



National University of Engineering (UNI)
School of Computer Science
Syllabus 2027-I

1. COURSE

CS3P2. Cloud Computing (Mandatory)

2. GENERAL INFORMATION

2.1 Course	: CS3P2. Cloud Computing
2.2 Semester	: 10 th Semester.
2.3 Credits	: 3
2.4 Hours	: 1 HT; 4 HP;
2.5 Duration of the period	: 16 weeks
2.6 Type of course	: Mandatory
2.7 Learning modality	: Face to face
2.8 Prerequisites	: CS370. Big Data. (9 th Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Modern computing relies heavily on distributed architectures and cloud-native platforms. This course provides an in-depth exploration of cloud computing services, distributed database concepts, and web platforms. Students will learn to design scalable architectures, understand the implications of distributed data consistency, and evaluate the privacy and security challenges inherent in cloud-based environments.

5. GOALS

- Understand the architecture and deployment models of cloud computing.
- Design distributed database solutions considering consistency and availability.
- Develop applications using modern cloud-based web platforms and frameworks.
- Analyze security and privacy implications in distributed and cloud services.

6. COMPETENCES

AG-C09) Design and Development of Solutions: Designs, implements, and evaluates solutions for complex computing problems. (Usage)

AG-C11) Use of Tools: Applies modern computing tools in problem solving. (Usage)

7. TOPICS

Unit 1: Cloud Architecture and Data Management (12 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	Learning Outcomes
• Web platforms, frameworks, or meta-frameworks – Cloud services – API, Web Components • Computing services NoSQL Systems – Cloud Hosting – Scalability (e.g., Autoscaling, Clusters) – Cost estimation for services • Architecture – Monoliths vs Microservices – Micro-frontends – Event-Driven vs RESTful architectures: advantages and disadvantages – Serverless, cloud computing on demand	• Design and implement a web-based application using a microservice architecture design [Design] • Describe the constraints, such as hosting, services, and scalability, related to web platforms [Describe]
Readings : [Marinescu2022], [Buyya2013]	

Unit 2: Distributed Databases/Cloud Computing (10 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	Learning Outcomes
• Distributed databases/cloud-based systems	• Describe the key components of a distributed DBMS, including distributed data storage, query processing, and transaction management [Describe] • Analyze the tradeoffs between parallel DBMS architectures: shared memory, shared disk, and shared nothing [Analyze] • Describe data replication strategies and weak consistency models in distributed database systems [Describe]
Readings : [Buyya2013]	

Unit 3: Parallel and Distributed Computing (8 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	Learning Outcomes
• Safety and liveness: – Race conditions – Dependencies/preconditions – Fault models – Termination • Programming models: – Actor models – Procedural and reactive models – Synchronous/asynchronous programming models – Data parallelism • Properties: – Order-based properties: * Commutativity * Independence – Consistency-based properties: * Atomicity * Consensus • Execution control: – Async await – Promises – Threads • Communication and coordination: – Mutexes – Message-passing – Shared memory – Cobegin-coend – Monitors – Channels – Threads – Guards • Futures • Language support for data parallelism such as forall, loop unrolling, map/reduce • Effect of memory-consistency models on language semantics and correct code generation • Representational State Transfer Application Programming Interfaces (REST APIs) • Technologies and approaches: cloud computing, high performance computing, quantum computing, ubiquitous computing • Overheads of message-passing	• Explain why programming languages do not guarantee sequential consistency in the presence of data races and what programmers must do as a result [Explain] • Implement correct concurrent programs using multiple programming models, such as shared memory, actors, futures, synchronization constructs, and data-parallelism primitives [Implement] • Use a message-passing model to analyze a communication protocol [Use] • Use synchronization constructions such as monitor/synchronized methods in a simple program [Use] • Modeling data dependency using simple programming constructs involving variables, read and write [Implement] • Modeling control dependency using simple constructs such as selection and iteration [Implement] • Explain how REST API's integrate applications and automate processes [Explain] • Explain benefits, constraints and challenges related to distributed and parallel computing [Explain]

Unit 4: Networked Applications (6 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	Learning Outcomes
• Distributed application paradigms (e.g., client/server, peer-to-peer, cloud, edge, and fog). See also: Communication, Coordination	• Define the principles of naming, addressing, resource location [Define] • Analyze the needs of specific networked application demands [Analyze] • Describe the details of one application layer protocol [Describe] • Implement a simple client-server socket-based application [Implement]
Readings : [Marinescu2022]	

Unit 5: Algorithms: Application Domains Survey (4 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	Learning Outcomes
• Survey of common application domains across multicore, reactive, data parallel, cluster, cloud, open distributed systems, and frameworks (with reference to the following table). – Multicore: Typical Execution agents: Threads. Typical Communication mechanisms: Shared memory, Atomics, locks. Typical Algorithmic domains: Resource management, data processing. Typical Engineering goals: Throughput, latency, energy. – Reactive: Typical Execution agents: Handlers, threads. Typical Communication mechanisms: I/O Channels. Typical Algorithmic domains: Services, real-time. Typical Engineering goals: Latency. – Data parallel: Typical Execution agents: GPU, SIMD, accelerators, hybrid. Typical Communication mechanisms: Heterogeneous memory. Typical Algorithmic domains: Linear algebra, graphics, data analysis. Typical Engineering goals: Throughput, energy. – Cluster: Typical Execution agents: Managed hosts. Typical Communication mechanisms: Sockets, channels. Typical Algorithmic domains: Simulation, data analysis. Typical Engineering goals: Throughput. – Cloud: Typical Execution agents: Provisioned hosts. Typical Communication mechanisms: Service APIs. Typical Algorithmic domains: Web applications. Typical Engineering goals: Scalability. – Open distributed: Typical Execution agents: Autonomous hosts. Typical Communication mechanisms: Sockets, Data stores. Typical Algorithmic domains: Fault tolerant data stores and services. Typical Engineering goals: Reliability.	• Extend an event-driven sequential program by establishing a new activity in an event handler (for example a new thread in a GUI action handler) [Design] • Improve the performance of a sequential component by introducing parallelism and/or distribution [Create] • Choose among different parallel/distributed designs for components of a given system [Assess]
Readings : [Marinescu2022]	

Unit 6: Security Mindset and Principles (8 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	Learning Outcomes
• Applications of a security mindset: web, cloud, and mobile devices System Design, Common Aspects: Platforms, APIs and Constraints, Common Aspects: Platform Languages and Web Programming Patterns. • Privacy implications of widespread data collection including but not limited to transactional databases, data warehouses, surveillance systems, cloud computing, and artificial intelligence.	• Analyze the security risks associated with distributed cloud applications [Usar]. • Evaluate the privacy implications of data storage in multi-tenant cloud environments [Evaluar].
Readings : [Marinescu2022]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

Theory Sessions:

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

Lab Sessions:

In order to verify their competences, several activities including active learning and roleplay will be developed during lab sessions.

Oral Presentations:

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

Reading:

Throughout the course different readings are provided, which are evaluated. The average of the notes in the readings is considered as the mark of a qualified practice. The use of our virtual campus allows each student to access the course information, and interact outside the classroom with the teacher and with the other students.

The final grade NF depends on the following grades:

- PE : Partial Exam.
- FE : Final Exam.
- P : Permanent Evaluation (Average of practical work and graded assignments).

The final grade NF is calculated as follows:

$$NF = \frac{PE + FE + P}{3}$$

10. BASIC BIBLIOGRAPHY