CURRÍCULO DE LA FIGURA PROFESIONAL "DESARROLLO DE SOFTWARE"

1. Objetivo general

Desarrollar soluciones informáticas mediante el análisis, diseño, codificación, pruebas, mantenimiento y documentación, aplicando metodologías de desarrollo, conexión a bases de datos, marcos de trabajo y herramientas colaborativas, conforme a la normativa vigente según el enfoque de la aplicación, con comunicación, trabajo en equipo, pensamiento crítico, responsabilidad y adaptabilidad frente a entornos tecnológicos cambiantes.

2. Plan de estudios

Total periodos pedagógicos tronco común		**1ro**	**2do**	**3ro**
		19	**19**	**19**
Módulos Genéricos de la Familia Profesional	Fundamentos de las Tecnologías de la Información y la Comunicación	3	2	
	Pensamiento Computacional y Resolución de Problemas	3	3	
	Ética, Legislación y Ciudadanía Digital	2		
Módulos Especialización	Programación Estructurada	6		
	Programación Orientada a Objetos		5	5
	Base de Datos	2	3	4
	Aplicaciones de Escritorio	3	3	3
	Aplicaciones Web y Móviles		3	6
Módulo práctico/experimental		2	2	3
Total de períodos pedagógicos de formación técnica		**21**	**21**	**21**

3. Módulos genéricos

Durante el primer y segundo año de formación, el estudiante desarrolla competencias genéricas vinculadas a la Familia Profesional Tecnologías. Gracias a las características de los módulos trabajados en esta etapa, el estudiante adquiere herramientas que le permiten construir una opinión más informada y tomar decisiones con mayor fundamento. Esto favorece su capacidad para, en caso de que lo desee, transitar entre distintas figuras profesionales dentro de la misma familia, continuar con su trayectoria educativa, insertarse en el mundo laboral o emprender un proyecto propio.

Se estructuran los siguientes módulos genéricos:

- Fundamentos de las Tecnologías de la Información y la Comunicación
- Pensamiento Computacional y Resolución de Problemas
- Ética, Legislación y Ciudadanía Digital

Módulo Genérico Nro. 1	
Nombre del módulo:	Fundamentos de las Tecnologías de la Información y la Comunicación
Nivel:	1ro, 2do
Duración:	200 periodos pedagógicos
Unidad de competencia asociada:	**UC1:** Aplicar fundamentos de tecnologías de la información y la comunicación mediante el análisis de su evolución, la identificación de la arquitectura de hardware, software y principios básicos de electricidad y electrónica, la operación básica de redes y sistemas operativos y, la gestión de información digital, con el fin de resolver requerimientos tecnológicos.
Objetivo del módulo: Desarrollar en el estudiante competencias que analice la evolución de las tecnologías de la información y la comunicación, identifique la arquitectura, el funcionamiento de hardware y software, manipule sistemas de redes y sistemas operativos, organice la información digital, con el fin de resolver de manera efectiva requerimientos en distintos entornos tecnológicos.	
Resultados de Aprendizaje (RA) y Criterios de Evaluación (CE)	
RA.1. Analizar la evolución de las tecnologías de la información y la comunicación reconociendo sus hitos históricos, transformaciones tecnológicas y su impacto en los entornos productivos, educativos y sociales. **CE1.1:** Distingue la evolución de las TIC con base en líneas de tiempo o esquemas cronológicos. **CE1.2:** Emplea los avances tecnológicos con sus aplicaciones en distintos sectores de la sociedad. **CE1.3:** Define las generaciones de computadoras y dispositivos en función de su capacidad y arquitectura. **CE1.4:** Argumenta el impacto social, económico y cultural del uso de las TIC, mediante análisis de casos actuales o discusiones dirigidas. **RA.2 Examinar la arquitectura de sistemas informáticos, identificando componentes de hardware, software y su funcionamiento integrado en distintos entornos tecnológicos.** **CE2.1:** Inspecciona los componentes físicos de un sistema informático, clasificándolos según su función principal (entrada, procesamiento, almacenamiento, salida y comunicación). **CE2.2:** Distingue entre software de sistema, de aplicación y de desarrollo, explicando su propósito funcional con ejemplos representativos. **CE2.3:** Elabora esquemas de arquitectura de computadoras donde ubicar los componentes funcionales y describe el flujo básico de información entre ellos. **CE2.4:** Determina el tipo de sistema informático (doméstico, industrial, educativo, entre otros) a partir de su estructura y características técnicas, según el contexto de uso. **RA.3 Emplear sistemas operativos y entornos digitales gestionando recursos, configuración inicial y funciones elementales en la administración del sistema**.	

CE3.1: Ejecuta tareas básicas de administración de archivos y carpetas, diferenciando procedimientos entre sistemas operativos gráficos y basados en línea de comandos.
CE3.2: Identifica parámetros esenciales del sistema operativo como nombre de usuario, configuración de red y capacidad de almacenamiento, utilizando herramientas del entorno.
CE3.3: Aplica funciones básicas de herramientas de mantenimiento del sistema operativo para optimizar su funcionamiento.
CE3.4: Emplea instrucciones en los entornos gráficos o de línea de comandos según el tipo de operación administrativa requerida, justificando su elección.
RA.4. Utiliza los fundamentos básicos de electricidad y electrónica digital para la construcción de circuitos simples.
CE4.1: Examina los principios básicos de electricidad y su relación con el funcionamiento de dispositivos electrónicos utilizados en computación.
CE4.2: Analiza componentes electrónicos digitales y su aplicación en sistemas informáticos.
CE4.3: Establece conocimientos de lógica en la resolución de problemas digitales.
CE4.4: Diseña esquemas y diagramas electrónicos básicos para comprender la estructura de circuitos eléctricos simples.
RA.5 Diseñar redes informáticas comprendiendo su estructura, tipos, dispositivos, protocolos de comunicación y funciones esenciales.
CE5.1: Clasifica elementos físicos y lógicos de una red informática según su función en el sistema.
CE5.2: Diferencia tipos de redes (LAN, WAN, WLAN) en función de su alcance, propósito y características.
CE5.3: Representa dispositivos de red en esquemas funcionales según su rol en la infraestructura.
CE5.4: Elabora cuadros técnicos sobre el funcionamiento de protocolos de comunicación en la transmisión de datos.
RA.6. Gestionar información digital utilizando herramientas tecnológicas mediante la organización, almacenamiento, protección y compartición de datos.
CE6.1: Analiza información digital considerando su tipo, relevancia y nivel de confidencialidad en contextos educativos o profesionales.
CE6.2: Organiza carpetas y archivos con nombres coherentes en soportes físicos y virtuales con base en criterios de estructuración lógica.
CE6.3: Implementa medidas básicas de protección de datos mediante contraseñas, permisos de acceso o cifrado en entornos digitales.
CE6.4: Transfiere información en plataformas colaborativas o servicios en la nube respetando estándares de seguridad y compatibilidad.

Contenidos		
Conceptuales	**Procedimentales**	**Actitudinales**
Evolución histórica de las Tecnologías de la Información y la Comunicación (TIC).	Analizar líneas de tiempo y esquemas cronológicos para identificar hitos y evolución de las TIC.	Mostrar responsabilidad y ética en el manejo y protección de la información digital.

Hitos tecnológicos y generaciones de computadoras. Impacto social, económico y cultural de las TIC. Software libre y software propietario, licenciamiento y pagos. Arquitectura de sistemas informáticos: componentes de hardware y sus funciones. Tipos de software: de sistema, de aplicación y de desarrollo, según su contexto, funciones y características. Estructura funcional de computadoras y flujo de información entre componentes. Clasificación de sistemas informáticos según su contexto de uso. Redes informáticas, elementos físicos y lógicos. Tipos de redes (LAN, WAN, WLAN) y sus características principales. Dispositivos de red y sus funciones (switch, router, módem, punto de acceso, repetidor). Protocolos de comunicación en redes. Conceptos de gestión, organización, almacenamiento, normas y	Clasificar dispositivos de hardware y software según su función y tipo. Organizar esquemas y diagramas de arquitectura de sistemas informáticos. Identificar el flujo de información entre componentes de hardware y software. Manipular sistemas operativos para gestionar archivos, carpetas y configuraciones básicas. Configurar parámetros de usuario, red y almacenamiento en sistemas operativos. Aplicar funciones básicas de mantenimiento del sistema operativo. Utilizar entornos gráficos y línea de comandos para tareas de administración. Catalogar y diferenciar elementos físicos y lógicos de redes informáticas. Describir tipos de redes y sus características mediante ejemplos prácticos. Configurar dispositivos de red y simular funciones de protocolos de comunicación. Organizar, almacenar y proteger información digital mediante herramientas tecnológicas.	Valorar la importancia de la evolución tecnológica y su impacto en la sociedad. Desarrollar una actitud crítica y reflexiva ante los cambios y avances en las TIC. Fomentar el interés por el aprendizaje continuo y actualización en tecnologías emergentes. Promover el trabajo colaborativo y la comunicación efectiva en ambientes tecnológicos. Adoptar prácticas de seguridad digital para proteger datos personales y profesionales. Reconocer la importancia del respeto a la privacidad y la confidencialidad en el uso de tecnologías. Desarrollar autonomía en la gestión y solución de problemas tecnológicos básicos. Mantener una postura abierta y flexible ante nuevas metodologías y herramientas tecnológicas. Impulsar la conciencia sobre el impacto ambiental y social del uso responsable de las TIC. Mantener una actitud positiva y resiliente ante fallos y errores técnicos.

