



| 1. DATOS DE IDENTIFICACIÓN   |   |                          |                           |
|--|---|--------------------------|---------------------------|
| UNIDAD DE APRENDIZAJE O MÓDULO:  | Desarrollo e implementación de sistemas   |                          |                           |
| Clave:   | 19405   |                          |                           |
| Ubicación:   | Periodo: 4  | Área: Básico Disciplinar |                           |
| Horas y créditos:  | Teóricas: 80  | Prácticas: 56            | Estudio Independiente: 40 |
|  | Total, de horas: 176  |                          | Créditos: 11              |
| Competencia(s) del perfil de egreso al que aporta:   | CE2. Desarrolla la capacidad para adquirir, obtener, formalizar y representar el conocimiento humano en una forma computable para la resolución de problemas mediante la automatización y sistemas informáticos en cualquier ámbito de aplicación, particularmente los relacionados con aspectos de computación. Además, cultiva conocimientos básicos sobre el uso de sistemas operativos.<br>CE5. Aplica técnicas de aprendizaje computacional y sistemas de información con criterios de calidad para su desarrollo y operación basado en conocimientos de métodos computacionales y teorías de la informática para resolver problemas en los diferentes sectores. |                          |                           |
| Unidades de aprendizaje relacionadas:  | Programación estructurada, Manejadores y lenguajes de consulta de datos, Fundamentos de base de datos, Paradigmas de programación, Programación para procesamiento de datos, Análisis y diseño de software, Entornos de desarrollo para programación, Pruebas de Software.  |                          |                           |
| Responsable(s) de elaborar el programa:  | Dra. Gladys Azucena Bernal Salgueiro<br>Dr. Adrián Torres Sánchez<br>Dr. Carlos Alberto López Iribe<br>Dr. Fidel Bojórquez Solís<br>Dr. Jorge Anselmo Rodríguez Castro  |                          | Fecha:<br>Enero 2025      |
| Responsable(s) de actualizar el programa:  |   |                          | Fecha:                    |
| 2. PROPÓSITO   |   |                          |                           |
| El curso de Desarrollo e Implementación de Sistemas tiene como objetivo formar profesionistas capaces de diseñar, desarrollar, implementar y evaluar sistemas informáticos que den solución a problemas reales en diversos contextos organizacionales. Se busca que los estudiantes adquieran competencias tanto técnicas como actitudinales, que les permitan actuar con responsabilidad, ética y compromiso en el desarrollo de soluciones tecnológicas. |   |                          |                           |
| 3. SABERES   |   |                          |                           |
| Teóricos:  | ● Definición y componentes de un sistema de información.  |                          |                           |



|            |  |
|------------|--|
|            | <ul style="list-style-type: none"><li>● Importancia de los sistemas de información en las organizaciones.</li><li>● Clasificación de sistemas de información: operativos, administrativos, estratégicos.</li><li>● Etapas del SDLC: planificación, análisis, diseño, implementación y mantenimiento.</li><li>● Ventajas y limitaciones del enfoque estructurado del SDLC.</li><li>● Características principales de los modelos Cascada, Agile y DevOps.</li><li>● Diferencia entre requisitos funcionales y no funcionales.</li><li>● Conceptos fundamentales de la orientación a objetos: encapsulación, herencia, polimorfismo y abstracción.</li><li>● Conceptos y uso de los diagramas UML en el modelado de sistemas.</li><li>● Reglas de construcción de los diagramas de casos de uso, clases y secuencia.</li><li>● Concepto y estructura de bases de datos relacionales.</li><li>● Diferencias entre UX (experiencia de usuario) y UI (interfaz de usuario).</li><li>● Ventajas del prototipado en el proceso de diseño de sistemas.</li><li>● Funcionalidades clave de herramientas como Figma y Axure.</li><li>● Fundamentos de programación: variables, estructuras de control y funciones.</li><li>● Conceptos básicos de control de versiones: repositorio, commit, branch y merge.</li><li>● Características principales de MySQL y PostgreSQL.</li><li>● Conceptos básicos del modelo cliente-servidor.</li><li>● Importancia de la documentación en el desarrollo de sistemas.</li><li>● Definición y objetivos de las pruebas unitarias, funcionales, de integración y de usuario.</li><li>● Conceptos básicos de la automatización de pruebas: scripts, herramientas y frameworks.</li><li>● Tipos de errores comunes en programación: sintácticos, lógicos y de ejecución.</li><li>● Métodos para evaluar la experiencia del usuario: encuestas, pruebas de usabilidad y análisis heurístico.</li><li>● Elementos básicos de un informe de pruebas: objetivos, casos de prueba, resultados y conclusiones.</li></ul> |
| Prácticos: | <ul style="list-style-type: none"><li>● Identificación de sistemas de información existentes en organizaciones locales.</li><li>● Análisis de casos prácticos sobre el impacto de los sistemas de información en la toma de decisiones.</li><li>● Elaboración de un esquema visual del SDLC.</li><li>● Aplicación del SDLC a un caso práctico sencillo, como el desarrollo de un sistema básico de gestión.</li><li>● Simulación de un proyecto sencillo utilizando un enfoque en cascada.</li><li>● Creación de un formato de captura de requisitos.</li><li>● Análisis y documentación de requisitos a partir de un caso práctico.</li><li>● Funcionalidades clave de herramientas como Trello y Jira.</li><li>● Creación de un proyecto básico en Jira con tareas asignadas, prioridades y etiquetas.</li><li>● Diseño de una estructura básica de clases para resolver un problema sencillo.</li><li>● Ejercicios prácticos de identificación de objetos, atributos y métodos en un caso real.</li><li>● Elaboración de un diagrama de casos de uso para un sistema práctico, como una tienda en línea.</li><li>● Diseño de un diagrama entidad-relación para un sistema práctico, como una biblioteca.</li></ul>  |



