**Project Title:** - Distributed Resource Optimization System

DUE DATE: 12th of April 2024.

**Objective:**

Design and implement a distributed system that monitors and optimizes resource utilization across multiple nodes in a network. The system should dynamically allocate and deallocate resources based on demand, ensuring efficient utilization, and preventing resource bottlenecks.

**Tools:**

1. **Programming Language:**

* Choose a language suitable for distributed systems, such as Python, Java, or Go.

1. **Resource Monitoring:**

* Utilize Prometheus for monitoring resource usage across nodes. It provides metrics collection and monitoring capabilities.

1. **Communication Protocol:**

* Use gRPC for efficient communication between nodes. It supports bidirectional streaming, making it suitable for real-time updates on resource utilization.

1. **Dynamic Resource Allocation:**

* Implement a dynamic resource allocation algorithm that adjusts resource allocation based on real-time demand. Consider load forecasting and predictive analysis.

1. **Fault Tolerance:**

* Implement fault tolerance mechanisms to handle node failures and ensure continuous resource optimization. Use distributed databases like Apache Cassandra or etcd for storing critical state information.

1. **Security:**

* Implement secure communication using TLS for resource information exchange between nodes. Apply access controls to prevent unauthorized access to resource allocation decisions.

1. **User Interface:**

* Develop a user interface for administrators to visualize resource utilization, allocation decisions, and historical trends.

Timeline:

**Week 1: Project Planning and Design**

* Define project objectives and requirements.
* Choose the programming language and development tools.
* Design the overall architecture of DistResourceOptimizer.

**Week 1: Resource Monitoring and Communication**

* Implement resource monitoring using Prometheus.
* Set up gRPC for efficient communication between nodes.

**Week 1: Dynamic Resource Allocation**

* Develop and implement a dynamic resource allocation algorithm.
* Test the algorithm's effectiveness in adjusting resource allocation based on demand.

**Week 1: Fault Tolerance and Security**

* Implement fault tolerance mechanisms to handle node failures.
* Integrate Apache Cassandra or etcd for storing critical state information.
* Implement TLS for secure communication and access controls.

**Week 1: User Interface and Testing**

* Develop a user interface for administrators to visualize resource utilization and allocation decisions.
* Conduct extensive testing, including stress testing and simulations.

**Week 1: Documentation and Finalization**

* Create detailed documentation for setting up, configuring, and using DistResourceOptimizer.
* Finalize the project, addressing any remaining issues.
* Prepare for the project presentation.

**Week 1: Project Presentation and Evaluation**

* Each group presents their DistResourceOptimizer implementation.
* Evaluate projects based on criteria such as functionality, fault tolerance, security, and efficiency.
* Provide feedback and discuss lessons learned.