem penjualan tiket biasa menerima pengguna yang sangat banyak yang bertujuan untuk membeli sebuah tiket pada aplikasi atau situs tersebut. Proses pembelian tersebut membuat *backend* menerima banyak *requests* secara bersamaan (*concurrency*) dan akan rentan mengalami *race condition*, di mana alokasi data tiket menjadi tidak konsisten. Hal ini dapat menyebabkan *database inconsistency* yang mengakibatkan hal seperti tiket terjual ganda dan akan menyebabkan rusaknya reputasi dan kerugian finansial bagi perusahaan. Penelitian ini bertujuan untuk memitigasi permasalahan *race condition* di *backend* sistem penjualan tiket.

Penelitian ini mengusulkan solusi berbasis *message queue* pada *message broker* RabbitMQ. RabbitMQ digunakan sebagai *middleware* untuk membuat *queue* agar *requests* dapat dieksekusi satu persatu secara *asynchronous*. RabbitMQ juga mampu memberikan konfirmasi jika suatu proses telah berhasil. Hal ini dapat memastikan penghapusan dari *queue* hanya terjadi pada *message* yang sudah diproses.

Solusi ini menawarkan beberapa keunggulan dibandingkan dengan *database transactions*, seperti skalabilitas yang lebih baik, kompleksitas kode yang lebih rendah, dan performa yang lebih tinggi. Kontribusi utama penelitian ini adalah implementasi solusi *message queue* yang efektif untuk memitigasi *race condition* dalam sistem penjualan tiket berskala besar.

Kata kunci: Penjualan tiket, *concurrency*, *race condition*, *message broker*, *message queue*.

ABSTRACT

Ticket sales systems typically handle a high volume of concurrent user requests, making them susceptible to race conditions. Race conditions can lead to database inconsistencies, resulting in issues as double ticket sales which will lead to reputational damage and financial losses for the company. This research aims to mitigate race condition issues in the backend of ticket sales systems.

The proposed solution utilizes a message queue approach based on the RabbitMQ message broker. RabbitMQ acts as a middleware to create message queues, enabling asynchronous execution of requests. RabbitMQ allows consumers to acknowledge messages after successful processing (ack), ensuring that only processed messages are removed from the queue.

This solution offers several advantages over database transactions, including improved scalability, reduced code complexity, and enhanced performance. The primary contribution of this research lies in the implementation of an effective message queue solution to mitigate race conditions in large-scale ticket sales systems.

Keyword**:** *Ticket sales, concurrency, race condition, message broker, message queue.*