TUHH

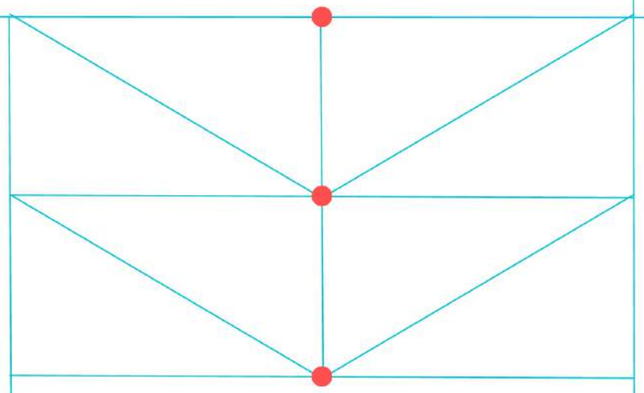
BD25_Project_F4_B

Real-Time Traffic Monitoring

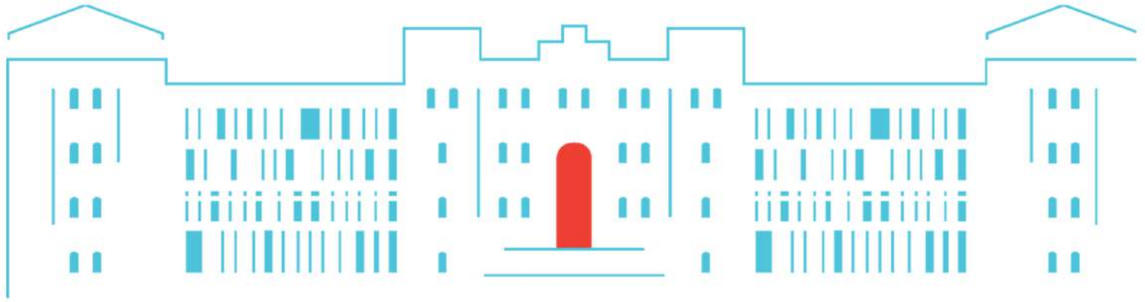
Final Presentation

Big Data PBL (SoSe25)

TUHH
Technische
Universität
Hamburg



18.06.2025



Presented by:
Gayathri Shiburaj, Manali Suhas Bansode, Ravi Raj, Shinu Shaju

Meet the Team

TUHH

Gayathri
Shiburaj

Manali
Suhas Bansode

Ravi
Raj

Shinu
Shaju

Project Goal & Objectives

TUHH

- Build a real-time, scalable system to monitor 10,000+ taxis live in Beijing using taxi GPS trajectory data
- Process high-throughput, continuous data streams with low latency
- Compute taxi speed, average speed, distance run, and violation alerts in real-time
- Present actionable insights live on a responsive dashboard for large-scale taxi fleet monitoring
- Design for scalability, fault tolerance, and future extensibility

Dataset Overview

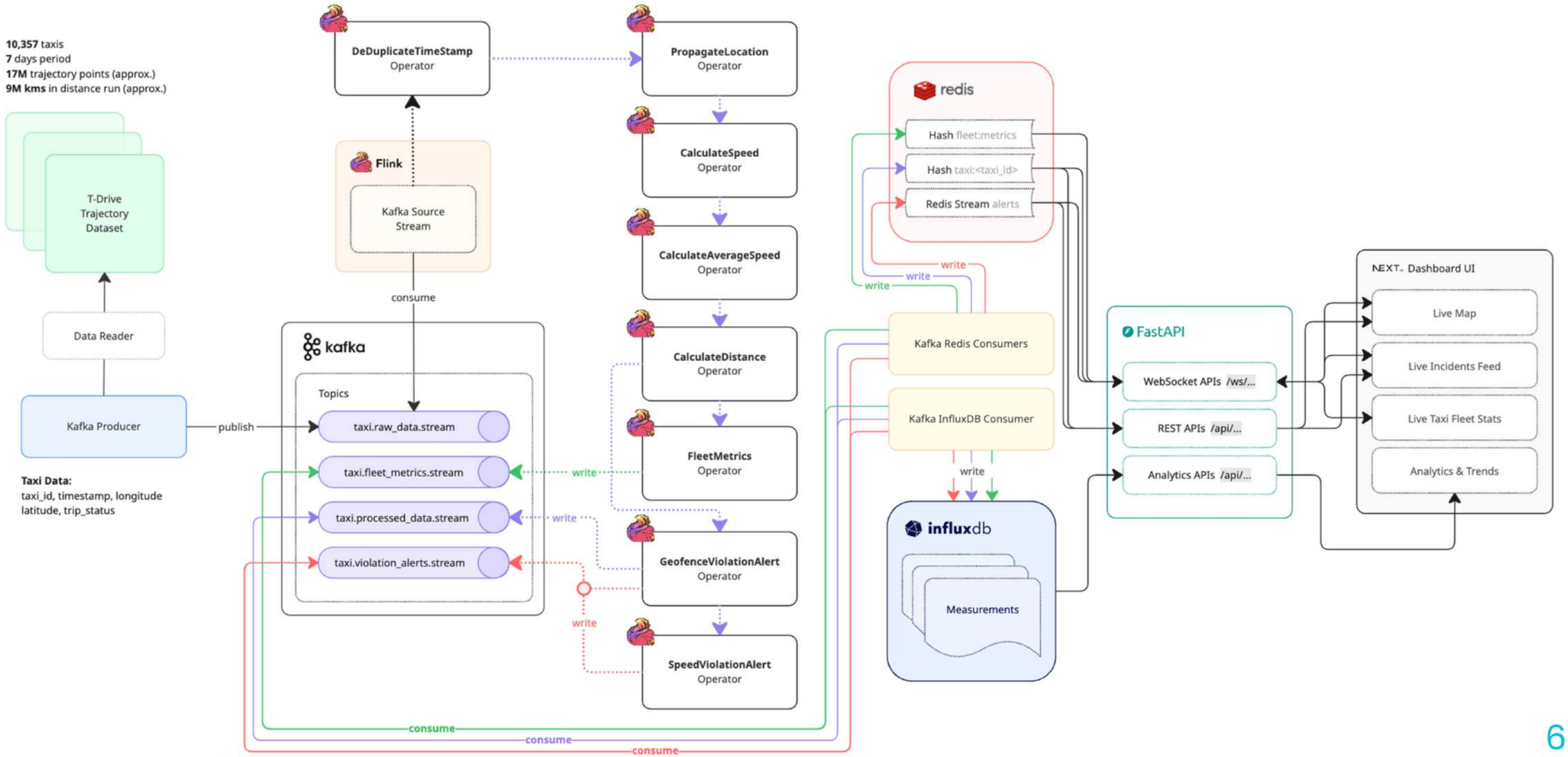
TUHH

- **T-Drive Trajectory Dataset** (Microsoft Research), Beijing, 2008
- **10,357** taxis, **~17M** taxi GPS data points over **7 days**, **~9M** total distance run
- Features: taxi_id, timestamp, latitude, longitude
- Provides high-frequency taxi updates to enable controlled real-time data ingestion
- Serves as a reliable ground truth for urban mobility systems under realistic conditions