|                |  |
|----------------|--|
|                | <ul style="list-style-type: none"><li>● Análisis de interfaces existentes para identificar fortalezas y debilidades.</li><li>● Creación de un prototipo interactivo básico utilizando Figma.</li><li>● Creación de programas básicos en Java y Python para entender estructuras de control y manejo de datos.</li><li>● Instalación y configuración inicial de Git en un entorno local.</li><li>● Instalación y configuración de MySQL o PostgreSQL en un entorno local o en la nube.</li><li>● Configuración de un servidor backend para procesar solicitudes y enviar respuestas.</li><li>● Uso de herramientas como Javadoc o Docstrings para generar documentación técnica.</li><li>● Identificación de casos de prueba para cada tipo de prueba en un sistema sencillo.</li><li>● Instalación y configuración de Selenium para pruebas en aplicaciones web.</li><li>● Identificación y corrección de errores en un fragmento de código con fallos conocidos.</li><li>● Aplicación de pruebas de usabilidad en un prototipo o sistema terminado.</li></ul> |
| Actitudinales: | <ul style="list-style-type: none"><li>● Colabora con trabajo en equipo para fortalecer sus capacidades y habilidades Disponibilidad, interés e iniciativa para el aprendizaje.</li><li>● Busca y comparte ejemplos prácticos sobre contenidos del programa.</li><li>● Asume y desarrolla una postura ética de liderazgo.</li><li>● Participa con responsabilidad y respeto en dinámicas de trabajo colaborativo.</li><li>● Cumplir de forma disciplinada con sus tareas académicas.</li><li>● Adopta una postura analítica y crítica para realizar las actividades prácticas del curso. Construye su conocimiento, procesa datos, manejo eficiente y oportuno de la información y así aplicarlo en la definición de estrategias básicas empresariales.</li></ul>   |

#### 4. CONTENIDOS

##### Unidad 1: Introducción al Desarrollo de Sistemas

- 1.1 Introducción a los sistemas de información.
- 1.2 El ciclo de vida del desarrollo de sistemas (SDLC).
- 1.3 Modelos de desarrollo: Cascada, Agile y DevOps.
- 1.4 Análisis de requisitos funcionales y no funcionales.
- 1.5 Introducción a herramientas para la gestión de proyectos (Trello, Jira).

##### Unidad 2: Diseño de Sistemas

- 2.1 Principios de diseño de software orientado a objetos.
- 2.2 Diagramas UML: Casos de uso, clases y secuencia.
- 2.3 Diseño de bases de datos relacionales: normalización y diagramas entidad-relación.
- 2.4 Diseño de interfaces centradas en el usuario (UX/UI).
- 2.5 Prototipado con herramientas (Figma, Axure).

##### Unidad 3: Desarrollo de Sistemas



- 3.1 Introducción a lenguajes de programación (Java, Python).
- 3.2 Control de versiones con Git y GitHub.
- 3.3 Implementación de bases de datos con MySQL o PostgreSQL.
- 3.4 Conexión entre backend y frontend.
- 3.5 Documentación del código y de las funcionalidades del sistema.

#### Unidad 4: Pruebas y Validación de Sistemas

- 4.1 Tipos de pruebas: unitarias, funcionales, de integración y de usuario.
- 4.2 Automatización de pruebas con herramientas (Selenium, JUnit).
- 4.3 Gestión de errores y depuración de código.
- 4.4 Evaluación de la experiencia del usuario (UX).
- 4.5 Elaboración de informes de pruebas y métricas de calidad.

#### 5. ACTIVIDADES PARA DESARROLLAR LAS COMPETENCIAS

##### Actividades del docente:

- Actividades de inicio: Exposición, conferencia, la pregunta, elaborar video.
- Actividad de desarrollo: investigación bibliográfica, lección, exposición, explicación en video, ejemplificación de simulaciones de los componentes de las redes, prácticas demostrativas.
- Actividad de evaluación: Mapa conceptual, cuadro sinóptico, cuadro comparativo, portafolio de evidencias, rúbrica.

