Term Project Update 2: Distributed Vector Database

Project Information

- **Name**: Nihar Patel
- **Advanced Area**: Distributed databases for high-dimensional vector data (AI embeddings)

Current Implementation Status

I have successfully implemented a distributed vector database system with the following components:

1. **LSH-Based Sharding**: Implemented a Locality-Sensitive Hashing approach for distributing vectors across shards, ensuring similar vectors are likely to be placed on the same shard.

2. **Coordinator-Based Architecture**: Created a central coordinator that:
   - Distributes vectors to appropriate shards using LSH
   - Routes search queries to relevant shards
   - Merges results from multiple shards
   - Handles fault tolerance with configurable replication
3. **Shard Nodes**: Implemented shard nodes that:
   - Store vector data using FAISS indices
   - Perform efficient similarity search
   - Provide performance statistics
4. **API Layer**: Built REST APIs for both coordinator and shard nodes to enable:
   - Vector insertion
   - Similarity search
   - System statistics retrieval
5. **Docker Integration**: Created Docker configuration for containerized deployment with:
   - Coordinator service
   - Multiple shard services
   - Volume mapping for persistent storage
6. **Performance Analysis**: Implemented comprehensive performance analysis tools:
   - Latency vs. k (number of results) benchmarking
   - Vector distribution analysis across shards
   - Dimension reduction from 2048D to 512D using PCA

7. **Visualization**: Added visualization capabilities for:
    - Shard distribution
    - Latency metrics
    - Search results

Learning Goals Progress

1. **Distributed Indexing**: Successfully implemented LSH-based sharding for distributing vectors across nodes while preserving similarity relationships.
2. **Query Optimization**: Implemented query routing to minimize latency by selecting the most relevant shards for each query.
3. **CAP Theorem Trade-offs**: Configured system with adjustable consistency modes and replication factors to demonstrate CAP theorem principles.

Next Steps

1. **Optimization**: Fine-tune LSH parameters for better sharding distribution and query performance.
2. **Fault Tolerance**: Enhance replication strategy with automatic failover mechanisms.
3. **Benchmarking**: Conduct comprehensive benchmarks with larger datasets to evaluate scalability.
4. **Documentation**: Create detailed documentation on system architecture and performance characteristics.

Feedback Request

I apologize for not being able to attend office hours due to my hectic course schedule with multiple assignments and research work. I would appreciate your feedback on my current implementation, particularly regarding:

1. The effectiveness of LSH for vector sharding
2. The coordinator-based architecture approach
3. Any potential optimizations for distributed query execution

I'm committed to maintaining active communication via email throughout the development process and will share updates as I make further progress.