Key Challenges

- **High Throughput:** Processing 17M+ GPS points from 10000+ taxis with a goal of lowest latency possible
- **Data Quality:** Handling invalid and null data gracefully without disrupting pipeline flow or compromising data quality
- **Real-Time Guarantees:** Deliver continuous, low-latency insights to end-users without delays
- **State Management:** Tracking each taxi's prior location data for computing speed, distance, and speed and gefence violations incidents
- **Scalability vs. Accuracy:** Balancing resources while maintaining stability and correctness under load
- **Fault Tolerance:** Recover reliably and quickly from failures without any data loss

System Architecture



System Architecture

TUHH

DataReader:

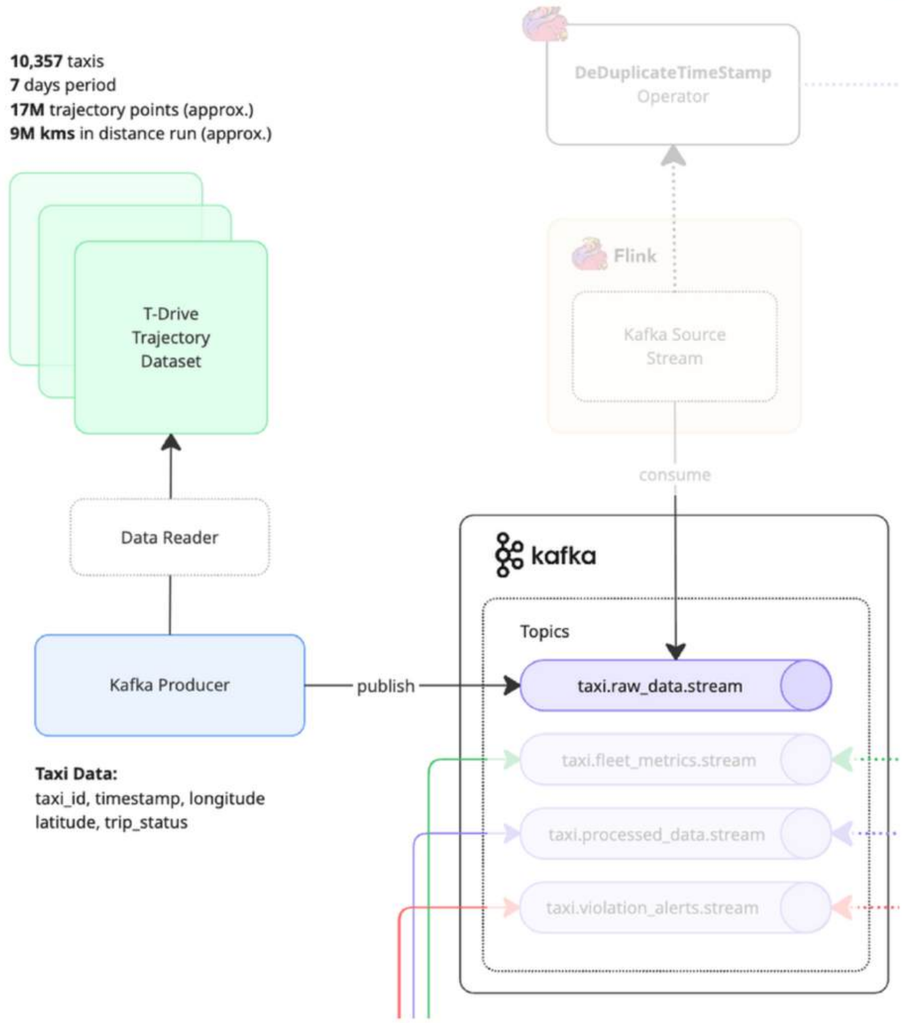
- Adds a trip_status field (active/end) to a taxi message
- Sorts records by timestamp for event-time correctness
- Serializes entire data as fast-loading pickle file
- Supports dataset size limit for controlled testing

Apache Kafka Producer:

- Supports high-throughput data streaming to match pipeline capacity
- Two data replay modes (actual event delays, simulated 1s)
- Data ingestion speed controls (1x–3x)
- Batches all records with same timestamp for efficiency

Kafka Broker:

- Deployed in KRaft mode (1 broker)
- Automated creation of Kafka topics with docker init container
- Web UI for monitoring

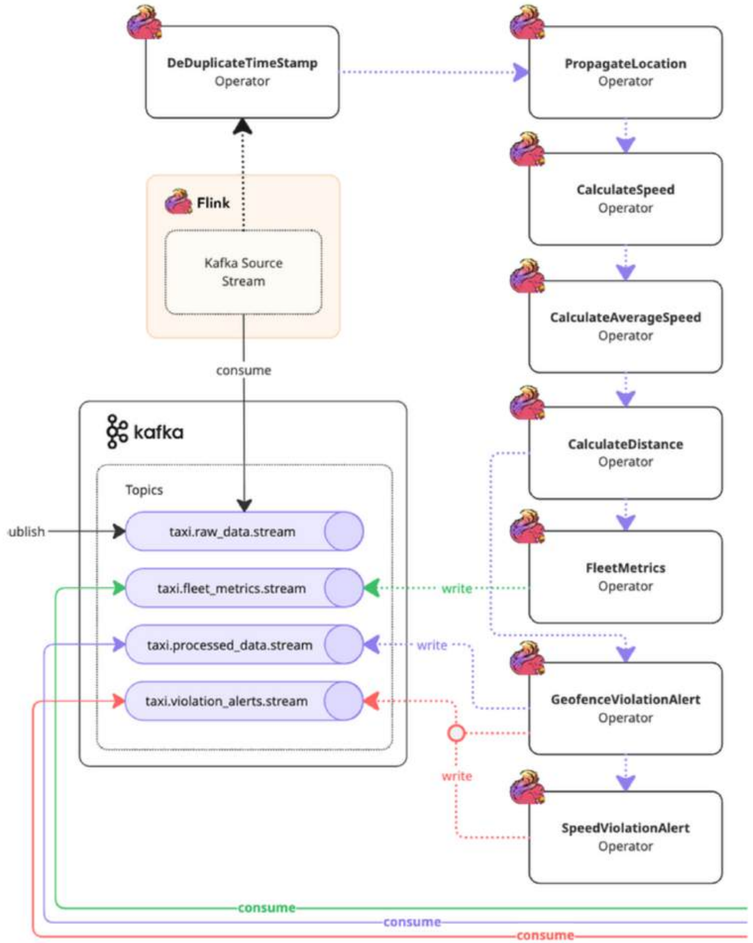


System Architecture

TUHH

Apache Flink DataStream Pipeline & Operators:

