National University of Engineering (UNI)
School of Computer Science
Syllabus 2026-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 Prerrequisites	:	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-C11) Use of Tools: Applies modern computing tools in problem solving. (Usage)

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

7. TOPICS

Unit 1: Arquitectura Cloud y Gestión de Datos (12 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	**Learning Outcomes**
• Plataformas, frameworks o meta-frameworks web – Servicios en la nube – API, Componentes Web (Web Components) • Servicios de computación Gestión de Datos (DM)-NoSQLSystems – Alojamiento en la Nube (Cloud Hosting) – Escalabilidad (por ejemplo, Autoescalado (*Autoscaling*), Clústeres) – Estimación de costos para servicios • Arquitectura – Monolitos vs Microservicios – Micro-frontends – Arquitecturas Dirigidas por Eventos (Event-Driven) vs RESTful: ventajas y desventajas – Sin servidor (Serverless), computación en la nube bajo demanda	• Diseñar e implementar una aplicación basada en web utilizando un diseño de arquitectura de microservicios [Diseñar] • Describir las restricciones, como alojamiento, servicios y escalabilidad, relacionadas con las plataformas web [Describir]
Readings : [Mar22], [BVS13]	

Unit 2: Bases de Datos Distribuidas/Computación en la Nube (10 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	**Learning Outcomes**
• Bases de datos distribuidas/sistemas basados en la nube	• Describir los componentes clave de un DBMS distribuido, incluyendo almacenamiento de datos distribuido, procesamiento de consultas y gestión de transacciones [Describir] • Analizar las ventajas y desventajas entre arquitecturas de DBMS paralelo: memoria compartida, disco compartido y nada compartido [Analizar] • Describir estrategias de replicación de datos y modelos de consistencia débil en sistemas de bases de datos distribuidas [Describir]
Readings : [BVS13]	

Unit 3: Computación Paralela y Distribuida (8 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	**Learning Outcomes**
• Seguridad (safety) y vivacidad (liveness): – Condiciones de carrera (race conditions) – Dependencias/precondiciones – Modelos de fallos (fault models) – Terminación • Modelos de programación: – Modelo de actores (*actor models*) – Modelos procedurales y reactivos – Modelos de programación síncrona/asíncrona – Paralelismo de datos (*data parallelism*) • Propiedades: – Propiedades basadas en orden: ∗ Conmutatividad ∗ Independencia – Propiedades basadas en consistencia: ∗ Atomicidad ∗ Consenso • Control de ejecución: – Espera asíncrona *Async await* – Promesas (*promises*) – Hilos (*threads*) • Comunicación y coordinación: – Mutexes – Paso de mensajes (*message-passing*) – Memoria compartida (*shared memory*) – Cobegin-coend – Monitores – Canales (*channels*) – Hilos (*threads*) – Guardas (*guards*) • Futuros (futures) • Soporte del lenguaje para paralelismo de datos como forall, desenrollado de bucles (loop unrolling), map/reduce • Efecto de los modelos de consistencia de memoria en la semántica del lenguaje y la generación correcta de código • Interfaces de Programación de Aplicaciones de Transferencia de Estado Representacional (REST APIs) • Tecnologías y enfoques: computación en la nube (*cloud computing*), computación de alto	• Explicar por qué los lenguajes de programación no garantizan la consistencia secuencial en presencia de condiciones de carrera (data races) y qué deben hacer los programadores como resultado [Explicar] • Implementar programas concurrentes correctos usando múltiples modelos de programación, como memoria compartida, actores, futuros, construcciones de sincronización y primitivas de paralelismo de datos [Implementar] • Usar un modelo de paso de mensajes para analizar un protocolo de comunicación [Usar] • Usar construcciones de sincronización como métodos monitorizados/sincronizados en un programa simple [Usar] • Modelar dependencia de datos usando construcciones de programación simples que involucran variables, lectura y escritura [Implementar] • Modelar dependencia de control usando construcciones simples como selección e iteración [Implementar] • Explicar cómo las APIs REST integran aplicaciones y automatizan procesos [Explicar] • Explicar beneficios, limitaciones y desafíos relacionados con la computación distribuida y paralela [Explicar]