buenas prácticas en seguridad informática y protección de datos. Herramientas tecnológicas para compartir información en entornos digitales. Fundamentos de trabajo colaborativo y comunicación efectiva en entornos tecnológicos. Organización, planificación y gestión de proyectos tecnológicos. Unidades de medidas de la información. Fundamentos eléctricos y electrónicos: conceptos básicos y práctica con circuitos. Protección eléctrica y puesta a tierra. Componentes eléctricos y electrónicos. Esquemas, diagramas y circuitos electrónicos Lógica digital.	Implementar medidas básicas de seguridad en entornos digitales (contraseñas, permisos, cifrado). Compartir información en plataformas colaborativas y servicios en la nube, respetando normas de seguridad y formatos. Probar nuevas herramientas tecnológicas y entornos digitales. Diagnosticar y resolver problemas técnicos con actitud analítica y perseverante. Documentar procesos y resultados técnicos de manera clara y ordenada. Utilizar plataformas colaborativas para la gestión del trabajo en equipo. Identificar componentes eléctricos y electrónicos básicos visualmente y por código. Construir circuitos eléctricos básicos en protoboard y simuladores digitales.	Fomentar el trabajo en equipo con comunicación abierta, respeto y cooperación. Priorizar la gestión adecuada del tiempo y recursos para cumplir objetivos. Demostrar responsabilidad en el manejo seguro de herramientas e instrumentos eléctricos y electrónicos.
Perfil del o la docente		
• Experiencia en el área técnica, poseer título de tercer o cuarto nivel, registrados y reconocidos por el órgano rector del Sistema de Educación Superior en: Tecnologías de la Información, Ciencias de la Computación, Ingeniería en Sistemas, Ingeniería en Software, Tecnologías de la Información y Comunicación, o ramas afines • Experiencia en el campo amplio de la Educación, debidamente certificada.		
Orientaciones Metodológicas		
• Aprendizaje Basado en Proyectos (ABP)		

<table>
<tr><td colspan="3">

- Aprendizaje en Contextos Reales
- Role-Playing y Simulaciones
- Metodología STEAM (Integración de Ciencia, Tecnología, Arte y Matemáticas)

</td></tr>
<tr><td colspan="3">Materiales y recursos</td></tr>
<tr><td>Denominación</td><td>Especificaciones técnicas</td><td>Cantidad</td></tr>
<tr><td>Infraestructura/espacio</td><td>Entorno de aprendizaje (aula)</td><td>1</td></tr>
<tr><td>Laboratorio</td><td>Computadoras con acceso a internet
Proyector</td><td>1</td></tr>
<tr><td colspan="3">Referencias Bibliográficas</td></tr>
<tr><td colspan="3">

Libros:

- García, R., & Pérez, M. (2022). Introducción a las Tecnologías de la Información y la Comunicación: Bases y Aplicaciones. Editorial Alfaomega, México.
- Sánchez, L. (2021). Redes de Computadoras para Principiantes. Editorial McGraw-Hill, España.
- Morales, J. (2020). Fundamentos de Hardware y Software. Editorial Pearson, España.
- Torres, A. (2023). Sistemas Operativos: Principios y Administración. Editorial Marcombo, España.
- Ramírez, C. (2019). Tecnologías de la Información y Comunicación en el Mundo Actual. Editorial Trillas, México.

Sitios Web:

- Ministerio de Educación del Ecuador (2023). Currículo Nacional para el Bachillerato Técnico: Fundamentos de las Tecnologías de la Información y la Comunicación. Quito, Ecuador.
- Ministerio de Telecomunicaciones y de la Sociedad de la Información (MINTEL) (2020). Política Nacional de Tecnologías de Información y Comunicación. Quito, Ecuador.
- Educatina. https://educatina.com
- Khan Academy en español. https://es.khanacademy.org
- AulaFacil TIC. https://aulafacil.com/cursos/tecnologia-informacion-comunicacion

</td></tr>
</table>

Módulo Genérico Nro.2	
Nombre del módulo:	Pensamiento Computacional y Resolución de Problemas
Nivel:	1ro, 2do
Duración:	240 periodos pedagógicos
Unidad de competencia asociada:	**UC2:** Proponer soluciones informáticas y computacionales mediante el análisis lógico-matemático, manejo de sistemas numéricos, lógica de conjuntos, formulación algorítmica con diagramas y pseudocódigo, aplicación de programación básica,

	levantando documentación técnica, como fundamento del desempeño profesional en tecnologías de la información y comunicación.

Objetivo del módulo: Fortalecer en el estudiante competencias técnicas en la formulación y representación algorítmica, programación básica, aplicando metodologías de desarrollo, conceptos lógicos matemáticos y sistemas numéricos para la resolución de problemas computacionales.

Resultados de aprendizaje (RA) y Criterios de Evaluación (CE)

RA.1 Aplicar conceptos de lógica matemática, conjuntos y sistemas numéricos en la resolución de problemas computacionales.

CE1.1: Distingue proposiciones y operaciones lógicas utilizando tablas de verdad y leyes del álgebra booleana.

CE1.2: Representa operaciones de conjuntos en problemas computacionales mediante diagramas y notación formal.

CE1.3: Transforma valores entre sistemas numéricos binario, decimal, hexadecimal y octal como base para el tratamiento de datos.

CE1.4: Resuelve problemas que combinan lógica proposicional y sistemas numéricos en contextos computacionales específicos.

RA.2 Resolver algoritmos de aplicaciones informáticas, representarlos mediante diagramas de flujo y pseudocódigo.

CE2.1: Descompone problemas en pasos secuenciales mediante la elaboración de diagramas de flujo estructurados.

CE2.2: Traduce diagramas de flujo a pseudocódigo utilizando sintaxis estructurada y lógica funcional.

CE2.3: Verifica algoritmos mediante pruebas y simulaciones, identificando errores de lógica y eficiencia.

CE2.4: Contrasta resultados esperados y obtenidos para evaluar el comportamiento funcional del algoritmo.

RA.3 Utilizar programación en la construcción de soluciones computacionales.

CE3.1: Emplea estructuras secuenciales, condicionales y repetitivas al desarrollar programas en un lenguaje de programación resolviendo problemas computacionales básicos.

CE3.2: Construye clases, atributos y métodos aplicando principios de encapsulamiento, herencia y polimorfismo en un lenguaje orientado a objetos.

CE3.3: Elabora estructuras de datos simples como vectores, listas y matrices para la gestión de información en aplicaciones básicas.

CE3.4: Codifica programas simples verificando la funcionalidad conforme a los requerimientos establecidos y resultados esperados.

CE3.5: Documenta mediante comentarios técnicos explicando la lógica, estructura y funcionamiento del código fuente.

RA.4 Diseñar interfaces básicas facilitando la interacción con usuarios en aplicaciones de consola y GUI.

CE4.1: Aplica elementos visuales, etiquetas y campos adecuados al tipo de aplicación (consola o GUI) para representar la funcionalidad esperada.
CE4.2: Integra principios de usabilidad y accesibilidad en el diseño de interfaces, considerando la disposición, navegación y comprensión visual
CE4.3: Programa eventos y validaciones que responden correctamente a las acciones del usuario, previniendo errores en tiempo de ejecución.
CE4.4: Realiza pruebas para identificar mejoras en la interacción y funcionalidad de la interfaz desarrollada.

RA.5 Implementar metodologías de desarrollo de software en función del tipo de proyecto, integrando tiempos de entrega y colaboración en equipo.

CE5.1: Selecciona adecuadamente la metodología de desarrollo (ágil, tradicional, híbrida) según los requerimientos del proyecto.
CE5.2: Realiza las tareas de desarrollo de acuerdo con la metodología seleccionada, cumpliendo entregables y estándares.
CE5.3: Coordina la colaboración en equipo utilizando herramientas digitales que facilitan la comunicación y el seguimiento de actividades.
CE5.4: Documenta el proceso de desarrollo mediante registros técnicos que evidencian el avance y la gestión del proyecto.

Contenidos

Conceptuales	**Procedimentales**	**Actitudinales**
Principios del pensamiento computacional. Componentes de un problema computacional. Criterios de optimización en soluciones computacionales. Metodología para la resolución de problemas computacionales. Identificación de patrones y relaciones. Algoritmos, definición y características. Tipos de datos, variables, constantes y expresiones. Lenguajes de pseudocódigo y diagramas de flujo.	Descomponer problemas complejos en subproblemas manejables mediante estrategias de pensamiento lógico. Identificar patrones comunes en distintos problemas para proponer soluciones generalizables. Representar procesos a través de diagramas de flujo, pseudocódigo u otras formas de modelado lógico. Formular algoritmos eficientes considerando secuencia, selección, iteración y modularidad. Simular algoritmos en entornos digitales o físicos para comprobar su funcionalidad.	Emplear el pensamiento analítico en la resolución de problemas. Mostrar disposición para explorar situaciones problemáticas desde múltiples perspectivas antes de proponer soluciones. Asumir responsabilidad en la coherencia lógica de los diseños propuestos, considerando el contexto del problema. Mostrar flexibilidad para ajustar los modelos diseñados ante nuevas condiciones o sugerencias. Aplicar buenas prácticas en la elaboración de algoritmos organizados, claros y eficientes.

