NIRMA UNIVERSITY

Institute:	Institute of Technology, School of Technology
Name of Programme:	MTech CSE
Course Code:	6CS206CC25
Course Title:	Systems Design
Course Type:	Core
Year of Introduction:	2025-26

L	T	Practical Component				C
		LPW	PW	W	S	
2	0	4	-	-	-	4

Course Learning Outcomes (CLO):

At the end of the course, the students will be able to:

1. demonstrate proficiency in contemporary tools for communication and data storage (BL2)
2. apply suitable tools to implement, test, and deploy the applications (BL3)
3. analyse the system design with respect to reliability and efficiency (BL4)
4. design the high-level architecture of the system. (BL6)

Unit	Contents	Teaching Hours (Total 30)
Unit-I	**Introduction to Systems Design:** Systems Design Basics, Need for Systems Design, Monolithic and Microservice Approaches	02
Unit-II	**Network and Distributed Systems:** IP and Addressing, TCP/UDP, HTTP, Concurrency and Synchronization, Scheduling and Logging, Rate Limiter, Leader Election, Clustering, Availability, Scalability	05
Unit-III	**APIs and Asynchronous Communication:** REST API, GraphQL, gRPC, Websockets, Long-Polling, Server-Sent Events, API Gateway, Message Queues, Message Brokers, Publish and Subscribe, Distributed Queues	05
Unit-IV	**Databases and Storage:** SQL and NoSQL Databases, Indexing, Searching, Normalization, Replication and Sharding, Consistency models, Distributed Transactions, CDN, Blob Storage and S3	06
Unit-V	**Performance Aspects and Resiliency:** Load Balancing, Caching, Hashing, Service Discovery, Circuit Breakers, Disaster Recovery, Testing and Monitoring, Securing the System	05
Unit-VI	**Architecture, Infrastructure and Case Studies:** Patterns and Architectures for System Design, Virtual Machines, Serverless Computing, Containers and Orchestration Tools, **Case Studies:** e-Commerce Product Listing, API Rate Limiter, Web Crawler, Twitter Trends, Netflix, Uber.	07

Self-Study:

The self-study contents will be declared at the commencement of the semester. Around 10% of the questions will be asked from self-study content.

Suggested Readings/ References:

1. Martin Kleppmann, Designing Data-Intensive Applications, O'Reilly
2. Roberto Vitillo, Understanding Distributed Systems: What every developer should know about large distributed applications, Shroff Publishers
3. Mark Richards and Neal Ford, Fundamentals of Software Architecture: An Engineering Approach, O'Reilly
4. Dennis, Wixom, and Roth, System Analysis and Design, Wiley
5. Titus Winters, Tom Manshreck, and Hyrum Wright, Software Engineering at Google, O'Reilly
6. Alex Xu, System Design Interview – An Insider's Guide, Shroff Publishers.

Suggested List of Experiments:

Sr. No.	Name of Experiments/Exercises	Hours
1	Implement concurrent server application using socket programming and deploy it on containers	06
2	Design REST API based server with CRUD operations as well as applying GraphQL concepts and deploy it on local resources in form containerized services	08
3	Implement a producer-consumer problem using a message queue (RabbitMQ, Kafka) with Open-source tools as well using containers and cloud platform	08
4	Implement horizontal sharding for a basic database application and extend it with Docker Compose and containers	06
5	Set up master-slave replication between multiple databases spread across containers	04
6	Build a simple LRU (Least Recently Used) cache server and containerize it	06
7	Implement a simple round-robin load balancer demonstrating how traffic is distributed across multiple servers hosted as containers with Docker Swarm	06
8	Implement JWT (JSON Web Tokens) based authentication for a simple web application and deploy this service as a container	04
9	Implement an API rate limiter using token bucket or leaky bucket algorithms and extend it with an API gateway component on cloud platform	08
10	Build a fault-tolerant system by introducing retries and circuit breakers in an API based application and extend the deployment on orchestrated containers.	04