Unit 4: Aplicaciones en Red (6 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	**Learning Outcomes**
• Paradigmas de aplicaciones distribuidas (por ejemplo, cliente/servidor, peer-to-peer, nube, edge y fog). Ver también: Computación Paralela y Distribuida (PDC)-Communication, Computación Paralela y Distribuida (PDC)-Coordination	• Definir los principios de denominación, direccionamiento y localización de recursos [Definir] • Analizar las necesidades de demandas específicas de aplicaciones en red [Analizar] • Describir los detalles de un protocolo de capa de aplicación [Describir] • Implementar una aplicación simple cliente-servidor basada en sockets [Implementar]
Readings : [Mar22]	

Unit 5: Algoritmos: Dominios de Aplicación (4 hours)

Competences Expected: AG-C09,AG-C11

Topics	**Learning Outcomes**
• Panorama de dominios de aplicación comunes en multinúcleo, reactivo, datos paralelos, clúster, nube, sistemas distribuidos abiertos y marcos de trabajo (con referencia a la siguiente tabla). – Multinúcleo: Agentes de ejecución típicos: Hilos. Mecanismos de comunicación típicos: Memoria compartida, Atómicos, bloqueos. Dominios algorítmicos típicos: Gestión de recursos, procesamiento de datos. Objetivos de ingeniería típicos: Rendimiento (*throughput*), latencia, energía. – Reactivo: Agentes de ejecución típicos: Manejadores, hilos. Mecanismos de comunicación típicos: Canales de E/S. Dominios algorítmicos típicos: Servicios, tiempo real. Objetivos de ingeniería típicos: Latencia. – Datos paralelos: Agentes de ejecución típicos: GPU, SIMD, aceleradores, híbridos. Mecanismos de comunicación típicos: Memoria heterogénea. Dominios algorítmicos típicos: álgebra lineal, gráficos, análisis de datos. Objetivos de ingeniería típicos: Rendimiento (*throughput*), energía. – Clúster: Agentes de ejecución típicos: Hosts gestionados. Mecanismos de comunicación típicos: Sockets, canales. Dominios algorítmicos típicos: Simulación, análisis de datos. Objetivos de ingeniería típicos: Rendimiento (*throughput*). – Nube: Agentes de ejecución típicos: Hosts aprovisionados. Mecanismos de comunicación típicos: APIs de servicio. Dominios algorítmicos típicos: Aplicaciones web. Objetivos de ingeniería típicos: Escalabilidad. – Distribuido abierto: Agentes de ejecución típicos: Hosts autónomos. Mecanismos de comunicación típicos: Sockets, Almacenes de datos. Dominios algorítmicos típicos: Almacenes de datos y servicios tolerantes a fallos. Objetivos de ingeniería típicos: Confiabilidad.	• Extender un programa secuencial dirigido por eventos estableciendo una nueva actividad en un manejador de eventos (por ejemplo, un nuevo hilo en un manejador de acciones de GUI) [Diseñar] • Mejorar el rendimiento de un componente secuencial introduciendo paralelismo y/o distribución [Crear] • Elegir entre diferentes diseños paralelos/distribuidos para componentes de un sistema dado [Evaluar (valorar)]

Readings : [Mar22]

Unit 6: Principios y Mentalidad de Seguridad (8 hours)	
Competences Expected: AG-C09,AG-C11	
Topics	**Learning Outcomes**
• Aplicaciones de una mentalidad de seguridad: web, nube y dispositivos móviles Fundamentos de Sistemas (SF)-Design,SPD-CommonAspectsSharedConcerns. • Implicaciones de privacidad de la recopilación generalizada de datos, incluyendo pero no limitado a bases de datos transaccionales, almacenes de datos, sistemas de vigilancia, computación en la nube e inteligencia artificial.	• Analyze the security risks associated with distributed cloud applications [Usar]. • Evaluate the privacy implications of data storage in multi-tenant cloud environments [Evaluar].
Readings : [Mar22]	

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

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[BVS13] Rajkumar Buyya, Christian Vecchiola, and S. Thamarai Selvi. *Mastering Cloud Computing: Foundations and Applications Programming*. 1st. Morgan Kaufmann, 2013.

[Mar22] Dan C. Marinescu. *Cloud Computing: Theory and Practice*. 3rd. Morgan Kaufmann, 2022.