Estructuras de control secuencial, condicional y repetitiva. Tipos de errores, sintácticos, lógicos y de ejecución. Buenas prácticas en la resolución de problemas computacionales Heurísticas para la toma de decisiones. Operadores lógicos y tablas de verdad. Funciones de modelamiento de bases de datos. Fundamentos de lógica y matemáticas aplicadas, operadores lógicos, proposiciones, tablas de verdad, operaciones entre conjuntos, sistemas numéricos y conversiones. Tipos de lógica, proposicional, booleana y matemática. Interfaz gráfica de usuario GUI, usabilidad y accesibilidad. Metodologías de desarrollo, herramientas colaborativas y documentación técnica y de usuario.	Evaluar soluciones en función de su eficiencia, claridad y capacidad de resolución del problema. Aplicar heurísticas para resolver problemas donde no existe un camino único o completamente definido. Construir soluciones computacionales básicas empleando herramientas de programación. Revisar errores lógicos o sintácticos en algoritmos o programas mediante procesos de depuración. Integrar estrategias de mejora continua en la solución de problemas computacionales. Verificar sistemáticamente la funcionalidad lógica de las soluciones antes de validarlas. Aplicar operadores lógicos en la resolución de problemas computacionales. Transformar números entres distintos sistemas numéricos, entendiendo los procesos computacionales. Diseñar interfaces graficas con elementos interactivos aplicando principios de usabilidad y accesibilidad con enfoque al usuario.	Reflexionar sobre los resultados obtenidos como una oportunidad para mejorar las soluciones propuestas. Aceptar con apertura la retroalimentación de otros como parte fundamental del proceso de aprendizaje. Mostrar comunicación asertiva en la resolución de problemas computacionales, Colaborar activamente con sus pares para enriquecer ideas y construir soluciones colectivas. Valorar la importancia de la lógica matemática en la resolución de problemas. Adoptar una actitud perseverante en la resolución de problemas con algoritmos claros y eficientes. Demostrar empatía hacia el usuario final en el diseño de interfaces. Fomentar la colaboración de trabajo en equipo

	Utilizar herramientas colaborativas para el trabajo en equipo. Elaborar documentación técnica y de usuario del software.	

Perfil del o la docente

- Experiencia en el área técnica, poseer título de tercer o cuarto nivel, registrados y reconocidos por el órgano rector del Sistema de Educación Superior en: Tecnologías de la Información, Ciencias de la Computación, Ingeniería en Sistemas, Ingeniería en Software, Tecnologías de la Información y Comunicación, o ramas afines
- Experiencia en el campo amplio de la Educación, debidamente certificada.

Orientaciones Metodológicas

- Aprendizaje Basado en Proyectos (ABP)
- Aprendizaje en Contextos Reales
- Role-Playing y Simulaciones
- Metodología STEAM (Integración de Ciencia, Tecnología, Arte y Matemáticas)

Requisitos básicos de infraestructuras, espacio y equipamiento:

Denominación	Especificaciones técnicas	Cantidad
Infraestructura/espacio	Entorno de aprendizaje (aula)	1
Laboratorio	Computadoras con acceso a internet Proyector	1

Referencias Bibliográficas

Libros:

- López, A., & Torres, J. (2022). Algoritmos y Programación para Principiantes. Editorial Pearson, México.
- Mendoza, F. (2021). Pensamiento Computacional: Fundamentos y Aplicaciones. Editorial Universitaria, Chile.
- Castro, L. (2023). Introducción a la Programación y Lógica Computacional. Editorial Alfaomega, México.
- Jiménez, R. (2019). Estructuras de Datos y Algoritmos Básicos. Editorial McGraw-Hill, España.
- Hernández, P. (2020). Resolución de Problemas con Programación. Editorial Trillas, México.

Sitios Web:

- Ministerio de Educación del Ecuador (2023). Currículo Nacional para el Bachillerato Técnico: Pensamiento Computacional y Resolución de Problemas. Quito, Ecuador.
- Instituto Nacional de Estadística y Censos (INEC) Ecuador (2020). Informe sobre

competencias digitales en la educación secundaria. Quito, Ecuador. • Code.org (versión en español). https://code.org • Scratch (MIT). https://scratch.mit.edu • Programamos. https://programamos.com

Módulo Genérico Nro.3	
Nombre del módulo:	Ética, Legislación y Ciudadanía Digital
Nivel:	1ro
Duración:	80 periodos pedagógicos
Unidad de competencia asociada:	**UC3**: Integrar principios éticos y legales en el uso de tecnologías, protección de datos, propiedad intelectual y responsabilidad digital con el fin de asumir desempeño consciente y responsable en entornos digitales.
Objetivo del módulo: Fomentar la comprensión y aplicación de principios éticos, normativas legales y prácticas responsables en el uso de las tecnologías digitales, fortaleciendo una ciudadanía digital crítica, respetuosa, segura y consciente de los derechos y deberes en entorno virtuales.	
Resultados de aprendizaje (RA) y Criterios de Evaluación (CE)	
RA.1. Analizar los principios éticos relacionados con el uso responsable de las tecnologías digitales. **CE1.1:** Analiza casos o situaciones que evidencias conflictos éticos en el uso de tecnologías digitales. **CE1.2:** Relaciona el uso de herramientas digitales con posibles impactos éticos y sociales en distintos contextos. **CE1.3:** Muestra respeto a los derechos de propiedad intelectual en entornos digitales. **CE1.4:** Reflexiona críticamente sobre sus prácticas digitales cotidianas en el uso ético y responsable de la tecnología en actividades personales, académicas, culturales o profesionales. **RA.2. Diferenciar el uso lícito e ilícito de recursos digitales, software y contenidos en línea.** **CE2.1:** Clasifica recursos digitales según su tipo de licencia y condiciones de uso. **CE2.2:** Discrimina prácticas ilícitas como el uso de software pirata, descargas ilegales, reproducción sin autorización o plagio digital. **CE2.3:** Aplica criterios de legalidad y responsabilidad en la selección, uso y difusión de recursos digitales en contextos educativos o institucionales. **CE2.4:** Evalúa las consecuencias legales y éticas del uso indebido de software y contenidos protegidos por derechos de autor. **RA.3. Aplicar normas legales relacionadas con la protección de datos personales y privacidad digital.**	

CE3.1: Distingue el tipo de información que se considera dato personal y sensible según la normativa vigente.

CE3.2: Analiza situaciones de vulneración de privacidad en entornos digitales y sus implicaciones legales y éticas.

CE3.3: Aplica buenas prácticas en la recolección, almacenamiento y uso responsable de datos personales en contextos laborales, académicos o comunitarios.

CE3.4: Verifica que las plataformas, formularios o recursos digitales utilizados respeten principios de confidencialidad, seguridad y protección de datos.

RA.4. Promover comportamientos responsables y seguros en el uso de tecnologías y redes digitales.

CE4.1: Distingue situaciones de riesgo digital derivadas de vulnerabilidades de hardware y software, y propone formas de prevención.

CE4.2: Emplea normas básicas de conducta digital (netiqueta) promoviendo valores como el respeto, la responsabilidad, la empatía y la honestidad en entornos laborales, educativos y sociales, haciendo un uso ético y consciente de las tecnologías.

CE4.3: Participa de manera crítica, ética y responsable en entornos digitales, promoviendo el uso de la tecnología y contribuyendo a una cultura digital inclusiva y respetuosa.

CE4.4: Propone actividades que promuevan la ciudadanía digital, el respeto a los demás y el cuidado del entono virtual.

Contenidos

Conceptuales	Procedimentales	Actitudinales
Ética y moral en entornos digitales: • Definición de ética y moral. • Diferencias entre ética personal, profesional y digital • Dilemas éticos en el entorno tecnológico **Principios éticos aplicados a la tecnología:** • Responsabilidad • Integridad • Justicia y equidad • Respeto a la privacidad • Transparencia **Legislación Digital:** • Marco legal • Derechos y deberes • Delitos cibernéticos • Importancia **Ciudadanía Digital:** • Definición	Analizar casos reales o simulados de dilemas éticos digitales. Evaluar el cumplimiento de principios éticos en el uso de plataformas tecnológicas. Aplicar criterios éticos en la toma de decisiones digitales (uso de información, comunicación, privacidad). Elaborar códigos de conducta o guías de comportamiento digital para contextos educativos o laborales. Participar en debates o foros sobre temas éticos	Mantener compromiso con el uso responsable de las tecnologías. Respetar la privacidad y la identidad digital propia y ajena. Promover el uso ético de las TIC evitando acoso digital, difusión de contenido falso, piratería. Manifestar actitud crítica y reflexiva frente a la información y a la conducta digital. Demostrar empatía y responsabilidad en la interacción virtual.

