Practical No.	Module No.	Title of the **Experiments**	Type of Experiment		Topics to be highlighted	CO Map
			PBL	Newly Added		
		Design distributed system for a collaborative real-time multiplayer gaming platform.				
1	1	Identify the concept on which operating system distributed computing works			Distributed OS	CO1
2	2	Implement a Distributed application using socket. Application consists of a server which takes an integer value from the client, calculates factorial and returns the result to the Client program. Activity 02			Communication	CO2
3	2	Design a Distributed Application for remote computation			RPC/RMI	CO2
4	3	Simulate a distributed system with multiple nodes (processes or computers), each with its own local clock.	Thought provking	new	Clock Synchronization	C03
5	3	Using messages between nodes at random intervals, with timestamps assigned based on the node's local clock.			Multiple Nodes and Local Clocks	CO3
6	3	Simulate a distributed system where multiple processes request and hold shared resources			Deadlock Management	CO3
7	3	Apply concept of Mutual Exclusion algorithm for distributed system othim) activity 06			Mutual Exclusion	CO3
8	3	Compute concept for Token based Mutual Exclusion (Raymond Tree) Activity 07		New	Mutual Exclusion	CO3

9	4	Create multiple "server nodes" (processes or threads) to handle incoming tasks.	Yhought provking		Load management	CO4
10	4	Design a distributed application which consist of a server and client using threads. Activity 09			Multithreading	CO4
11	6	Understanding the mounting and unmounting process of files using NFS Activity 10			DFS	CO6
		Scenario: Imagine a distributed cloud computing platform that handles user requests for running computational tasks, such as data analysis, machine learning training, and video rendering. The system consists of multiple geographically distributed nodes with varying capacities and workloads. Users expect their tasks to be executed quickly and efficiently while the system must minimize costs, balance workloads, and ensure fairness across nodes. However, challenges arise when: 1. Some nodes become overloaded while others are underutilized. 2. High-priority tasks get delayed because resources are occupied by lower-priority tasks. 3. Task dependencies across nodes lead to inefficient execution and delays.	PBLE			

		Problem Statement: How can the distributed system implement an efficient scheduling algorithm that: 1. Balances workloads across nodes to optimize resource utilization. 2. Ensures fairness and prioritization for critical tasks. 3. Handles task dependencies and communication overhead effectively. 4. Adapts to dynamic changes in workload and resource availability				
		Scenario: A team building a distributed ticket booking system for a global event. The system is hosted across multiple servers in different regions to handle high user traffic. Each server maintains a replica of the ticket inventory to reduce latency and improve availability. However, since thousands of users are attempting to book tickets simultaneously, the following synchronization issues arise: 1. **Overbooking:** Multiple users are allocated the same ticket due to race conditions between servers. 2. **Inconsistent State:** Some servers show available	PBLE			

		tickets while others show sold-out, leading to user dissatisfaction. 3. **Delayed Updates:** Due to network delays, updates about ticket availability take too long to propagate across servers, creating confusion.				