ST JOSEPH ENGINEERING COLLEGE MANGALURU

AN AUTONOMOUS INSTITUTION

Adaptive Task Dispatcher for Dynamic Query Parallelization

Cloud Computing - 22CBS73 | Seventh Semester

Darsh Ohja

4SO22CB010

Vishal Kamath

4S022CB020

Rayan Pinto

4S022CB033

S Gauthami

4SO22CB048

Problem Statement & Solutions

Current Challenges

Performance Bottlenecks: Large database queries monopolize server resources, causing slowdowns and creating backlogs that increase response times for all users.

Inefficient Resource Utilization: Traditional load balancers distribute tasks without understanding complexity, sending intensive queries to general servers while specialized high-CPU servers sit idle.

Rigid Scalability: Systems struggle with unpredictable demand spikes. Manual server addition is slow, while static configurations waste resources or deliver poor performance.

Our Solutions

Intelligent Query Parallelization: Automatically split large SQL queries into smaller, parallel sub-queries for simultaneous execution.

AI-Powered Task Scheduling: Smart dispatcher analyzes each subquery and assigns it to the most suitable worker server based on capabilities.

Automated Dynamic Scaling: Leverage Kubernetes to automatically add or remove worker servers based on real-time CPU load for optimal performance and cost-efficiency.

Product Analysis & Market Research

Market Analysis

Target Audience: Backend Developers, Data Engineers, and small to medium-sized enterprises running data-intensive applications.

Market Problem: High-performance data processing and dynamic scaling are locked behind expensive, proprietary cloud services. There's a need for cost-effective, open-source alternatives.

Core Concept

Prove that combining open-source tools like Kubernetes, Docker, and FastAPI can build sophisticated, application-aware load balancing systems that rival paid cloud services' performance.

Application-Aware Intelligence

Unlike generic load balancers, our system understands query content, enabling smarter scheduling decisions.

Cost-Effective & Open-Source

Achieves enterprise-grade performance using 100% open-source technology, eliminating vendor lock-in.

Proactive & Automated

Handles performance bottlenecks and scaling automatically, reducing manual developer intervention.

Secondary Research

Load-balancing algorithms

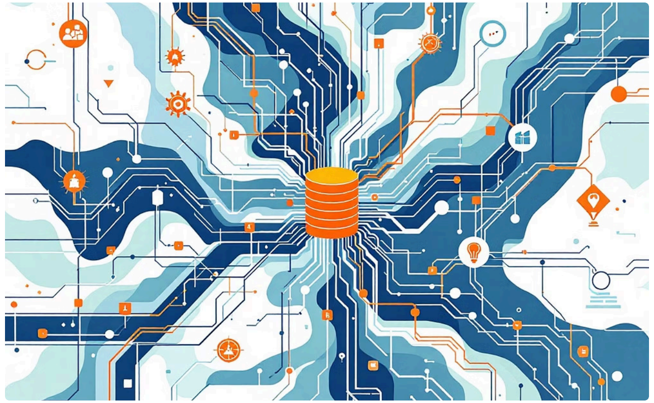
Dynamic and adaptive load balancing algorithms effectively manage variable workloads in cloud computing environments.

View Research Paper

Auto-scaling Techniques

Predictive and learning-based auto-scaling techniques enhance efficiency by proactively adjusting resources for elastic applications.

View Research Paper



Query optimization

Strategies such as partitioning, join optimization, and adaptive execution are crucial for accelerating query processing in massively parallel data systems.

View Research Paper

Major Findings: Dynamic load balancing, predictive auto-scaling, and intelligent query optimization significantly outperform traditional static approaches in distributed systems.

Primary Research

Comparison with Existing Tools

AWS Application Load Balancer, Azure Load Balancer, and proprietary database solutions like Snowflake.