• Componentes • Habilidades • Participación • Ética • Seguridad **Diferencias culturales y legales** **Tendencias futuras en ética digital** Ejemplos de ética digital • Privacidad de datos • Bias algorítmico • Ciberseguridad • Brecha digital **Uso responsable de las tecnologías digitales** • Comportamiento ético en redes sociales y plataformas digitales. • Derechos y deberes de los usuarios digitales • Netiqueta y ciudadanía digital **Consecuencias del uso no ético de la tecnología**: • Ciberacoso, suplantación de identidad, desinformación. • Impactos sociales, legales y personales del mal uso de las TIC Ética profesional en el ámbito tecnológico Sociedad digital Datos y seguridad	relacionados con la tecnología. Participar en campañas sociales digitales o en espacios de discusión respetuosa. Configurar la privacidad en redes sociales. Realizar prácticas de recolección de datos no éticas. Usar contraseñas seguras, reconocer correos fraudulentos o evitar el acceso a sitios web peligrosos	Mantener disposición a actuar con integridad y honestidad en entornos digitales.

Perfil del o la docente
• Experiencia en el área técnica, poseer título de tercer o cuarto nivel, registrados y reconocidos por el órgano rector del Sistema de Educación Superior en: Tecnologías de la Información, Ciencias de la Computación, Ingeniería en Sistemas, Ingeniería en Software, Tecnologías de la Información y Comunicación, o ramas afines • Experiencia en el campo amplio de la Educación, debidamente certificada.

Orientaciones Metodológicas

- Aprendizaje Basado en Proyectos (ABP)
- Aprendizaje en Contextos Reales
- Role-Playing y Simulaciones
- Metodología STEAM (Integración de Ciencia, Tecnología, Arte y Matemáticas)

Requisitos básicos de infraestructuras, espacio y equipamiento:

Denominación	Detalle de especificaciones técnicas	Cantidad
Infraestructura/espacio	Entorno de aprendizaje Laboratorio informático	1
Laboratorio	Computadoras con acceso a internet Proyector	1

Referencias Bibliográficas

Normativas y Regulaciones

- Ley Orgánica de Protección de Datos Personales (LOPDP)
- Código Orgánico Integral Penal (COIP), sección delitos informáticos
- Ley de Comercio Electrónico, Firmas y Mensajes de Datos
- Reglamentos emitidos por la Superintendencia de Protección de Datos Recursos Digitales y Plataformas

Marcos de referencia y estrategias:

- Estrategia Nacional de Ciberseguridad del Ecuador (última versión disponible)
- OWASP Top 10 (riesgos en aplicaciones web)
- MITRE ATT&CK (catálogo de técnicas y tácticas de ataque)

Sitios Web:

- Plataforma para licencias abiertas y uso responsable de contenido digital. https://creativecommons.org/
- UNESCO: Alfabetización en privacidad https://unesdoc.unesco.org/ark:/48223/pf0000377064
- Organización Mundial de la Propiedad Intelectual (OMPI)
- Legislación internacional, derechos de autor y propiedad intelectual.
- https://www.wipo.int/portal/es/
- Instituto Ecuatoriano de la Propiedad Intelectual (SENADI)
- En caso de estar en Ecuador.
- https://www.propiedadintelectual.gob.ec/
- Common Sense Education
- Recursos para educadores sobre ciudadanía digital, ética, redes y privacidad.
- https://www.commonsense.org/education
- INTEF (España)
- Competencia Digital Docente – incluye módulos sobre ética digital.
- https://intef.es/
- UNESCO – Ciudadanía digital

• Informes y guías para formar en el uso responsable de la tecnología. • https://unesdoc.unesco.org/ark:/48223/pf0000377064

4. Módulos de especialización

Las y los estudiantes del Bachillerato Técnico en Desarrollo de Software se caracterizarán por contar con una sólida formación en el análisis, diseño, programación, implementación y mantenimiento de aplicaciones y sistemas informáticos, garantizando el desarrollo de soluciones tecnológicas eficientes y de calidad.

Se estructuran los siguientes módulos de especialización:

- Programación Estructurada.
- Programación Orientada a Objetos.
- Base de Datos.
- Aplicaciones de Escritorio.
- Aplicaciones Web y Móviles.

Módulo de Especialización Nro. 1	
Nombre del módulo:	Programación Estructurada
Nivel:	1ro
Duración:	240 periodos pedagógicos
Unidad de competencia asociada:	**UC1**: Aplicar programación estructurada en la resolución de problemas computacionales, mediante el análisis de requerimientos, el diseño de algoritmos y flujogramas, uso de lenguajes estructurados en la construcción de soluciones funcionales, empleando buenas prácticas de codificación y lógica computacional.
Objetivo del módulo: Aplicar programación estructurada en la resolución de problemas computacionales, mediante el análisis de requerimientos, diseño de algoritmos, codificación en lenguajes estructurados y aplicación de buenas prácticas de lógica y codificación.	
Resultados de aprendizaje (RA) y Criterios de Evaluación (CE)	
RA.1 Formular problemas computacionales mediante el análisis de requerimientos, identificando entradas, procesos y salidas, a partir de situaciones contextualizadas y especificaciones funcionales básicas. **CE1.1:** Reconoce las necesidades del usuario mediante la interpretación de descripciones orales o escritas del problema, en contextos definidos. **CE1.2:** Identifica los elementos clave del problema utilizando tablas de entrada/salida o diagramas simples, según los datos proporcionados. **CE1.3:** Determina las entradas, procesos y salidas necesarias que permiten plantear soluciones funcionales, considerando las restricciones del problema.	

CE1.4: Define el propósito de la solución esperada a partir del análisis del contexto y los objetivos planteados.

RA.2 Diseñar soluciones algorítmicas utilizando flujogramas y pseudocódigos, aplicando análisis estructurado y secuencia lógica de instrucciones, en función de los requerimientos establecidos.

CE2.1: Elabora flujogramas con simbología técnica y secuencia lógica, de acuerdo con la estructura del algoritmo.

CE2.2: Convierte flujogramas en pseudocódigos estructurados utilizando un lenguaje técnico y estructuras de control apropiadas.

CE2.3: Contrasta la validez de flujogramas y pseudocódigos ante diferentes condiciones de entrada, según los escenarios planteados.

CE2.4: Evalúa la coherencia entre flujograma y pseudocódigo, verificando la secuencia y correspondencia lógica de instrucciones.

RA.3 Codificar algoritmos utilizando un lenguaje de programación estructurado, garantizando la funcionalidad, claridad y mantenibilidad del código.

CE3.1: Construye programas aplicando la sintaxis y semántica del lenguaje de programación estructurado seleccionado.

CE3.2: Integra estructuras condicionales y repetitivas para modelar correctamente el flujo lógico del algoritmo.

CE3.3: Verifica el comportamiento funcional del código mediante pruebas que simulen diferentes entradas y condiciones.

CE3.4: Justifica la elección de estructuras y operadores utilizados en función de su pertinencia técnica dentro del algoritmo.

RA.4 Validar el comportamiento funcional de programas estructurados mediante la ejecución de pruebas, identificación y corrección de errores, optimizando su rendimiento bajo condiciones definidas.

CE4.1 Ejecuta programas con datos de prueba representativos verificando su funcionalidad conforme a los requerimientos.

CE4.2: Analiza errores sintácticos, semánticos y lógicos aplicando técnicas de depuración manual o automatizada.

CE4.3: Corrige el código fuente ajustando fallas e inconsistencias de acuerdo con el comportamiento esperado.

CE4.4: Optimiza el rendimiento del programa implementando mejoras que faciliten su ejecución y comprensión.

RA.5 Documentar programas estructurados mediante el uso de comentarios explicativos, convenciones de codificación y organización lógica, para facilitar la comprensión, mantenibilidad y revisión del código bajo criterios técnicos establecidos.

CE5.1: Examina puntos clave del flujo del programa que requieren explicación mediante comentarios técnicos.

CE5.2: Inserta comentarios descriptivos en líneas y bloques de código explicando estructuras y procesos implementados.

CE5.3: Aplica convenciones de estilo organizando el código con indentación, nomenclatura significativa y secciones lógicas. **CE5.4:** Evalúa la claridad de la documentación del código proponiendo mejoras conforme a estándares técnicos establecidos.		
Contenidos		
Conceptuales	**Procedimentales**	**Actitudinales**
Fundamentos de los lenguajes de programación y sistemas. Definición y propósito de los lenguajes de programación. Clasificación de lenguajes: Lenguaje máquina, ensamblador (bajo nivel), lenguajes de alto nivel. Intérpretes, compiladores y traductores. Lenguaje estructurado: sintaxis, semántica, estructura básica y funciones del editor. Ciclo de vida de una aplicación informática: análisis, diseño, codificación, prueba, mantenimiento y documentación. Tipos de documentación: técnica, de usuario, interna (comentarios), externa. Análisis de problemas informáticos Naturaleza del problema informático: definición, tipos (concretos, abstractos), componentes.	Interpretar requerimientos funcionales a partir de enunciados técnicos o situaciones problemáticas del entorno. Identificar entradas, procesos y salidas a partir del análisis de un problema informático. Descomponer un problema en subproblemas aplicando estrategias de análisis lógico. Establecer relaciones causa-efecto entre los datos de entrada y los resultados esperados del sistema. Clasificar los tipos de datos requeridos según su estructura y tratamiento en el sistema. Representar procesos lógicos utilizando símbolos normalizados en diagramas de flujo. Elaborar pseudocódigos estructurados empleando lenguaje técnico y sintaxis adecuada.	Demostrar ética, responsabilidad y disciplina en la formulación, diseño y codificación de soluciones algorítmicas. Asumir con compromiso el análisis de requerimientos, respetando el rol que cumplen las entradas, procesos y salidas en una solución computacional. Valorar el trabajo bien hecho, cuidando la precisión en la elaboración de flujogramas y pseudocódigos como parte del proceso de resolución de problemas. Mantener interés y disposición para aprender nuevas formas de representar instrucciones mediante lenguajes estructurados. Trabajar con perseverancia y autonomía en la depuración y validación de programas, enfrentando errores como oportunidades de mejora. Participar activamente en la mejora continua del código, aplicando buenas prácticas y estándares técnicos con actitud profesional.