- Built and deployed a custom Docker image to support PyFlink DataStream APIs for efficient pipeline execution
- 1 x JobManager, 1 x TaskManager (parallelism = 1, task slots = 1)
- Docker init container to automate flink job creation & execution
- Kafka source connector creates datastream from source topic
- Source stream undergoes transformations by sequential operators
- Kafka sink connector sinks datastreams to target topics
- Event-time processing with monotonic watermarks
- Embedded RocksDB for state backend
- Incremental checkpointing for fault tolerance
- Web UI for monitoring and debugging



System Architecture

TUHH

Kafka Consumers:

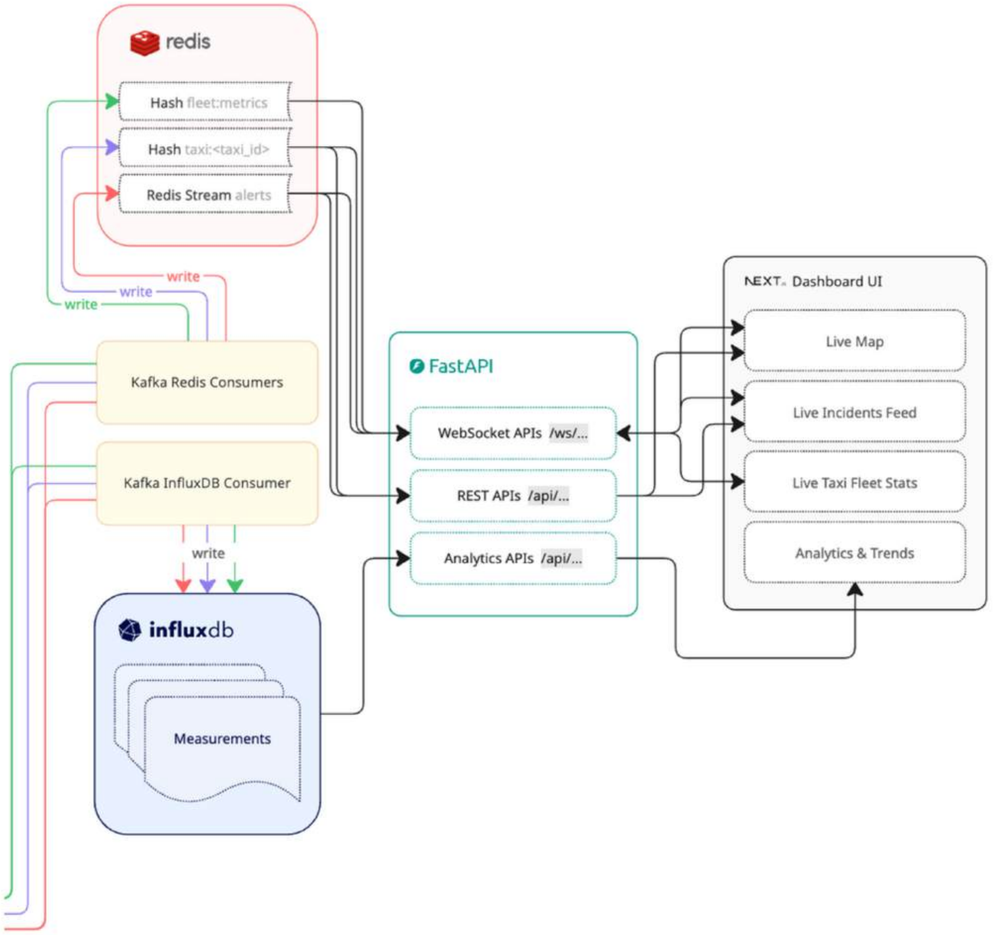
- Redis:
  - Redis Hash for caching taxi data and fleet metrics
  - Redis Streams for storing taxi violation alerts
- InfluxDB:
  - Uses InfluxDB Measurements for storing taxi data, fleet metrics and violation alerts for analytics

FastAPI Backend APIs:

- REST + WebSocket APIs for live & historical taxi data, violation alerts and analytics

NextJS Dashboard UI:

- Interactive map with Leaflet + clustering for scalable, responsive taxi tracking under load
- Live and historical incidents feed for speed and geofence violation alerts
- Taxi fleet-level and per-taxi metrics and analytics



Initial DataStream Pipeline Performance Challenges

TUHH

- High and increasing latency, hindering real-time responsiveness
- Persistent operator tasks backpressure and growing consumer lag under high load
- Unbounded state & timer growth in operators, increasing memory, resource usage and latency
- Performance degraded under load beyond 3,000 taxis due to throughput limits
- Unable to achieve low and stable latency while scaling to 9,000+ taxis

DataStream Pipeline Optimization Strategies

TUHH

- **PropagateLocation Logic:** Replaced timers with timestamp-based emission to prevent backpressure
- **Monotonic Watermarks:** Reduced latency by avoiding out-of-order and late arrival waits
- **DataStream Serialization & Deserialization:** Parsed Kafka JSON messages once at the Kafka source to a structured data, avoiding repeated serialization/deserialization in downstream operators
- **Fast Serialization:** Switched to orjson instead of standard json for faster, low-CPU serialization
- **EmbeddedRocksDB State Backend:** Enabled scalable, disk-backed state with incremental checkpoints
- **TaskManager Memory Tuning:** Increased managed memory, tuned network buffers for load
- **Frequent Checkpointing:** Incremental checkpoints (every 1s) for fast recovery with minimal overhead
- **State Cleanup:** Cleared trip ended taxis state in operators to prevent state bloat
- **Early Propagation:** Moved **PropagateLocation** operator to pipeline start to reduce downstream load