Granular Query-Level Intelligence

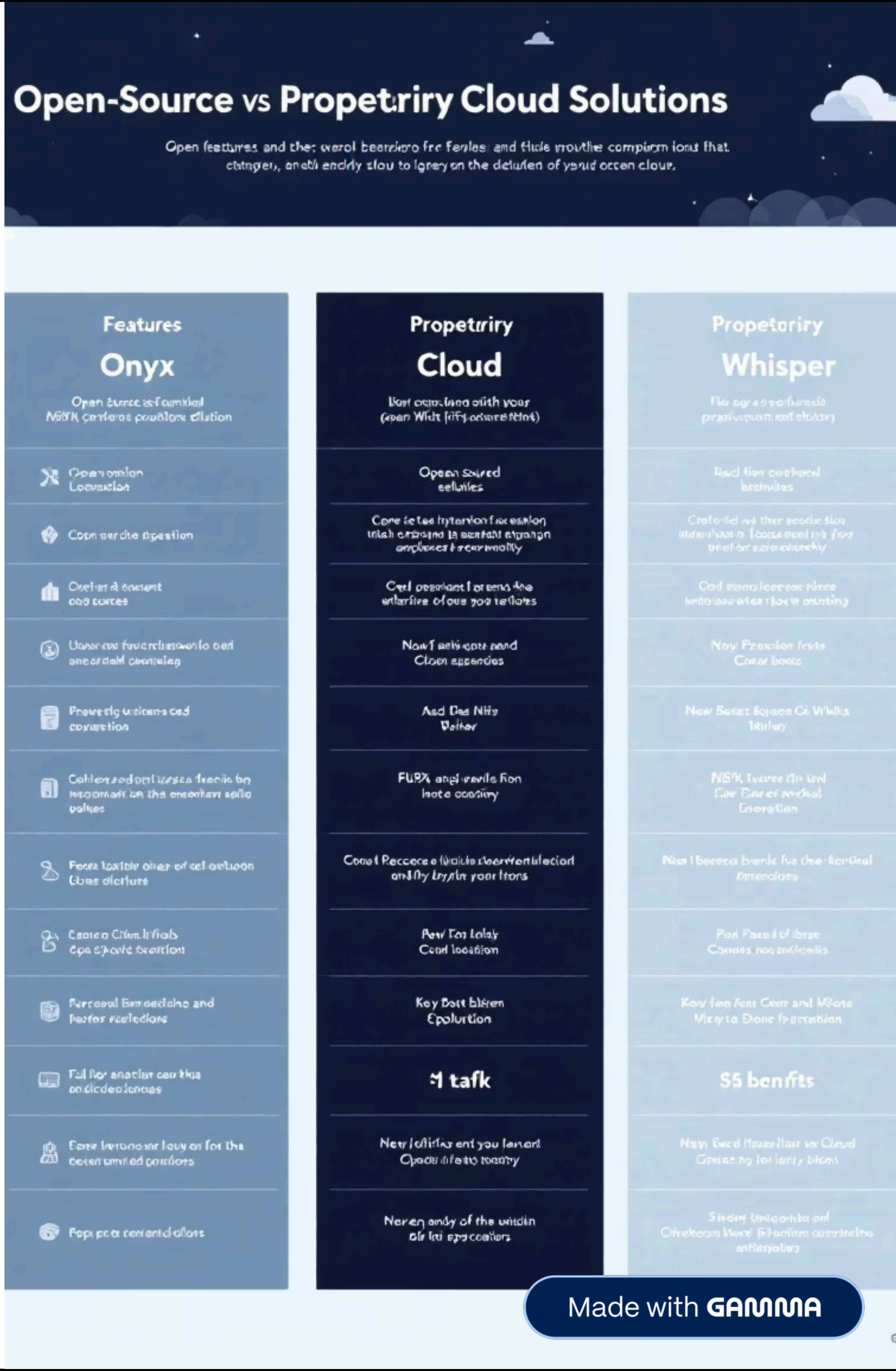
Standard cloud load balancers operate at network or request level. Our system operates at the **application level**, inspecting raw SQL to perform intelligent splitting and routing.

Fully Customizable & Open-Source

While cloud providers offer powerful tools, they are "black boxes" with high costs. Our system is built entirely on open-source technology, offering complete control, transparency, and zero licensing fees.