Requerimientos funcionales y no funcionales: interpretación técnica desde el contexto. Elementos Entrada / Proceso / Salida (E/P/S): identificación, clasificación y representación. Abstracción en programación: simplificación lógica del problema. Técnicas de análisis: esquemas funcionales, tablas E/S, representaciones previas al diseño algorítmico. Análisis estructurado: descomposición funcional del problema para su solución paso a paso. Algoritmos y estructuras de control Algoritmo: definición, características (finitud, claridad, efectividad, orden lógico). Clasificación de algoritmos: cualitativos vs cuantitativos. Diseño de algoritmos mediante: Diagramas de flujo (flujogramas): símbolos normalizados (inicio, proceso, decisión, entrada/salida, conectores).	Aplicar estructuras secuenciales, condicionales y repetitivas en pseudocódigos y diagramas. Utilizar técnicas de modularización para organizar procesos en funciones y subrutinas. Relacionar las etapas de diseño con el modelo lógico del programa y su estructura jerárquica. Traducir pseudocódigos a un lenguaje estructurado utilizando sintaxis correcta. Codificar programas empleando estructuras de control (secuenciales, selectivas y repetitivas). Utilizar tipos de datos, operadores y expresiones adecuadas en la solución de problemas. Implementar módulos, funciones y procedimientos para estructurar el código. Aplicar buenas prácticas de codificación (indentación, nomenclatura, reutilización de código). Diseñar y ejecutar casos de prueba para verificar el comportamiento del programa.	Reconocer el valor del pensamiento lógico y secuencial en la construcción de algoritmos, fomentando una actitud reflexiva ante los problemas. Valorar el uso adecuado de las estructuras de control (secuenciales, selectivas y repetitivas) como herramientas fundamentales en la solución de problemas. Mostrar interés por comprender el entorno de desarrollo del lenguaje de programación y por utilizarlo de forma efectiva. Respetar las normas de seguridad digital, propiedad intelectual y uso ético de los recursos informáticos durante todo el ciclo de desarrollo.

Pseudocódigo estructurado: reglas, sintaxis y convenciones. Estructuras de control: Secuencial: instrucciones lineales. Condicional: simple (if), doble (if-else), anidada, múltiple (switch/case). Repetitiva: while, do-while, for; tipos de repetición (manual vs automática). Operadores en programación: aritméticos, relacionales y lógicos. Jerarquía de operadores. Funciones matemáticas básicas: uso, fórmulas y prioridad operativa. Variables, constantes y tipos de datos Elementos básicos de un programa: definición y rol de los datos. Variables y constantes: definición, declaración, ámbito, convenciones de nomenclatura. Tipos de datos primitivos: entero, real, carácter, cadena, booleano. Tipos de datos estructurados: arreglos (vectores y matrices), estructuras, registros.	Comprobar resultados esperados comparándolos con salidas reales del programa. Aplicar técnicas de depuración para identificar errores de lógica, sintaxis o ejecución. Ajustar el código para mejorar la funcionalidad, eficiencia y legibilidad del programa. Incluir comentarios descriptivos en el código que expliquen su funcionamiento. Aplicar convenciones de nomenclatura para variables, funciones y estructuras. Elaborar documentación básica del programa (resumen, propósito, entradas, salidas, versión). Utilizar formatos estándar para presentar la lógica de desarrollo y la funcionalidad del código. Sistematizar la documentación técnica como apoyo a la comprensión del código por terceros.	

Tipos de datos dinámicos: uso de punteros, pilas, colas, listas. Operaciones con punteros: creación y liberación de memoria dinámica (malloc, free). Representación y acceso a datos: lectura por filas/columnas, codificación ASCII. Modularidad y funciones Módulos y funciones: definición, ventajas y organización del código en bloques reutilizables. Funciones y procedimientos: diferencias, estructura, invocación. Paso de parámetros: Por valor, Por referencia. Recursividad: definición, funcionamiento, casos de aplicación. Arreglos y estructuras complejas Arreglos unidimensionales (vectores): definición, declaración, recorrido. Arreglos bidimensionales (matrices): representación por filas y columnas. Estructuras repetitivas anidadas: uso en el manejo		

de arreglos y problemas tabulares. Tablas de decisión: representación lógica de condiciones múltiples. Métodos de ordenamiento: Burbuja, inserción, selección, Shell. Métodos de búsqueda: Secuencial, binaria, transformación de claves, manejo de colisiones, intercalación. Entrada, salida y estructura general del programa Entrada y salida de datos por consola: lectura de datos y presentación de resultados. Estructura general de un programa: declaración de variables, funciones, instrucciones principales. Reglas de estilo y organización del código: indentación, comentarios, bloques estructurados. Uso de editores de texto y herramientas de desarrollo (IDE, compilador, consola). Validación, pruebas y depuración Tipos de errores: Sintácticos: errores de		

forma, identificados por el compilador. Semánticos: instrucciones válidas, pero con significado incorrecto. Lógicos: código correcto que no cumple el propósito. Ciclo de validación: diseño de casos de prueba, ejecución, análisis de resultados. Métodos de prueba: Caja negra: sin conocimiento del código. Caja blanca: con análisis interno del flujo lógico. Herramientas básicas de depuración: impresión de valores, seguimiento paso a paso, uso de puntos de interrupción. Optimización básica del código: eliminación de redundancias, mejora de condiciones, eficiencia operativa.		

Perfil del o la docente

- Experiencia en el área técnica, poseer título de tercer o cuarto nivel, registrados y reconocidos por el órgano rector del Sistema de Educación Superior en: Tecnologías de la Información, Ciencias de la Computación, Ingeniería en Sistemas, Ingeniería en Software, Tecnologías de la Información y Comunicación, o ramas afines
- Experiencia en el campo amplio de la Educación, debidamente certificada.

Orientaciones Metodológicas

- Aprendizaje Basado en Proyectos (ABP)
- Aprendizaje en Contextos Reales
- Role-Playing y Simulaciones
- Metodología STEAM (Integración de Ciencia, Tecnología, Arte y Matemáticas)

Requisitos básicos de infraestructuras, espacio y equipamiento:

Denominación	Especificaciones técnicas	Cantidad
Infraestructura/espacio	Entorno de aprendizaje (aula)	1
Infraestructura/espacio	Aula taller	1
Laboratorio	Computadoras con acceso a internet Servidor Proyector Red de computadoras	1
Herramientas/equipos	Proyector o pizarra digital. Acceso a internet y red local. USB o almacenamiento en la nube.	
Recursos	Lenguajes: C, C++, Python (modo estructurado), Java básico. IDE/Entornos: Visual Studio Code, Thonny, Geany, Code::Blocks, Dev-C++. Simuladores / Diagramadores: Draw.io, Lucidchart, Pencil Project. Herramientas de depuración: Debugger nativo de IDEs, GDB. Documentación: Manuales de lenguaje estructurado, diagramación de flujo y pseudocódigo. Guías de ejercicios con estructuras condicionales, ciclos y funciones. Videos explicativos de algoritmos y estructuras básicas (YouTube, Khan Academy). Repositorios de código ejemplo (GitHub Educativo). Plataformas: W3Schools, Programiz, Sololearn.	

