QBurst | A High AI-Q™ Company

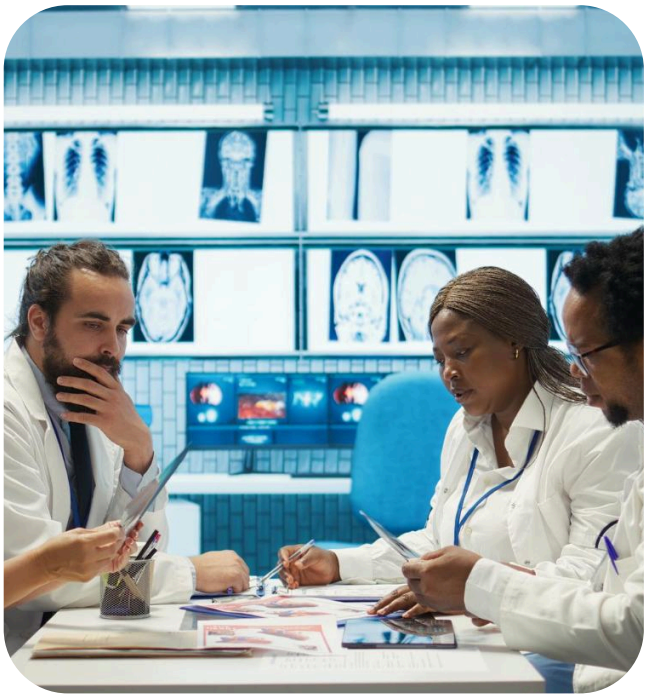
High-Availability Caching with Redis Enterprise for Mission-Critical Healthcare

Transforming healthcare application performance by migrating from legacy Memcached to a persistent, scalable Redis Enterprise cluster for real-time patient data.

Overview

We implemented Redis Enterprise as a centralized caching solution to provide high availability, persistence, and sub-millisecond response times for critical clinical applications.

- Achieved 99.99% uptime through automatic failover and clustering, ensuring uninterrupted healthcare operations even during node or data center failures.
- Reduced operational overhead by integrating enterprise-grade monitoring and automated sharding, freeing engineering teams to focus on clinical innovation.



Client Profile

The prominent US-based healthcare center serves thousands of patients daily through its multifaceted network of medical and research institutions. The center relies on high-speed digital infrastructure to deliver accurate, real-time data to healthcare professionals and researchers across the country.

Challenges: Volatility and Scalability Gaps

The reliance on open-source Memcached created significant risks for the center's mission-critical data layer:

- **Volatility and Data Loss:** As a purely in-memory solution, Memcached lost all cached data during server restarts, forcing heavy reloads on primary databases.
- **Limited Data Complexity:** Support was restricted to simple key-value pairs, failing to accommodate advanced clinical data types and complex search requirements.

- **Manual Scaling Hurdles:** Growing workloads required complex, manual intervention to scale clusters, leading to operational inefficiencies.
- **Absence of Failover:** Lack of built-in clustering meant a single node failure could cause significant service disruptions and data inconsistencies.

Solution: Enterprise-Grade Caching and Persistence

We transitioned the center to Redis Enterprise on Red Hat Linux, providing a robust, persistent caching platform. Our team migrated the existing Python applications to the redis-py client and re-architected simple key-value lookups into advanced Redis data structures.

The solution introduced a multi-tier resilience strategy.

- **Hybrid Storage & Persistence:** Implemented AOF (Append-Only File) and RDB snapshotting to safeguard cache data, with the flexibility to use SSD/NVMe-based Flash Shards for cost-effective scaling of large datasets.
- **Active-Active Replication:** Utilized conflict-free replicated data types (CRDTs) to maintain globally distributed caches with eventual consistency across regions.
- **Advanced Module Integration:** Leveraged RedisJSON and RediSearch to implement complex features directly within the cache, simplifying the overall application architecture.
- **Hardened Security:** Enforced enterprise-grade protection through TLS encryption and Role-Based Access Control (RBAC) to meet strict healthcare compliance standards.

Technical Highlights

- **Transparent Sharding:** Automated data distribution across multiple shards to improve performance under heavy clinical workloads.
- **Horizontal Scalability:** Seamless clustering allows the caching layer to grow in lockstep with the healthcare center's expanding digital services.

- **Unified Observability:** Integrated a web-based management console and CLI with real-time alerting for CPU, memory, and storage thresholds.
- **Automated Failover:** Zero-intervention failover mechanisms to ensure 24/7 availability for administrative and patient-facing portals.

Impact

- **Sub-Millisecond Latency:** Application response times dropped significantly, enabling faster transactions and a smoother experience for healthcare professionals.
- **99.99% Availability:** Automatic failover and active-active replication ensured high business continuity, even during infrastructure failures.
- **Operational Efficiency:** Automated monitoring and cluster management reduced manual maintenance effort by providing a single pane of glass for the entire caching layer.
- **Compliance Ready:** Built-in security features and TLS encryption ensured that sensitive patient data remained protected, aligning with US healthcare regulations.
- **Developer Productivity:** By utilizing Redis modules like RedisTimeSeries, developers could implement complex tracking features without needing additional data stores.