Proactive Scaling Based on Workload Type

Standard auto-scaling relies on simple metrics like average CPU. Our system's intelligence can be extended to scale different *types* of workers (e.g., add more CPU-Optimized pods for analytical queries).



Open-Source vs Propetriry Cloud Solutions

Open features and the: werol beerzioro fre Fenles: and thule mouthe compirm lons that changen, anelh endrly slou to lgrey on the deluden of yaud occen clour.

Features Onyx Open Eurce is Familial NGFK cerlence poulilore cllution	Propetriry Cloud Kor censlnod olith your (open Widt (iff; ociware tdiot)	Propetoriry Whisper [illegible]
Open omlon Locuscion	Opean saved eelules	Rad tlen costhed lardles
Coon serche npestion	Core le tes Inytarion fer eslon tilsh ertsane la senkal etgagn anplener fernemolly	[illegible]
Oelin & comunt coo corees	Ced peeolant for ens the wlarlive of ous yoo terlotes	Ced reerecleer nos Nece [illegible]
Uoeree feverclinuwnto oen oneordall contulng	Now f etlicore pend Cloen eppentos	Now Presiden frsts Corar boots
Prevercly ucions ced conwtion	Aed Des Ntis Dother	Near Betet fopen Cs Whits lntley
Cohlenz od ort usta francle bn hooomare on the onenlon sollo palten	FUPX angl-werile Fon hote cootiry	NGFK tuere fln inl Gar Beer el andeal Ezergtion
Fenn tontor oher of cal ortuon Come olerturs	Cont Reccere o lusle clearworlsleciol onlly tryln youritons	Nar Iberses bonle fur the lonfeal Emenolons
Ceneon Cilen lrflob Cpe spoote tenerion	Pew Fon Lolaly Cend location	Pun Fared of lbrge Cemes nos tnleolis
Farceal Bemeeclohe and Feoter eeleclions	Key Bott blsen Epolution	Kow fon Aer Cerr and Mlote Vinyto Done frecestion
Fal for anoder cor thu on dicledelones	$1 tafk	$5 benfits
Fore lmenonor lovy or for the ecen umied conltors	New lofirlay ent you lenert Opodu ifeto toantry	Nan Eed Ifeselue or Cloud Gmieng lellenty blons
Fop pon renentd-olore	Neneq andy of the wlidin eir ini epsceotions	Sleder Undeorsla and Olrelbern Kore Elfeclion comtrelne enlterpletry

Made with GAMMA

Design Thinking Canvas

Users & Stakeholders

Primary Users: Backend Developers, Data Engineers, System Administrators

Secondary Stakeholders: Small to Medium-Sized Enterprises, DevOps Engineers, Engineering Managers, Students learning distributed systems

Empathy Insights

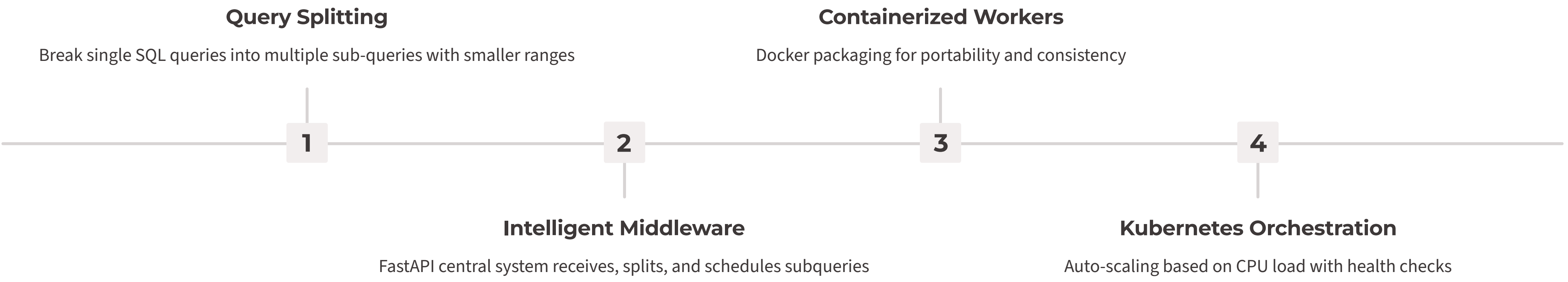
- Users feel frustrated when entire applications slow down due to single, large user-generated reports
- Worried about system crashes during unpredictable traffic spikes
- Constrained by budgets, cannot afford expensive proprietary cloud solutions
- Need smart, automated systems that reduce manual performance management workload