Referencias Bibliográficas

Libros:

- Sznajdleder, P. A. (2020). Programación estructurada a fondo: Implementación de algoritmos en C. Alfaomega. Edición reciente revisada.
- Battaglia, N. (2022). Programación estructurada (programa de asignatura PDF). Universidad Argentina (documento curricular de 2022).
- Márquez, J. (2019). Introducción a la programación estructurada en C. Pearson Educación. Edición actualizada en español.
- Martin, R. C. (2019). Código limpio: guía práctica para escribir código profesional. Anaya Multimedia (ensayos traducidos en español).
- Thomas, D., & Hunt, A. (2022). El programador pragmático: Viaje a la maestría. Anaya Multimedia (segunda edición)

Módulo de Especialización Nro. 2	
Nombre del módulo:	Base de Datos
Nivel:	1ro, 2do, 3ro
Duración:	360 periodos pedagógicos
Unidad de competencia asociada:	**UC3:** Gestionar bases de datos relacionales y no relacionales enlazadas al desarrollo de aplicaciones informáticas, mediante el modelado de datos, diseño conceptual, lógico y físico, implementación de estructuras relacionales y operaciones CRUD, garantizando la integridad, consistencia y disponibilidad de la información, conforme a estándares técnicos y buenas prácticas de administración de datos.
Objetivo del módulo: Gestionar bases de datos, mediante el modelado conceptual, diseño lógico y físico, y la implementación de operaciones CRUD sobre estructuras relacionales, para asegurar la integridad, consistencia y disponibilidad de la información.	
Resultados de aprendizaje (RA) y Criterios de Evaluación (CE)	
RA.1 Interpretar requerimientos funcionales mediante el análisis de entidades, atributos y relaciones del modelo de datos, utilizando simbología estándar y contrastando su completitud con las necesidades de la aplicación. **CE1.1:** Diferencia entidades, atributos y relaciones en función de especificaciones funcionales de una aplicación. **CE1.2:** Representa modelos entidad-relación empleando notación estándar conforme a los requerimientos técnicos. **CE1.3:** Determina claves primarias y foráneas considerando criterios de unicidad y coherencia entre entidades. **CE1.4:** Contrasta el modelo de datos frente a los requerimientos para verificar su completitud estructural y lógica. **RA.2 Estructurar esquemas relacionales a través de la aplicación de reglas de normalización y vínculos referenciales, eliminando redundancias y garantizando la integridad lógica del diseño.** **CE2.1:** Define tablas, campos, tipos de datos y restricciones conforme al modelo relacional. **CE2.2:** Aplica las formas normales (1FN, 2FN, 3FN) en el diseño de la base de datos eliminando duplicidades e inconsistencias. **CE2.3:** Establece relaciones entre tablas utilizando claves primarias y foráneas según el modelo lógico definido. **CE2.4:** Evalúa la integridad referencial mediante pruebas de inserción, consulta y análisis de dependencias.	

RA.3 Desarrollar bases de datos funcionales, aplicando operaciones CRUD con control de integridad y ejecución de consultas estructuradas.

CE3.1: Crea bases de datos, tablas y restricciones utilizando sentencias DDL en un sistema gestor de bases de datos.

CE3.2: Ejecuta operaciones de inserción, actualización y eliminación respetando las relaciones y reglas establecidas.

CE3.3: Formula consultas SELECT integrando filtros, funciones, ordenamientos y agrupaciones según los requerimientos.

CE3.4: Verifica la integridad de la información validando la ejecución correcta de las operaciones CRUD.

RA.4 Comprobar la estabilidad y consistencia de la base de datos mediante pruebas de actualización, recuperación, copias de seguridad y documentación de incidencias.

CE4.1 Revisa la consistencia de los datos ante operaciones de modificación y eliminación.

CE4.2: Ejecuta consultas de recuperación validando exactitud y pertinencia de los resultados.

CE4.3: Realiza copias de respaldo y restauración aplicando las herramientas del sistema gestor.

CE4.4: Registra observaciones técnicas sobre incidencias, ajustes o mejoras durante el uso de la base de datos.

Contenidos

Conceptuales	**Procedimentales**	**Actitudinales**
Fundamentos de bases de datos Definición y propósito de una base de datos en sistemas informáticos. Ventajas y desventajas del uso de bases de datos frente a almacenamiento tradicional. Roles: usuario final, desarrollador, administrador y gestor de base de datos (DBA). Arquitectura general de un sistema gestor de bases de datos (SGBD).	Identificar entidades, atributos y relaciones a partir del análisis de requerimientos. Clasificar atributos según su tipo (simples, compuestos, derivados, multivaluados). Definir claves primarias y relaciones entre entidades con base en reglas del negocio. Aplicar la simbología estándar para representar diagramas entidad-relación. Utilizar herramientas gráficas o CASE para modelar la estructura conceptual.	Asumir responsabilidad por el análisis, modelado y diseño de estructuras de datos conforme a criterios técnicos y éticos. Reconocer la importancia del uso de metodologías sistemáticas para el análisis, diseño y mantenimiento de bases de datos. Trabajar con autonomía, disciplina y sentido de eficiencia en la ejecución de tareas relacionadas con el modelado y administración de datos.

Tipos de modelos de datos: jerárquico, de red, relacional, orientado a objetos. Modelo entidad-relación (E/R): conceptos de entidad, atributo, dominio, identificadores. Tipos de atributos: simples, compuestos, multivaluados, derivados. Relaciones entre entidades: cardinalidad (1:1, 1:N, N:M), opcionalidad y participación. Notación gráfica del modelo E/R (Chen, MinChen o UML simplificado). Diseño lógico y normalización de datos (EC2) Principios del diseño lógico de bases de datos. Transformación del modelo E/R a modelo relacional. Concepto y estructura de una relación: tablas, filas (tuplas) y columnas (atributos). Claves en relaciones: clave primaria, clave ajena, claves alternativas y candidatas. Reglas de integridad: entidad, referencial y de dominio.	Validar el modelo conceptual conforme a criterios de completitud, consistencia y unicidad. Transformar el modelo entidad-relación en un modelo lógico relacional. Determinar claves primarias y foráneas para establecer relaciones e integridad referencial. Aplicar procesos de normalización (1FN, 2FN, 3FN) para optimizar el diseño de tablas. Representar el esquema lógico de la base de datos con herramientas de diseño relacional. Utilizar un sistema gestor de bases de datos (SGBD) para crear el modelo lógico. Verificar el cumplimiento de las dependencias funcionales en las estructuras diseñadas. Crear tablas en lenguaje SQL, definiendo tipos de datos y restricciones de integridad. Establecer relaciones entre tablas mediante claves foráneas. Insertar registros de prueba para comprobar la estructura y relaciones.	Adoptar una actitud crítica y propositiva frente a los problemas técnicos, aportando soluciones creativas e innovadoras en el diseño e implementación de bases de datos. Demostrar seguridad y gusto por el trabajo bien realizado en la estructuración, depuración y optimización de datos. Mantener una actitud de mejora continua mediante el autoaprendizaje y la actualización permanente sobre herramientas y estándares del sector. Valorar la integridad, consistencia y eficiencia de las estructuras relacionales como pilares fundamentales del diseño de sistemas informáticos. Demostrar flexibilización ante cambios de requerimientos, herramientas o entornos tecnológicos, adaptando los modelos y procesos sin perder calidad ni coherencia técnica. Colaborar activamente en equipos de trabajo, respetando roles,

Introducción a la dependencia funcional. Proceso de normalización: 1FN, 2FN, 3FN y 4FN. Impacto de la normalización en la reducción de redundancia y mejora de integridad. Diseño físico e implementación (EC3) Distinción entre diseño lógico y diseño físico. Tipos de datos en SQL: numéricos, cadenas, fechas y booleanos. Sintaxis SQL para: Creación de tablas (CREATE TABLE) Modificación de tablas (ALTER TABLE) Eliminación de tablas (DROP TABLE) Definición de claves primarias y ajenas en la estructura SQL. Creación de vistas (CREATE VIEW) y eliminación (DROP VIEW). Inserción, actualización y eliminación de datos (INSERT, UPDATE, DELETE).	Utilizar herramientas CASE para generar el código de creación de base de datos. Documentar la estructura física de la base de datos implementada. Escribir y ejecutar sentencias SQL para operaciones de inserción, consulta, actualización y eliminación. Utilizar filtros, operadores lógicos, funciones agregadas y cláusulas de agrupación en consultas. Ejecutar consultas multitabla utilizando combinaciones (JOIN). Automatizar tareas de consulta y reporte mediante vistas y subconsultas. Realizar copias de seguridad y restauración de datos utilizando funciones del SGBD. Documentar procedimientos de manipulación y mantenimiento de datos. Instalar y configurar una base de datos no relacional. Realizar de operaciones CRUD (crear, leer, actualizar, eliminar) en bases de datos no relacionales. Integrar bases de datos no relacionales con aplicaciones de escritorio o web.	aportando en la toma de decisiones y fomentando una comunicación clara y profesional. Promover el aprendizaje continuo sobre nuevas tecnologías y modelos de bases de datos. Mantener ética profesional en el manejo y protección de la información almacenada en una base de datos. Incentivar la responsabilidad en la documentación y mantenimiento de las bases de datos.

Operaciones de control condicional y estructurado en el SGBD (triggers básicos o pseudocódigo si no se aborda PL/SQL). Herramientas CASE para modelado: tipos, ejemplos y funcionalidades (UML, DB Designer, Draw.io). Consultas, pruebas y mantenimiento (EC4) Consultas SQL básicas (SELECT, WHERE, ORDER BY, GROUP BY, HAVING). Funciones de agregación: COUNT(), SUM(), AVG(), MAX(), MIN(). Consultas multitabla (JOINs): INNER JOIN, LEFT JOIN, RIGHT JOIN. Subconsultas y alias de tablas y columnas. Creación de respaldos de bases de datos (backups) y restauración básica. Validación de integridad de datos: reglas de validación, pruebas de inserción y consistencia. Métricas básicas de desempeño (tiempo de respuesta, volumen de datos, índices).	Respaldar y restaurar base de datos.	

Definición y características de bases de datos no relacionales (NoSQL). Tipos principales de bases de datos no relacionales: documentos, clave-valor, columnares y grafos. Comparación entre bases de datos relacionales y no relacionales. Modelos de datos y estructuras utilizadas en bases de datos no relacionales. Casos de uso y aplicaciones típicas de bases de datos NoSQL. Consistencia, disponibilidad y particionamiento (modelo CAP).		

