Term: Fall 2024 **Subject:** Computer Science & Engineering (CSE) **Number:** 512

Course Title: Distributed **D**atabase **S**ystems (CSE 512)

GROUP PROJECT GUIDELINES

Project Overview:

In this semester-long group project, students will work in teams of 3-4 to collaboratively design, implement, and evaluate a functional prototype or simulation of a distributed database system or one of its key components. The project will require students to apply various concepts and techniques learned throughout the course, with each phase focusing on a different aspect of distributed databases.

Project Objectives:

- Apply theoretical knowledge of distributed database systems to a practical project.
- Gain hands-on experience with designing and implementing distributed systems.
- Learn to collaborate effectively in a team, sharing responsibilities and knowledge.
- Produce a final report and pre-recorded video presentation demonstrating your project's key features, challenges, and performance.

Team Responsibility:

- Every team member should equally contribute to the overall project implementation and documentation.

Project Selection:

Each team will choose a project idea from the list provided or propose their own idea, subject to instructor approval. The chosen project should include the following components:

- **System Design**: Plan for data partitioning, replication, fault tolerance, and query processing.
- **Implementation**: A working prototype or simulation demonstrating core features.
- **Evaluation**: Metrics to evaluate performance, scalability, and fault tolerance.
- **Documentation**: Well-documented code, system architecture, and results

Project Topic/Application Pool (Not just limited to this)**:** The overall project idea covering the distributed database should be specific to an application area such as:

Traditional Systems	Modern Applications	Emerging Technologies
Library Management Inventory Management Metadata Management E-commerce Platforms Financial Systems Healthcare Information Systems Payroll Management Ticket Reservation Systems	Social Media Applications IoT Data Management Real-Time Analytics Donation Systems Insurance Management Systems Electric Bill Systems Restaurant Management Online Retail Applications Electronic Banking University Data Management	Blockchain and Distributed Ledgers Online Gaming Platforms Logistics and Supply Chain Management Autonomous Vehicle Networks Edge Computing for Smart Cities Telecommunications Infrastructure Media Streaming Services Collaborative Cloud-Based Applications

Project Ideas (Not just limited to this)**:** The project idea shall include concepts like data distribution, data modeling, fragmentation, replication strategies, query processing, consistency models, transaction management, fault tolerance, scalability, NoSQL database systems, etc.,

Distributed File Storage System: Build a basic distributed file storage system similar to HDFS (Hadoop Distributed File System) or Google File System (GFS).
Design a Distributed Key-Value Store: Develop a key-value store like Amazon DynamoDB or Apache Cassandra.
Distributed Query Processing and Optimization: Implement a distributed query processing engine that splits, distributes, and executes queries over multiple nodes.
Eventual Consistency in NoSQL Databases: Build a small-scale NoSQL system that demonstrates the trade-offs between consistency, availability, and partition tolerance.
Distributed Transaction Management: Implement a distributed transaction management system supporting ACID properties.
Fault-Tolerant Distributed Database: Build a fault-tolerant distributed database system where nodes can fail and recover without losing data or consistency.
Geo-Distributed Database: Develop a geo-distributed database that optimizes query processing across different geographic regions.
Google Spanner-like Distributed SQL Database: Recreate a simplified version of Google Spanner, focusing on global consistency and horizontal scaling.
MongoDB Sharding and Replication Simulator: Develop a project that simulates MongoDB's sharding and replication mechanisms.
Benchmarking Distributed Databases: Benchmark various distributed databases (e.g., Cassandra, MongoDB, CockroachDB) under different loads and workloads**.**

Project Milestones:

The following milestones must be completed and submitted by the respective dates. Teams should keep track of their progress and reach out to the instructor or TA if they encounter challenges.

Milestone 1: Project Proposal

- **Submission Date:** 10/13/2024
- **Description:** Submit your finalized project proposal, including the problem statement, project objectives, and initial design.

Milestone 2: System Design and Plan

- Submission Date: 11/03/2024
- Description: Submit the detailed design document, including architecture, partitioning scheme, replication model, fault-tolerance mechanism, and consistency model.

Milestone 3: Final Submission

- **Submission Date:** 12/02/2024
- **Description:** Submit the final project code, report, and presentation video.
- **Report Format:** (A template will be provided)
    - Introduction: Problem statement and objectives.
    - System Design: Detailed architecture and components.
    - Implementation: Key features, challenges, and technologies used.
    - Evaluation: Performance results, trade-offs, and conclusions.
    - Conclusion: Summary of project and future work.
    - Pre-recorded Video presentation
        - Create an engaging and informative video (~ 3 Minutes) that visually demonstrates the system's design, implementation, and functionality.
        - Narrate (voice-over) or add subtitles to explain the project's objectives, challenges, and solutions.
        - Include snapshots, diagrams, or animations to illustrate key concepts and results.
        - Emphasize the significance of the project in the context of the chosen topic.

    Submission Format:

    - Submit the ZIP file containing the project report, code/prototype sample, and video presentation on Canvas.

Tools and Technologies (Not just limited to this)**:**

You are free to use any programming language or framework, provided that the project's complexity is appropriate. Some suggested technologies:

- Languages: Java, Python, C++, Go
- Databases: MongoDB, Cassandra, CockroachDB, Spanner, etc.
- Frameworks: Apache Kafka, Hadoop, Spark
- Cloud Platforms: AWS, Google Cloud, Microsoft Azure

Data Usage:

- **Data Selection:** Each team is responsible for selecting appropriate data for their project that aligns with the chosen topic. The data should be relevant, representative, and sufficient to test the distributed database system effectively.
- **Data Sources:** Teams may use publicly available datasets, synthetic data generated for testing purposes, or data obtained from partner organizations, provided that proper permissions and ethical considerations are adhered to.
- **Testing Data**: Ensure to include a variety of data scenarios to comprehensively evaluate system performance, such as different load conditions, edge cases, and potential failure scenarios.

Evaluation Criteria:

Each project will be evaluated on the following criteria:

1. **Technical Complexity (30%)**
    - Design complexity (e.g., partitioning, replication, fault tolerance).
    - Implementation quality (e.g., scalability, performance).
    - Use of advanced concepts (e.g., consistency models, transaction processing).
2. **Functionality (50%)**
    - Working prototype/simulation.
    - Demonstration of key features as outlined in the project proposal.
3. **Teamwork and Contribution (5%)**
    - Clear distribution of tasks among team members.
    - Evidence of collaboration (teamwork will be peer-reviewed).

4. Project Report (10%)

- Clarity and depth of explanation.
- Detailed design and implementation documentation.
- Proper results (with snapshots) and analysis of results.

5. Presentation (5%)

- Clear communication of the project's objectives, design, and outcomes.
- Effective use of visuals and demonstrations.

----------XXXXXX----------