YugabyteDB Performance Testing Strategy (From Application Layer)

# Overview

This document outlines the approach to test the performance of YugabyteDB in our UAT environment from the application developer's perspective. The aim is to simulate realistic database usage patterns through our API application and evaluate YugabyteDB’s behavior under different workloads without directly accessing or tuning DB internals.

# Objectives

- Measure YugabyteDB query latency as observed from the API layer.  
- Evaluate database performance during CRUD operations under realistic load.  
- Identify query bottlenecks or slow responses from the app layer.  
- Validate how YugabyteDB behaves across regions (consistency, replication lag) during normal and peak usage.  
- Provide data points to collaborate with the DB team for deeper investigation.

# Key Metrics to Track

- Query response time: Time taken by YugabyteDB to return data  
- API-level DB latency: Time taken between query initiation and response  
- Throughput: Total DB ops per second observed via API  
- Error rate: Timeout, 5xx, or retry errors at DB level  
- Regional impact: Time differences in responses from different regions  
- Slow queries: Captured from Django debug tools

# Tools Required

- Load generation: Locust (Python-based) or k6  
- Django query analysis: Django Silk, Debug Toolbar  
- Performance logging: Custom logging middleware (timing DB calls)  
- Visualization: Grafana (if already integrated)

# Setup Instructions

1. Identify Target Workloads  
 - Read-heavy operations  
 - Write-heavy operations  
 - Mixed workflows  
  
2. Instrument Django Application for DB Timing  
 - Use Django Silk or custom middleware to capture per-view query performance  
  
3. Design Load Scenarios using Locust  
 - Simulate traffic through APIs connected to YugabyteDB  
  
4. Capture and Analyze Metrics  
 - Django: Avg/P95 query times  
 - YugabyteDB: From app layer logs  
 - API: DB call latency  
 - Load tool: RPS, failure rates  
  
5. Document Findings in a structured format

# Limitations

- Tests reflect application-side perception of DB performance.  
- Replication delays or internal DB issues are inferred.  
- In-depth diagnostics are out of scope for the dev team.

# Success Criteria

- Avg Query Latency < 250ms  
- P95 DB Time < 500ms  
- Error Rate < 1%  
- No data mismatch  
- API SLA met under load

# References

- https://docs.locust.io  
- https://github.com/jazzband/django-silk  
- Internal Grafana Dashboard  
- UAT Cluster Documentation