Perfil del o la docente

- Experiencia en el área técnica, poseer título de tercer o cuarto nivel, registrados y reconocidos por el órgano rector del Sistema de Educación Superior en: Tecnologías de la Información, Ciencias de la Computación, Ingeniería en Sistemas, Ingeniería en Software, Tecnologías de la Información y Comunicación, o ramas afines
- Experiencia en el campo amplio de la Educación, debidamente certificada.

Orientaciones Metodológicas

- Aprendizaje Basado en Proyectos (ABP)
- Aprendizaje en Contextos Reales
- Role-Playing y Simulaciones
- Metodología STEAM (Integración de Ciencia, Tecnología, Arte y Matemáticas)

Requisitos básicos de infraestructuras, espacio y equipamiento:

Denominación	Detalle de especificaciones técnicas	Cantidad
Infraestructura/espacio	Entorno de aprendizaje (aula)	1
Infraestructura/espacio	Aula taller	1
Laboratorio	Computadoras con acceso a internet	1

	Servidor Proyector Red de computadoras, Simuladores de Diseño de redes	
Equipos/herramientas	Conexión estable a internet para uso de clientes y servidores. Servidor local (XAMPP, WAMP, PostgreSQL, SQLite).	
Recursos	SGBD: MySQL, PostgreSQL, SQLite, SQL Server Express. Clientes de consulta: DBeaver, MySQL Workbench, pgAdmin. Herramientas de modelado: DBDesigner, DBDiagram.io, Lucidchart. Lenguajes de consulta: SQL (DDL, DML, DCL), SQLite queries. Conectores: JDBC (Java), ADO.NET (C#), sqlite3 (Python), ORM básicos (SQLAlchemy). Guías de ejercicios CRUD, relaciones, consultas JOIN. Manuales de normalización, integridad y relaciones entre tablas. Documentación oficial de SGBD utilizados. Cursos: Oracle Academy, Microsoft Learn, Khan Academy (SQL). Recursos legales: normativa ecuatoriana sobre protección de datos (Ley Orgánica de Protección de Datos Personales).	

Referencias Bibliográficas

Libros:

- Silberschatz, A., Korth, H., & Sudarshan, S. (2019). Database System Concepts (7ª ed.). McGraw Hill. (Disponible en trad. y formatos web interactivos).
- Silberschatz, A. (2021). Fundamentos de bases de datos. McGraw Hill (versión en PDF libre difusión educativa).
- Amarú Galvis, E., & Bustamante Martínez, A. B. (2023). Bases de datos relacionales a tu alcance. (Edición económica y breve)
- Christopher, C. J. et al. (2021). SchemaDB: Structures in Relational Datasets. ArXiv preprint (paper reciente sobre estructuras realistas en bases de datos).

Módulo especialización Nro. 3	
Nombre del módulo:	Aplicaciones de Escritorio
Nivel:	1ro, 2do y 3ro
Duración:	360 periodos pedagógicos
Unidad de competencia asociada:	**UC4:** Implementar aplicaciones de escritorio gestionando el ciclo de vida del software, desde el análisis de requerimientos, diseño, codificación, pruebas, mantenimiento y documentación, diseñando interfaces gráficas funcionales, integrando bases de datos y empleando herramientas colaborativas, control de versiones, así como metodologías de desarrollo asegurando la calidad técnica y usabilidad.
Objetivo del módulo: Implementar aplicaciones de escritorio gestionando el ciclo de vida del software, mediante el análisis de requerimientos, diseño de interfaces gráficas, codificación, pruebas, mantenimiento y documentación, para garantizar la funcionalidad, calidad técnica y usabilidad del producto final.	
Resultados de aprendizaje (RA) y Criterios de Evaluación (CE)	
RA.1 Analizar requerimientos funcionales y no funcionales de una aplicación de escritorio mediante técnicas de recolección de información alineadas a objetivos del usuario y factibilidad técnica. **CE1.1:** Analiza requerimientos a partir de entrevistas, encuestas o documentos técnicos según los objetivos a cumplir para el aplicativo. **CE1.2:** Clasifica requerimientos en funcionales y no funcionales considerando criterios técnicos, de negocio y experiencia de usuario. **CE1.3:** Representa los requerimientos utilizando modelos o herramientas como historias de usuario, casos de uso o diagramas estructurados. **CE1.4:** Contrasta los requerimientos establecidos con las condiciones de factibilidad técnica de implementación del aplicativo. **RA.2 Diseñar la interfaz gráfica de una aplicación de escritorio considerando principios de usabilidad, diseño modular y patrones visuales orientados al usuario final.** **CE2.1:** Organiza pantallas, formularios y componentes gráficos de acuerdo con el flujo lógico funcional de la aplicación empleando prototipos y diseños estandarizados. **CE2.2:** Aplica principios de jerarquía visual, consistencia y accesibilidad en la creación de interfaces gráficas enfocado en la usabilidad y accesibilidad. **CE2.3:** Delimita módulos de la aplicación integrando patrones de diseño adecuados al contexto del proyecto, justificando el empleo de diseños según las necesidades del usuario. **CE2.4:** Justifica decisiones de diseño mediante prototipos, wireframes o mockups estructurados según estándares técnicos. **RA.3 Desarrollar funcionalidades de una aplicación de escritorio integrando interfaz gráfica, lógica de negocio y acceso a bases de datos, utilizando herramientas de desarrollo y/o bibliotecas.**	

CE3.1: Codifica módulos funcionales empleando estructuras de control, formularios interactivos y validaciones lógicas.
CE3.2: Conecta la aplicación a una base de datos utilizando controladores, comandos SQL y objetos de acceso a datos.
CE3.3: Integra componentes gráficos e interacción con el usuario según la lógica del negocio definida en los requerimientos.
CE3.4: Verifica la funcionalidad del sistema comprobando la coherencia del flujo de datos entre la interfaz, la lógica y la base de datos.

RA.4 Evaluar la calidad funcional de la aplicación de escritorio mediante pruebas, ajustes y documentación técnica garantizando estabilidad, mantenimiento y comprensión del sistema.

CE4.1 Ejecuta pruebas unitarias, funcionales e integradas siguiendo casos de prueba establecidos para cada módulo.
CE4.2: Corrige errores lógicos, estructurales o visuales durante la fase de pruebas manteniendo la integridad del sistema.
CE4.3: Optimiza el código mediante refactorización técnica orientada a mejorar rendimiento, legibilidad y escalabilidad.
CE4.4: Elabora documentación técnica y de usuario describiendo instalación, estructura funcional y mantenimiento del sistema.
CE4.5 Publica entregables técnicos utilizando herramientas colaborativas y sistemas de control de versiones con seguimiento estructurado.

RA.5 Aplicar metodologías ágiles en el desarrollo de aplicaciones organizando tareas, gestionando control de versiones e integrando documentación técnica del proceso.

CE5.1: Organiza tareas del proyecto en tableros digitales siguiendo los principios y ciclos establecidos en la metodología ágil seleccionada.
CE5.2: Gestiona versiones del código fuente utilizando un sistema de control que emplea convenciones técnicas, mensajes claros y estandarizados.
CE5.3: Integra control de versiones respetando la asignación de tareas y completando revisiones dentro de ciclos definidos.
CE5.4: Documenta el proceso de desarrollo empleando formatos técnicos estandarizados que aseguran trazabilidad de cambios, decisiones técnicas y criterios de aceptación.

Contenidos

Conceptuales	**Procedimentales**	**Actitudinales**
Características de los requerimientos funcionales y no funcionales. Métodos de recolección de información: entrevistas, encuestas, observación, análisis documental.	Interpretar necesidades del usuario a partir de fuentes diversas. Elaborar especificaciones de requerimientos funcionales y no funcionales.	Valorar la importancia de comprender las necesidades reales del usuario antes de proponer soluciones técnicas. Asumir con responsabilidad la recolección y validación de requerimientos como base para un desarrollo efectivo.