##### Actividades del estudiante:

- ❖ Actividades de inicio: video, lección, encuesta diagnóstica.
- ❖ Actividades de desarrollo: ejercicios, tarea, ensayo, resolver problemas, glosario, visualización de contenidos interactivos, elaborar tareas, Investigación, análisis de casos y solución de problemas Foros de discusión y proyectos colaborativos.
- ❖ Actividades de cierre: Informe de investigación documental, mapa conceptual, cuadro sinóptico, cuadro comparativo, portafolio de evidencias, rúbrica.

#### 6. EVALUACIÓN DE LAS COMPETENCIAS

##### 6.1. Criterios de desempeño

- Aplica técnicas de modelado (diagramas UML, casos de uso) para representar los requisitos del sistema.
- Diseña diagramas UML que reflejen correctamente la estructura y comportamiento del sistema.
- Implementa código funcional en un lenguaje de programación adecuado (Java, Python, etc.), aplicando buenas prácticas de desarrollo.

##### 6.2 Portafolio de evidencias

- Informe de análisis de sistemas de información.
- Mapa conceptual del SDLC.
- Análisis comparativo de modelos de desarrollo
- Reporte de uso de herramientas de gestión de proyectos (Trello, Jira).
- Diagramas UML.
- Código fuente de un módulo funcional.
- Plan de pruebas.



- Diseña y ejecuta pruebas unitarias, de integración, funcionales y de usuario, documentando los resultados.
- Aplica técnicas de modelado (diagramas UML, casos de uso) para representar los requisitos del sistema.

- Guía de instalación y configuración.
- Entrega final del sistema funcionando.

#### 6.3. Calificación y acreditación:

##### Parcial:

- 10% de asistencia
- 50% Calificación en exámenes parciales
- 20 % Exposición de Temas/Estudios de casos de ejemplo/ trabajo en equipo/Proyecto final.
- 20% Tareas.

##### NOTA ACLARATORIA:

Debe cumplir por lo menos con 80 % de la asistencia para tener derecho a presentar evaluación parcial y ordinaria.

Para tener derecho a solicitar extraordinario deberá reunir al menos 60 % de la asistencia en el semestre.  
Asistencia y Participación 10%

#### 7. RECURSOS DIDÁCTICOS

##### Recursos Electrónicos:

1. Artículos y publicaciones científicas:
  - Revistas académicas sobre desarrollo de software, gestión de proyectos y tecnologías emergentes.
  - Casos prácticos publicados en plataformas como IEEE Xplore y ACM Digital Library.
2. Guías y estándares:
  - Estándares ISO/IEC 25010 para calidad de software.
  - Documentación oficial de herramientas y lenguajes de programación utilizados en el curso.
3. Entornos de desarrollo integrados (IDE):
  - Visual Studio Code, IntelliJ IDEA, PyCharm, Eclipse.
4. Herramientas de control de versiones:
  - Git y plataformas como GitHub, GitLab o Bitbucket.
5. Herramientas de diseño y modelado:
  - Lucidchart o Draw.io para diagramas UML.
  - Figma, Adobe XD o Axure para prototipos de interfaces UX/UI.
6. Bases de datos y servidores:
  - MySQL, PostgreSQL, MongoDB para la implementación de bases de datos.
  - Docker para entornos de prueba y despliegue.
7. Herramientas de pruebas y validación:
  - Selenium para pruebas automatizadas.
  - JUnit para pruebas unitarias.
  - Postman para pruebas de APIs.

##### Recursos Materiales:



Equipo Multimedia, pintarrón, plumones.

#### 8. FUENTES DE INFORMACIÓN

##### Bibliografía básica

| Autor(es)       | Título  | Editorial       | Año     | URL o biblioteca digital donde está disponible |
|-----------------|---|-----------------|---------|--|
| Sommerville, I. | <i>Ingeniería de Software.</i>  | Pearson.        | (2015). | Libro  |
| Pressman, R. S. | <i>Ingeniería de Software: Un Enfoque Práctico.</i>                           | McGraw-Hill.    | (2020). | Libro  |
| Fowler, M.      | <i>UML Distilled: A Brief Guide to the Standard Object Modeling Language.</i> | Addison-Wesley. | (2004). | Libro  |

#### 9. PERFIL DEL DOCENTE

Contar con estudios de Licenciatura / Ingeniería ya sea en Informática o Sistemas, preferentemente en Telecomunicaciones, y de ser posible en el mejor de los casos con Posgrado en el Área, contar con Título con grado académico y con experiencia en instalación de cableado de redes. Tener experiencia frente a grupo, así como facilidad de palabra.