Performance Test: **Setup and Benchmarking**

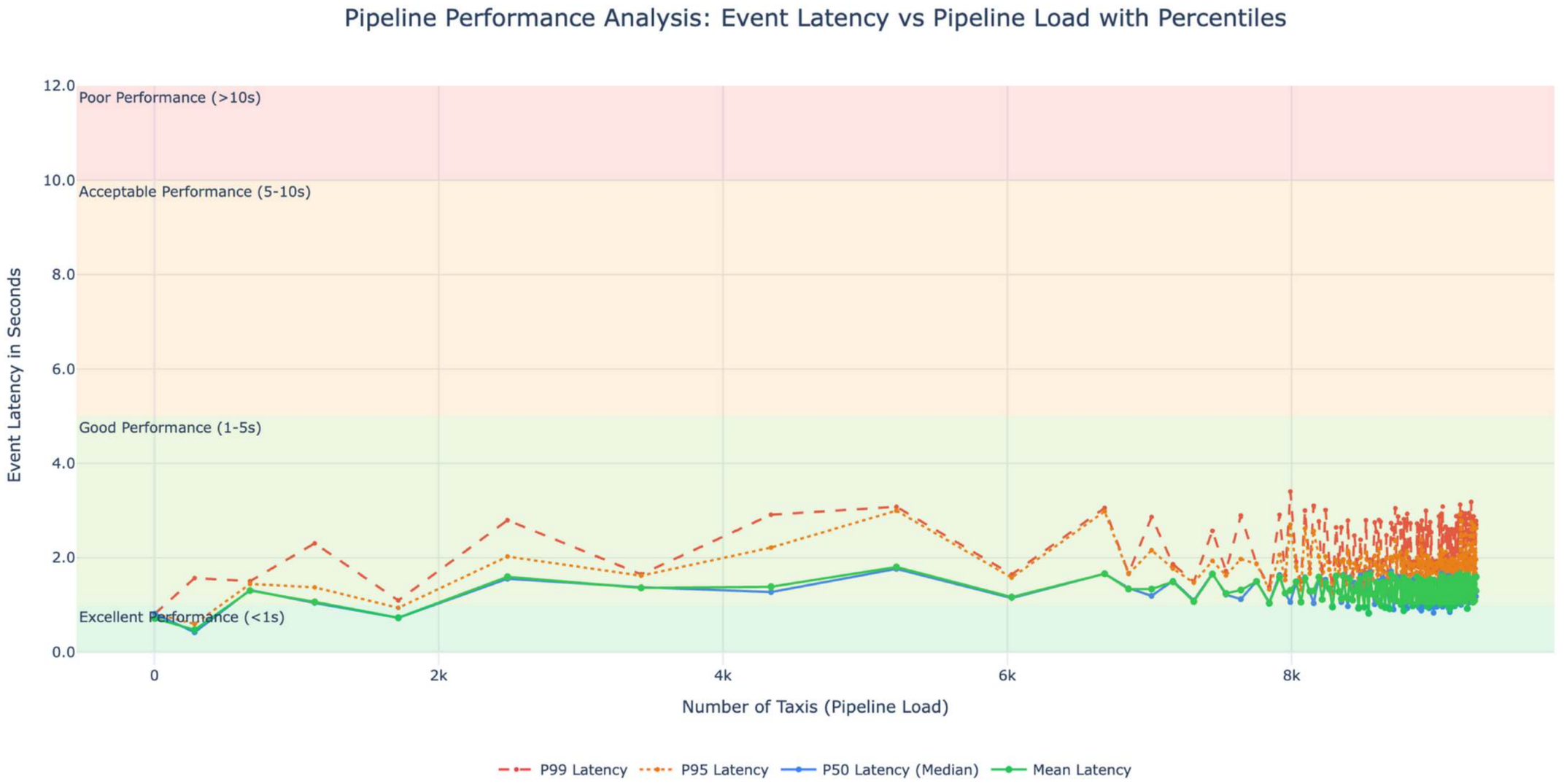
- **Benchmarking Measurements:**
  - **Event Latency:** Time from message ingestion to Kafka source topic (T1) to processing and sinking datastream to Kafka target topics (T2)
  - **Pipeline Throughput:** messages processed per second (40 msgs/s)
  - **Load simulation:** Variable taxi count over time (upto 10,000+ simultaneous taxis).

- **Performance Test Environment & Configurations:**
  - **Hardware Specifications:** 4 vCPU, 16GB RAM, 50GB SSD
  - **Kafka and Flink Configuration:** parallelism = 1, partitions = 1, task slots = 1
  - **Data Replay Configuration:** simulated mode at 1x speed, max 10,357 taxis
  - **Data Volume:** 400,000 messages processed over a 2-hour runtime

- **Custom Benchmarking Tool:** Developed a custom benchmarking tool to measure key metrics such as **event latency vs load**, **event latency per message**, and **throughput vs event latency** graphs