Técnicas de análisis de requerimientos: casos de uso, historias de usuario, diagramas UML. Criterios de validación de requerimientos según estándares de calidad del software. Principios de diseño de interfaces gráficas: consistencia, visibilidad, retroalimentación. Estándares de usabilidad y accesibilidad. Patrones de diseño orientados a la interfaz de usuario (MVC, MVVM). Estructura modular de aplicaciones: componentes, controladores, vistas. Características de lenguajes de programación orientados a escritorio (ej. Java, C#). Librerías gráficas y controladores de eventos. Conexión a bases de datos: drivers, consultas, transacciones. Principios de encapsulamiento, modularidad y reutilización de código. Pruebas funcionales y unitarias.	Representar requerimientos utilizando diagramas y descripciones estructuradas. Contrastar requerimientos con escenarios de uso previstos. Construir wireframes y prototipos navegables. Aplicar patrones de diseño en la organización lógica de la aplicación. Definir estructuras modulares y navegación entre ventanas o componentes. Evaluar el diseño mediante pruebas heurísticas o de usabilidad. Programar eventos, validaciones y flujos de interacción. Integrar operaciones CRUD con bases de datos relacionales. Utilizar entornos de desarrollo y herramientas de depuración. Aplicar buenas prácticas de codificación, manejo de errores y seguridad. Planificar y ejecutar pruebas con casos definidos.	Mostrar disposición al diálogo y la escucha activa al interactuar con usuarios o miembros del equipo técnico. Reconocer con apertura las limitaciones y ajustar las expectativas frente a los requerimientos identificados. Apreciar la estética y funcionalidad como elementos complementarios en el desarrollo de interfaces. Demostrar sensibilidad frente a las necesidades de accesibilidad y diversidad de usuarios. Mantener actitud crítica y autocrítica frente a las decisiones de diseño tomadas en el proceso. Actuar con disciplina y responsabilidad en la codificación siguiendo estándares y buenas prácticas. Mantener perseverancia y organización ante la solución de errores o dificultades técnicas. Colaborar activamente en equipos de desarrollo respetando ideas y roles definidos. Fomentar la ética profesional evitando prácticas que comprometan la seguridad o calidad del software. Asumir con compromiso la revisión crítica del propio

Estrategias de mantenimiento: correctivo, adaptativo, perfectivo. Tipología y estructura de la documentación técnica y de usuario. Indicadores de calidad del software (fiabilidad, eficiencia, mantenibilidad). Metodologías de desarrollo de software (tradicionales, ágiles, hibridas) Herramientas de control de versiones: Git. GitHub/GitLab/Bitbucket.	Analizar resultados y diagnosticar errores o inconsistencias. Documentar funcionalidades, estructura del sistema y manuales de usuario. Implementar mejoras sobre la base de resultados obtenidos. Configura un entorno de desarrollo para aplicaciones de escritorio (IDE, dependencias, gestor de versiones). Aplicar una metodología de desarrollo de software según el proyecto en entregas funcionales. Gestionar repositorios de código mediante Git, siguiendo una estrategia de ramas. Realizar commits con mensajes descriptivos y estandarizados. Colaborar en equipo utilizando pull requests, revisión de código y resolución de conflictos. Asegurar trazabilidad entre requerimientos, tareas, código y entregables documentados.	trabajo para garantizar la calidad del producto. Ser receptivo ante observaciones o sugerencias técnicas que mejoren el rendimiento de la aplicación. Mostrar rigor en la documentación como evidencia de responsabilidad técnica y profesional. Valorar la importancia del mantenimiento y mejora continua del software después de su implementación. Promover la mejora continua del producto y del proceso de trabajo. Asumir con responsabilidad la planificación y cumplimiento de tareas asignadas dentro del equipo.
Perfil del o la docente		
• Experiencia en el área técnica, poseer título de tercer o cuarto nivel, registrados y reconocidos por el órgano rector del Sistema de Educación Superior en: Tecnologías de la Información, Ciencias de la Computación, Ingeniería en Sistemas, Ingeniería en Software, Tecnologías de la Información y Comunicación, o ramas afines		

• Experiencia en el campo amplio de la Educación, debidamente certificada.		
Orientaciones Metodológicas		
• Aprendizaje Basado en Proyectos (ABP) • Aprendizaje en Contextos Reales • Role-Playing y Simulaciones • Metodología STEAM (Integración de Ciencia, Tecnología, Arte y Matemáticas)		
Requisitos básicos de infraestructuras, espacio y equipamiento:		
Denominación	**Especificaciones técnicas**	**Cantidad**
Infraestructura/espacio	Entorno de aprendizaje (aula)	1
Laboratorio	Computadoras con acceso a internet Servidor Proyector Red de computadoras	1
Equipos/herramientas	Acceso a internet y red para pruebas colaborativas. Servidor local (si se requiere acceso a bases de datos).	
Recursos	Lenguajes: Java (Swing, JavaFX), C# (WinForms, WPF), Python (Tkinter, PyQt5). IDE/Entornos: Visual Studio, NetBeans, Eclipse, PyCharm. Intellij Idea, entre otros	
Referencias Bibliográficas		
Libros: • Morales Arteaga, L. D. (2023). Desarrollo de aplicaciones de escritorio (documento resumen educativo). Obtenido de Scribd como guía práctica. • Herrera, L. (2021). Desarrollo de Aplicaciones de Escritorio con PHP. Publicación Kindle en español aplicable a entornos gráficos con acceso local.		

5. Módulo práctico/experimental

Este módulo fortalece la formación del estudiantado mediante su participación en situaciones reales de aprendizaje, simulaciones, acercamiento a entornos de trabajo, giras de observación, articulación con el sector productivo, proyectos interdisciplinarios y metodologías basadas en proyectos. Todas estas actividades tienen como finalidad consolidar los conocimientos teóricos y favorecer la aplicación de competencias en contextos reales. Asimismo, promueven el desarrollo de habilidades blandas, tales como el trabajo en equipo, la comunicación asertiva, la resolución de problemas y la adaptación a entornos cambiantes.

En síntesis, este módulo constituye la aplicación práctica e integral de lo aprendido en los módulos previos, permitiendo al estudiantado experimentar, analizar y resolver situaciones reales, al tiempo que refuerza sus capacidades técnicas y fomenta el

desarrollo de competencias conceptuales, procedimentales y actitudinales, tanto de los módulos genéricos como de los de especialización.

Módulo Práctico Experimental		
Nombre del módulo:	Práctico Experimental	
Nivel:	1ro, 2do, 3ro	
Duración:	280 periodos pedagógicos	
Unidad de competencia asociada:	**UC1:** Aplicar programación estructurada en la resolución de problemas computacionales, mediante el análisis de requerimientos, el diseño de algoritmos y flujogramas, uso de lenguajes estructurados en la construcción de soluciones funcionales, empleando buenas prácticas de codificación y lógica computacional. **UC4:** Implementar aplicaciones de escritorio gestionando el ciclo de vida del software, desde el análisis de requerimientos, diseño, codificación, pruebas, mantenimiento y documentación, diseñando interfaces gráficas funcionales, integrando bases de datos y empleando herramientas colaborativas, control de versiones, así como metodologías de desarrollo asegurando la calidad técnica y usabilidad. **UC5:** Crear aplicaciones para entornos web y móvil utilizando tecnologías y marcos de trabajo actuales, a través del diseño de interfaces adaptables, uso de lenguajes de marcado y de programación, integración de librerías, bases de datos modularizando el desarrollo del frontend y backend.	
Objetivo del módulo: Desarrollar en el estudiantado competencias para analizar requerimientos, diseñar algoritmos y aplicar programación en la creación de soluciones tecnológicas, mediante la codificación en lenguajes estructurados, de objeto o funcionales, el diseño y desarrollo de aplicaciones de consola, escritorio, web con HTML, responsivas y orientadas a dispositivos móviles, que permitan automatizar procesos básicos de las organización, incorporando buenas prácticas de programación y normativa legal según el enfoque de la aplicación.		
Resultados de aprendizaje (RA)	**Criterios de evaluación (CE)**	**Actividades Prácticas Experimentales**
RA1: Diseñar soluciones	**CE1.1:** Elabora flujogramas con	Crea la estructura del programa desarrollando un

algorítmicas utilizando flujogramas y pseudocódigos aplicando análisis estructurado y secuencia lógica de instrucciones.	simbología técnica y secuencias lógicas de acuerdo con la estructura del algoritmo. **CE1.2:** Convierte flujogramas en pseudocódigo estructural utilizando un lenguaje técnico.	algoritmo robusto para el correcto funcionamiento.
RA2: Codificar algoritmos utilizando un lenguaje de programación garantizando la funcionalidad, claridad y mantenibilidad del código.	**CE2.1:** Construye programas aplicando la sintaxis y la semántica del lenguaje de programación. **CE2.2**: Verifica el comportamiento funcional del código mediante pruebas que simule diferentes entradas, procesos y salidas.	Construye programas en un lenguaje de programación aplicando una correcta escritura de instrucciones, garantizando que el código cumpla su propósito funcional, con claridad, precisión lógica y calidad en los resultados.
RA3: Diseñar la interfaz gráfica de usuario para una aplicación de escritorio aplicando principios de usabilidad.	**CE3.1:** Organiza la estructura básica GUI con menús botones y formularios adecuados al flujo de la aplicación. **CE3.2:** Aplica principios de diseño visual para garantizar una experiencia de usuario de adecuada.	Diseña una GUI para una aplicación moderna aplicado al mundo real, utilizando herramientas como java, swing o pyqt, .Net, Delphi, entre otros. Evalúa y mejorar el diseño visual de la GUI creada, aplicando feedback de usuario o rubrica de usabilidad.
RA4: Planificar el desarrollo de una aplicación web, responsiva, orientada a dispositivos móviles, empleando marcos de trabajo y gestión colaborativa en equipo,	**CE4.1:** Identifica los componentes básicos de una aplicación web estructura, diseño funciones	Diseña el prototipo en papel o una herramienta digital la construcción de una aplicación web. Diagrama casos de uso, pruebas y depuración.

aplicada al mundo real.	**CE4.2:** Codifica, aplica diseño y funcionalidad de la aplicación. **CE4.3:** Publica los archivos de la aplicación web, realizando pruebas de usuario y de carga.	Ejecuta pruebas funcionales de la aplicación web.