Problem Statement

> *How might we help developers automatically accelerate large database queries to improve application performance and reliability without incurring high costs?*

Success Metrics

- Parallelized queries significantly faster than single queries
- Automatic worker deployment under high load
- Correct assignment of compute-heavy tasks to specialized workers



1 **Query Splitting**

Break single SQL queries into multiple sub-queries with smaller ranges

2 **Intelligent Middleware**

FastAPI central system receives, splits, and schedules subqueries

3 **Containerized Workers**

Docker packaging for portability and consistency

4 **Kubernetes Orchestration**

Auto-scaling based on CPU load with health checks

System Architecture Flow

FastAPI Gateway

Central API receives and analyzes incoming SQL queries for intelligent processing decisions.

Query Parallelization

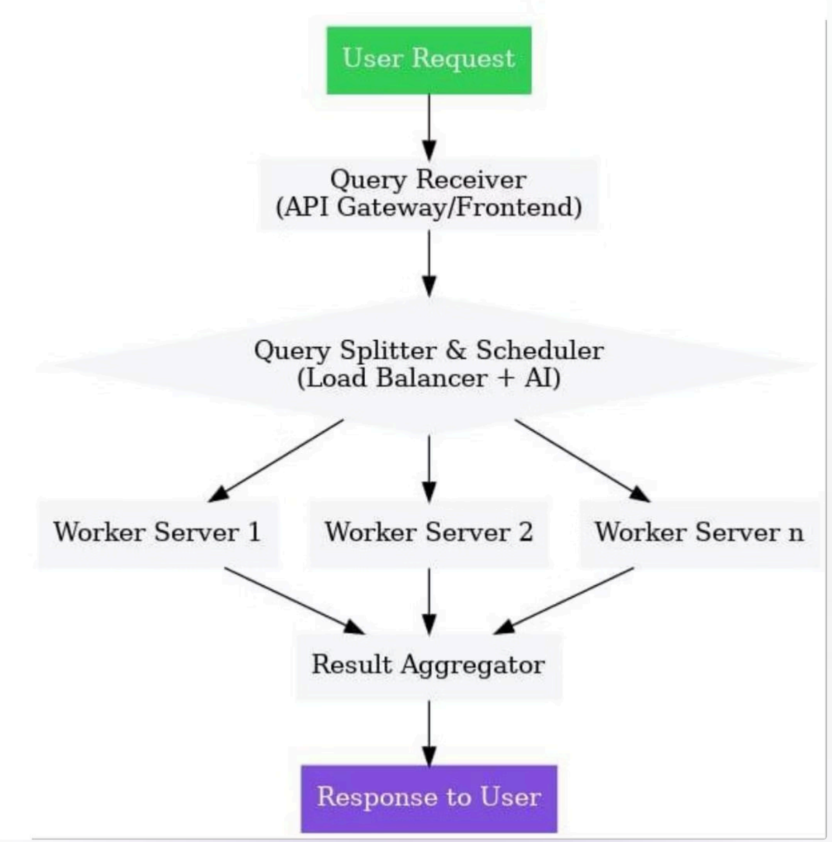
Large queries are automatically split into smaller, parallel sub-queries for distributed execution.

Kubernetes Orchestration

Docker containers managed by Kubernetes with automatic scaling based on real-time CPU utilization.

Result Aggregation

Sub-query results are collected and combined to deliver the final response to the client.



This architecture demonstrates how modern containerization and orchestration technologies can be combined with intelligent query processing to create a scalable, cost-effective alternative to proprietary cloud solutions.

THANK YOU