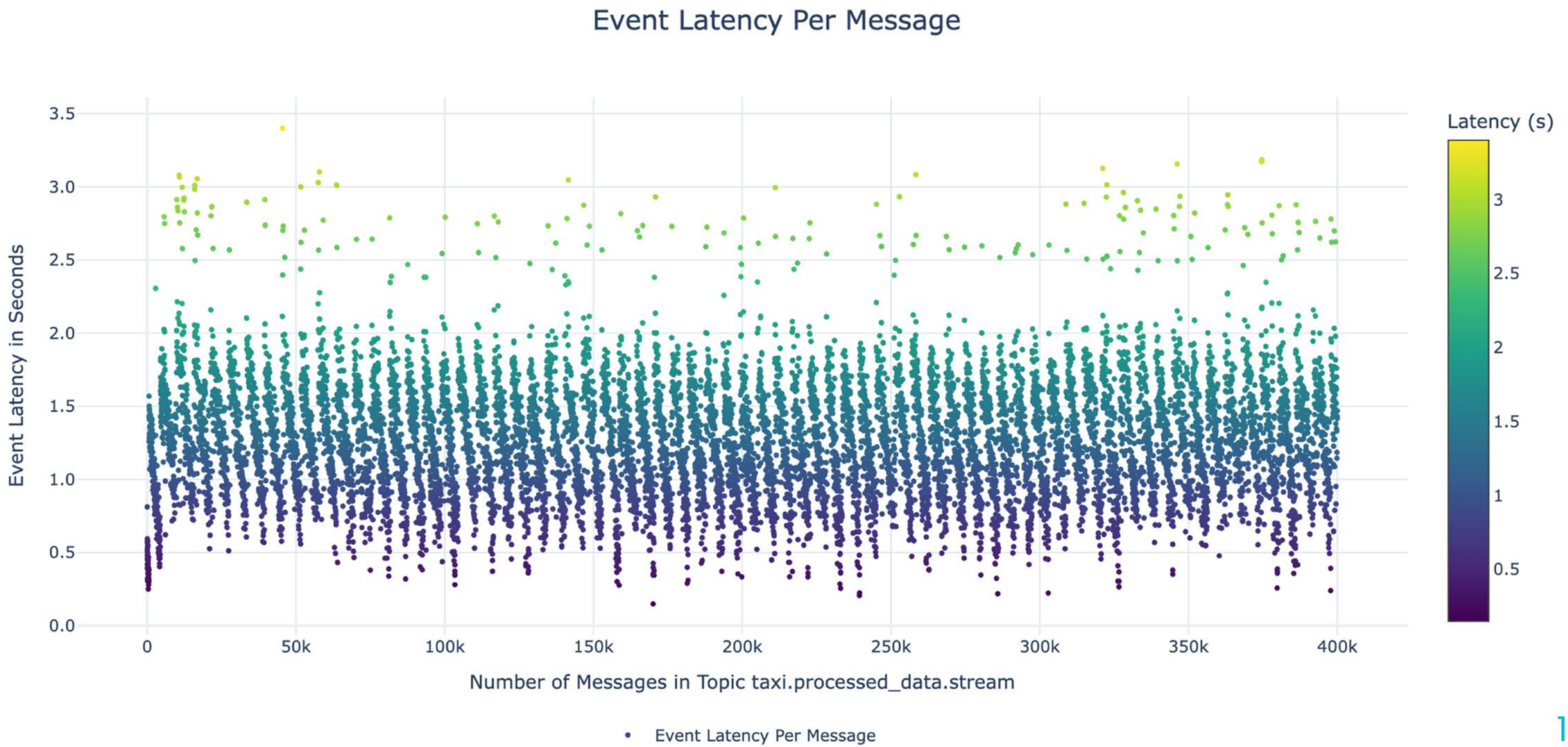
Performance Test: **Results and Analysis**

TUHH



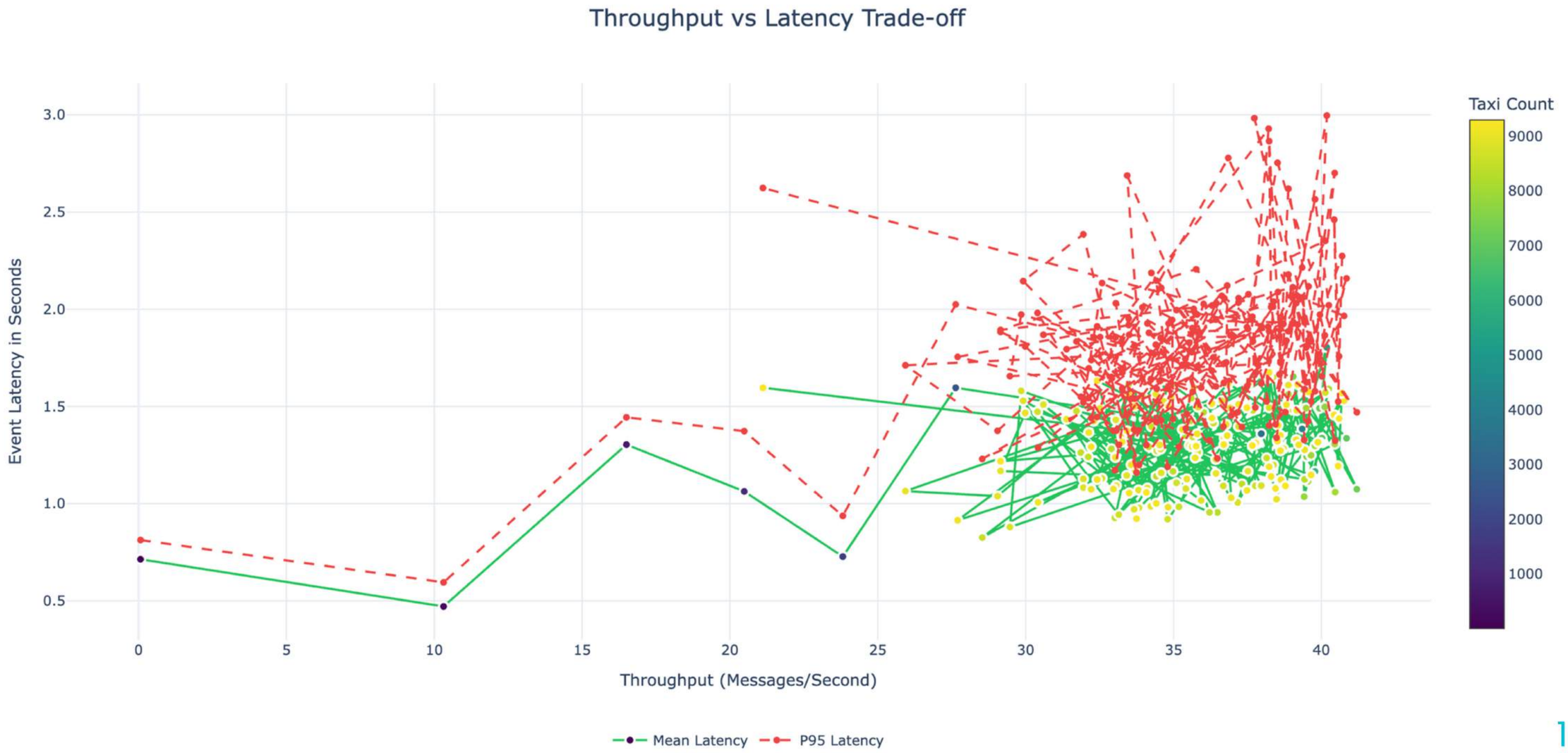
Performance Test: **Results and Analysis**

TUHH



Performance Test: **Results and Analysis**

TUHH



Performance Test: **Key Insights**

TUHH

- **Consistent Low Latency:** Processes 95% of messages within 2 seconds across all loads
- **Predictable Latency Growth:** Stable latency increase from low to higher test loads
- **Resilience:** Minor latency spikes at peak load, but quickly recover
- **Throughput-Latency Balance:** Efficiently handles medium to high workloads
- **No Backpressure:** Operators run with low utilization and smooth data flow
- **Fast Checkpointing:** Incremental RocksDB checkpoints complete in <1s
- **Stable Performance:** Achieves stable performance with simple pipeline setup
- **Scalable Real-Time Monitoring:** Supports reliable taxi fleet monitoring at scale

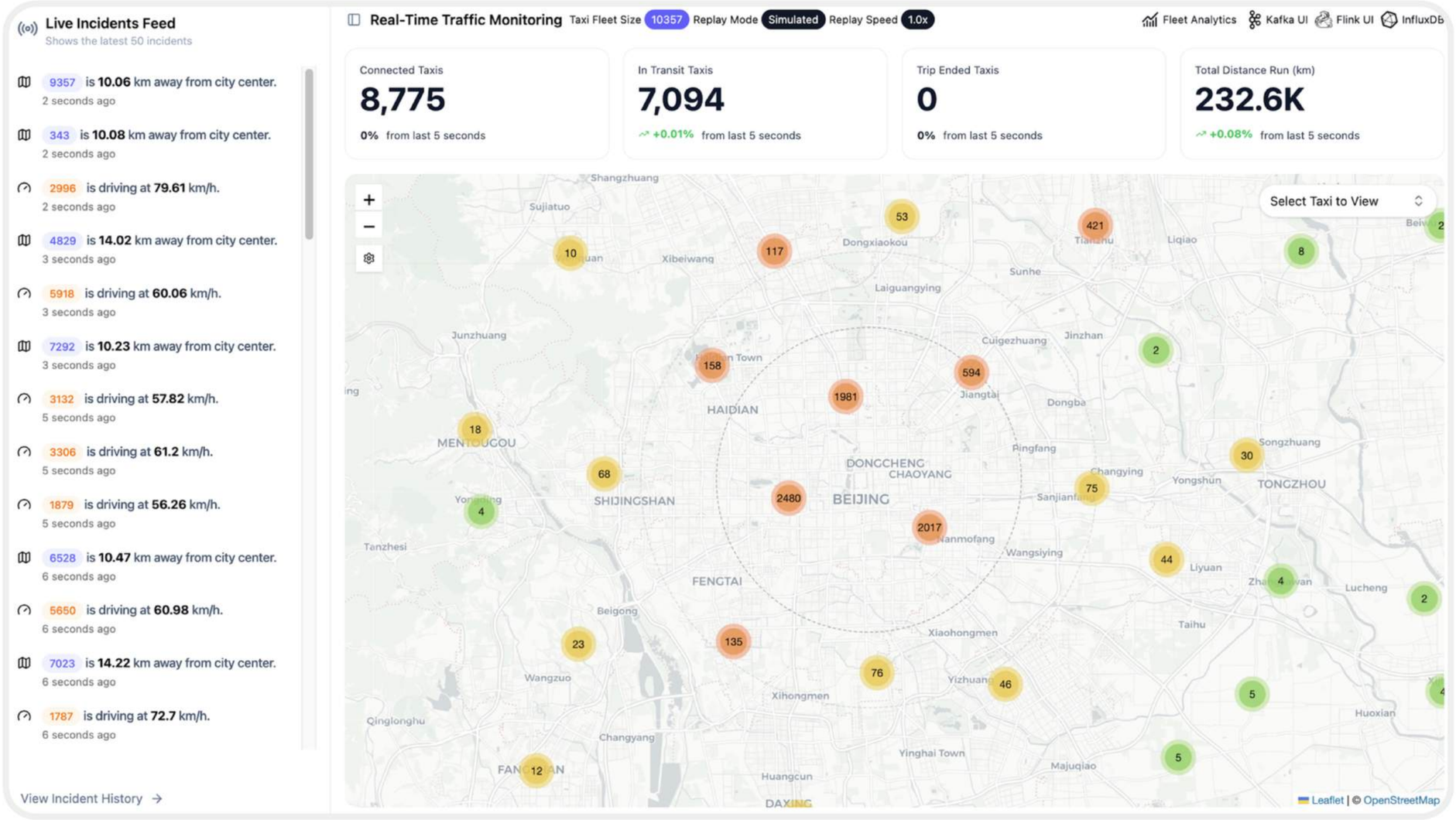
Extended Load Testing & Future Scaling

TUHH

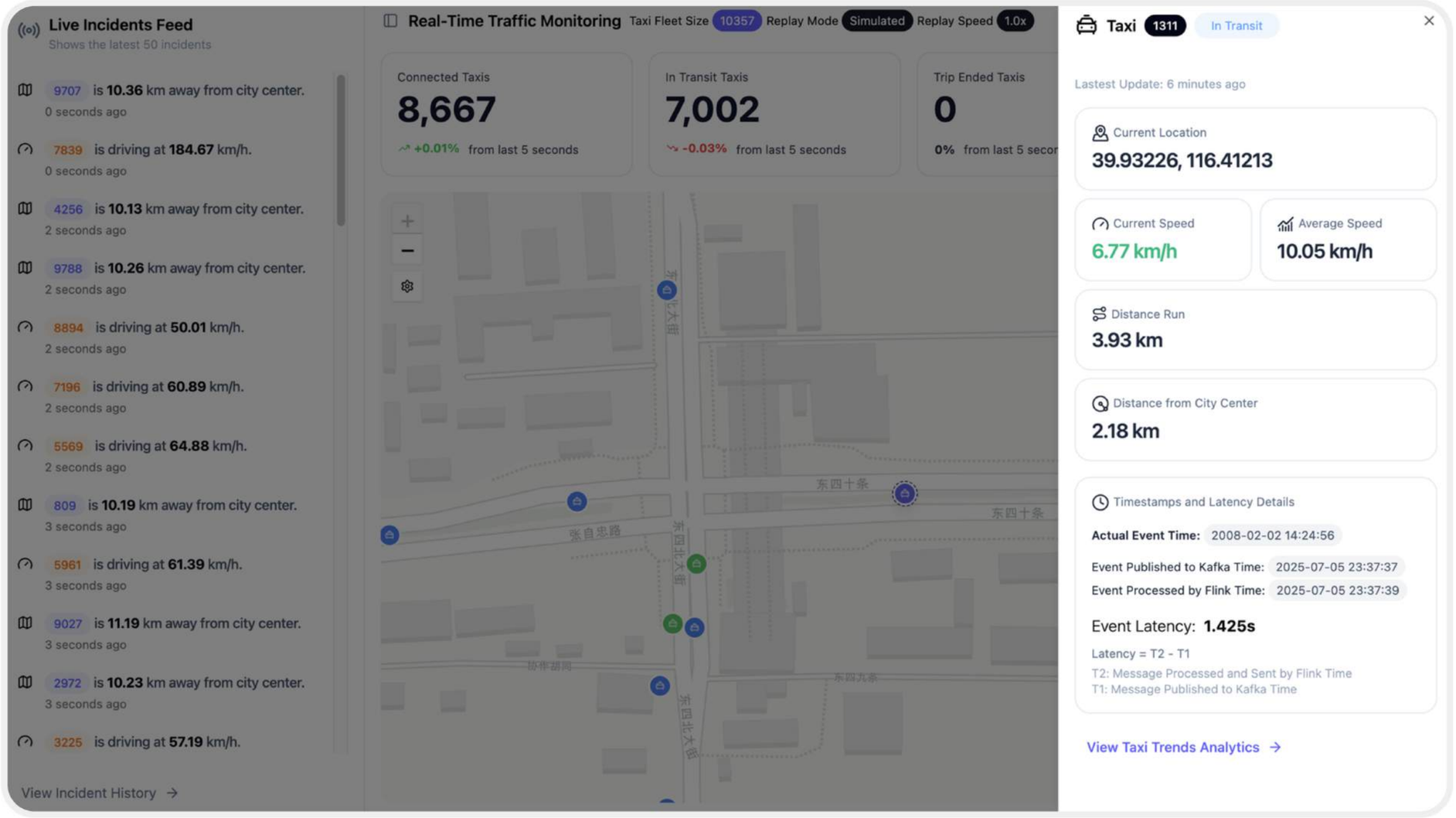
- **Extended Load Test:** At 3X replay speed (~120 msgs/s, 10,357 taxis), latency spiked sharply (60–250s) before stabilizing, indicating natural saturation near system capacity
- **Resilience Under High Load:** Longer busy periods and occasional backpressure observed for operator tasks, but the pipeline recovered quickly and remained stable
- Scaling checklist for higher throughput with low latency:
  - **TaskManager Scaling:** Increase TaskManager count and memory
  - **Increase Kafka Partitioning:** Use multi-partition topics for better load distribution
  - **Increase Parallelism & Task Slots:** Adjust operator parallelism and task slot allocation
  - **State Backend Optimization:** Tune RocksDB and checkpoint settings for high throughput
  - **Auto-Scaling:** Enable Flink dynamic scaling based on ingestion rates
  - **Dedicated VM Deployment:** Run high-load services on dedicated VMs for isolation and efficiency

Dashboard Key Features: **Live Fleet Monitoring**

TUHH

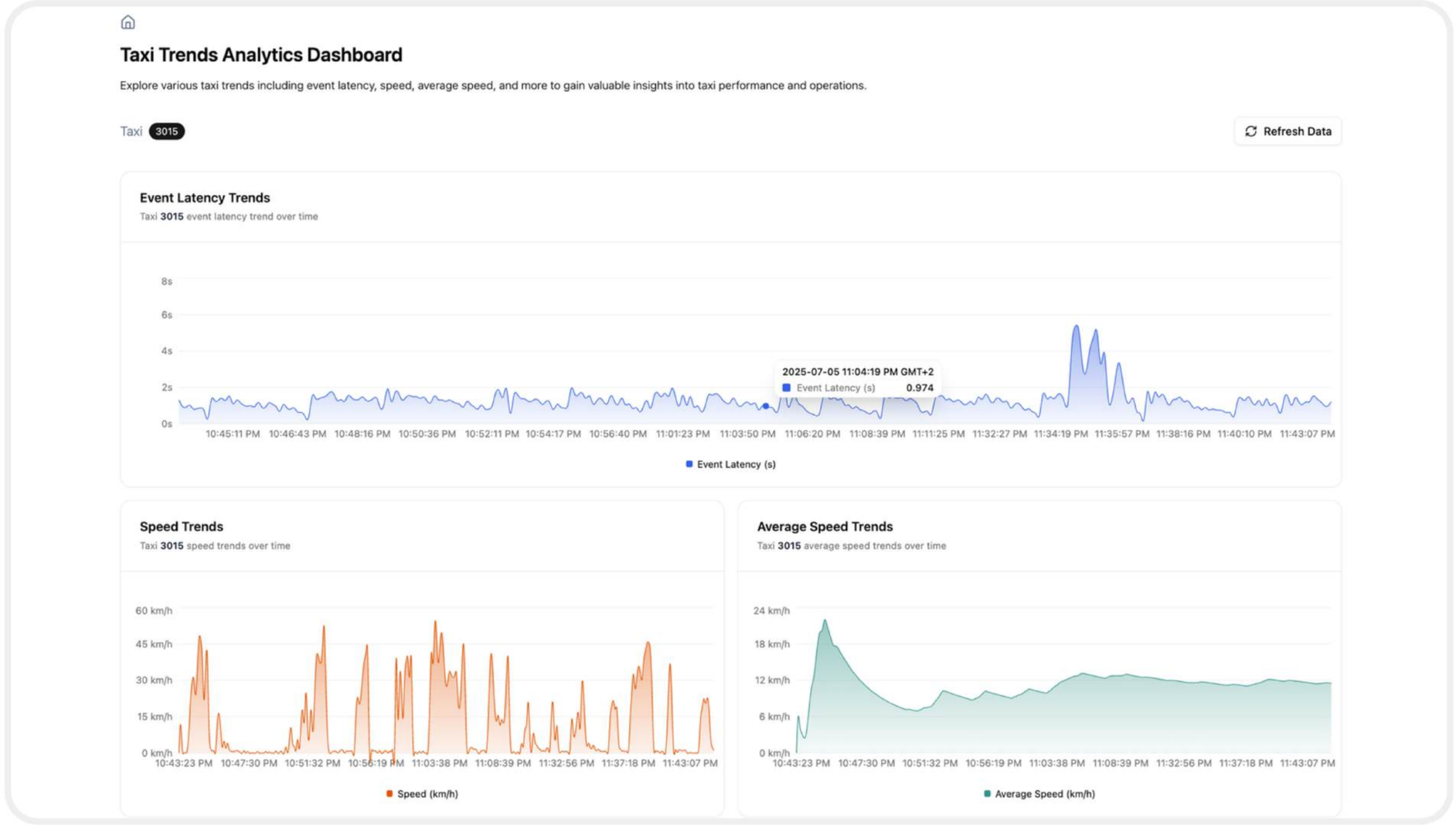


Dashboard Key Features: **Per-Taxi Monitoring**



Dashboard Key Features: **Per-Taxi Analytics**

TUHH



Dashboard Key Features: **Taxi Fleet Analytics**

TUHH

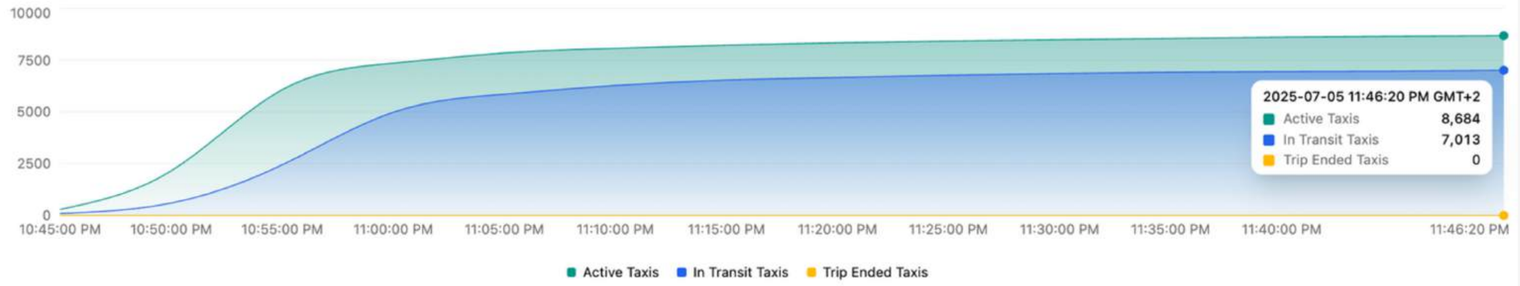
Taxi Fleet Trends Analytics Dashboard

Explore various taxi fleet trends to gain valuable insights into taxi fleet performance and operations.

Refresh Data

Taxi Fleet Metrics Trends

Taxi fleet metrics trends over time, including active taxis, in transit taxis, and trip ended taxis.



Taxi Fleet Event Latency Trends

Taxi fleet event processing latency trends over time

Max Latency	Min Latency	Average Latency
7.18s	**75ms**	**1.20s**
Highest Event Processing Latency in Seconds	Lowest Event Processing Latency in Seconds	Average Event Processing Latency in Seconds
+610.5% from last 5s	-89.6% from last 5s	+39.3% from last 5s

Dashboard Key Features: **Incidents History**

TUHH

Incidents

Total Incidents: **4426** Geofence Violations: **2513** Speed Violations: **1913**

Refresh Incidents

Taxi ID	Event	Timestamp	Actual Event Time	Current Coordinates [Lat, Lng]	Taxi Status
7740	speed_violation Taxi is driving at **51.86** km/h.	2025-07-01 11:21:47	2008-02-02 14:09:12	116.50778 40.01	active
7470	geofence_violation Taxi is **10.8** km away from city center.	2025-07-01 11:21:47	2008-02-02 14:09:14	116.28232 39.95712	active
10335	geofence_violation Taxi is **11.69** km away from city center.	2025-07-01 11:21:47	2008-02-02 14:09:13	116.45051 40.01315	active
7740	geofence_violation Taxi is **14.05** km away from city center.	2025-07-01 11:21:47	2008-02-02 14:09:12	116.50778 40.01	active
9375	speed_violation Taxi is driving at **59.18** km/h.	2025-07-01 11:21:44	2008-02-02 14:09:10	116.36329 39.96627	active
8767	geofence_violation Taxi is **11.17** km away from city center.	2025-07-01 11:21:44	2008-02-02 14:09:11	116.39371 39.81585	active
3316	geofence_violation Taxi is **10.72** km away from city center.	2025-07-01 11:21:44	2008-02-02 14:09:09	116.47845 39.9899	active
8554	speed_violation Taxi is driving at **119.29** km/h.	2025-07-01 11:21:41	2008-02-02 14:09:08	116.36673 39.8985	active
1277	speed_violation Taxi is driving at **53.69** km/h.	2025-07-01 11:21:41	2008-02-02 14:09:08	116.42946 39.90442	active
8549	speed_violation Taxi is driving at **50.01** km/h.	2025-07-01 11:21:41	2008-02-02 14:09:08	116.30526 39.94069	active
366	speed_violation Taxi is driving at **54.45** km/h.	2025-07-01 11:21:41	2008-02-02 14:09:08	116.46856 39.83864	active
4539	geofence_violation Taxi is **10.39** km away from city center.	2025-07-01 11:21:41	2008-02-02 14:09:07	116.46119 39.99579	active

Live Demo

TUHH

Let’s see it live… 🚀

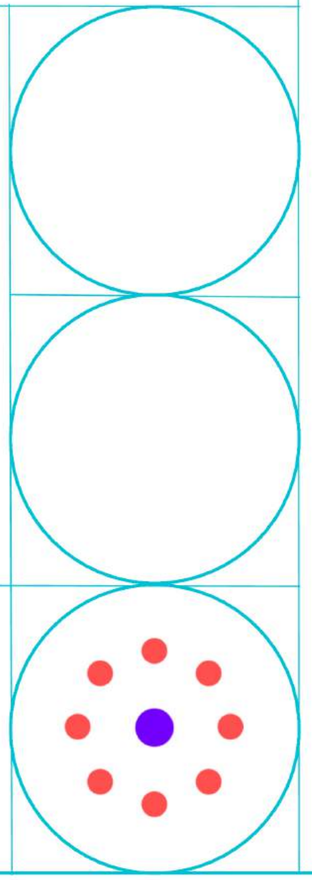
Conclusion & Key Achievements

TUHH

- Transformed slow system into a fast, scalable real-time taxi fleet monitoring platform
- Reduced average event processing time from ~35s or higher to <2s per message
- Supports 9,000+ taxis simultaneously without performance degradation
- 95% of messages processed within 2s across all load levels
- Smooth pipeline operation with no bottlenecks under heavy load
- Fast RocksDB checkpointing (<1s) ensures quick failure recovery
- Improvements are driven by optimized state management, memory tuning, and operator design
- System is reliable and ready for real-time taxi fleet traffic monitoring systems

Questions?

Thank You!



tuhh.de

TUHH
Technische
Universität
Hamburg