PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

FORMATO GUÍA DE APRENDIZAJE

1. IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE

Denominación del Programa de Formación: Técnico en programación de Software

Código del Programa de Formación: 233104 - v.2

Nombre del Proyecto Formativo: Desarrollo de Soluciones de Software para la Gestión Empresarial

Fase del Proyecto: Análisis Y Planeación

Actividad de Proyecto Formativo: Interpretar informe técnico de diseño de la solución propuesta del proyecto formativo.

SE DEBEN INTENSIFICAR LAS COMPETENCIAS Y LOS RESULTADOS DE APRENDIZAJE ASOCIADOS EN ESTA FASE

Competencia 1: ESTABLECIMIENTO DE REQUISITOS DE LA SOLUCIÓN DE SOFTWARE

Resultados de Aprendizaje:

- RAP 1: Caracterizar la información a recolectar de acuerdo con las necesidades del cliente.
- RAD 2: Interpretar el informe de requisitos de la solución conforme a la identificación de las necesidades del negocio.

Duración de la Guía de Aprendizaje:

- RAP 1: 48 Horas Directas y 10 Horas Indirectas.
- RAP 2: 67 Horas Directas y 13 Horas Indirectas.

2. PRESENTACIÓN

¡Bienvenido a esta experiencia de aprendizaje! En esta guía, de aprendizaje es una valiosa oportunidad para fortalecer tus habilidades en la **identificación de necesidades y establecimiento de requisitos de soluciones de software**, desarrollando competencias esenciales para tu crecimiento profesional. A lo largo del proceso, te guiarás por un enfoque sistemático y organizado que te permitirá caracterizar e interpretar la información clave de acuerdo con los requerimientos del cliente, integrando conocimientos previos y nuevos aprendizajes. Esta experiencia está pensada para fomentar tu autonomía, motivarte a la acción y promover el análisis crítico, permitiéndote construir soluciones sólidas basadas en la comprensión profunda del

negocio. Además, trabajarás de forma colaborativa, fortaleciendo el trabajo en equipo, la comunicación asertiva y el pensamiento analítico. Así, no solo avanzarás en el dominio técnico de los requisitos de software, sino también en tu desarrollo integral como aprendiz comprometido y competente. ***¡Participa activamente, aprende con propósito y aporta al crecimiento de tu grupo!*** 💻🤝📈

¡Hola a todos!

Para complementar su **Proyecto Productivo** y enriquecerlo con información valiosa, les invitamos a explorar el siguiente enlace. ¡Hagan clic aquí! y descubran recursos interesantes que les serán de gran utilidad!

3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

Competencia 1: ESTABLECIMIENTO DE REQUISITOS DE LA SOLUCIÓN DE SOFTWARE

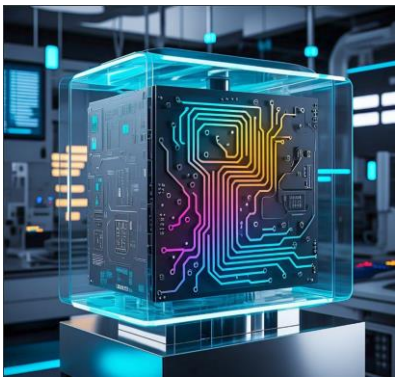
Resultados de Aprendizaje:

- RAP 1: Caracterizar la información a recolectar de acuerdo con las necesidades del cliente.
- RAD 2: Interpretar el informe de requisitos de la solución conforme a la identificación de las necesidades del negocio.

Actividad de Proyecto Formativo 1: Interpretar informe técnico de diseño de la solución propuesta del proyecto formativo.

3.1 Actividades de Reflexión Inicial:

Descripción de la Actividad: Embárcate en una experiencia única con la actividad ***"El Viaje del Requisito: De la Idea al Software"***, donde tu creatividad y conocimiento se unen para construir una caja mágica que represente el proceso completo del desarrollo de software. En esta actividad, explorarás conceptos clave como requisitos, roles, metodologías y herramientas, conectando ideas nuevas con aprendizajes previos. Además, podrás reflexionar críticamente, trabajar en equipo y presentar tu visión de manera dinámica e innovadora. ***¡Una oportunidad perfecta para aprender haciendo, crear y crecer junto a tus compañeros!***



Ambiente requerido: Sala de sistemas.

Estrategias o técnicas didácticas activas: "El Viaje del Requisito: De la Idea al Software"

🗺️ **Instrucciones:**

1. 📦 **Crea una caja mágica del software (física):** Cada pareja de aprendices creará una “caja mágica del software” decorada con elementos que representen la tecnología y la creatividad.
2. 🧩 **Construcción de las tarjetas del viaje:** Dentro de la caja, deberán incluir 10 tarjetas reflexivas, cada una representando una etapa o concepto clave del desarrollo de software (por ejemplo: metodologías, roles, técnicas de recolección, requisitos funcionales/no funcionales, trazabilidad, UML, control de versiones, etc.).

   Cada tarjeta debe incluir:

   - 🎯 Un concepto.
   - 🤔 Una pregunta reflexiva.
   - 💡 Una analogía creativa (ej. "La trazabilidad es como el GPS del proyecto").

3. 🎉 **Galería reflexiva con análisis crítico**

   Instrucciones:

   1. Comparte la caja mágica del software creada por los grupos.
   2. Haz una **galería caminante** : cada grupo visita los trabajos de los demás y anota observaciones en post-its:
      - ¿Qué conceptos faltan?
      - ¿Qué relaciones están poco claras?
      - ¿Qué les sorprendió?
      - ¿Qué conectarán con su experiencia personal o profesional?
   3. De regreso, cada grupo revisa los comentarios recibidos y ajusta su caja mágica del software.

4. 🎬 ***Exposición interactiva:*** En parejas, presentarán su “viaje del requisito” a los compañeros, explicando cómo cada tarjeta representa una parte del proceso. La presentación puede ser con dramatización, mapa visual, juego de roles o infografía animada.

5. 📓 ***Reflexión escrita individual:*** Cada aprendiz redactará una reflexión donde responda:
    - ¿Qué aprendí sobre el proceso de desarrollo del software?
    - ¿Qué parte del recorrido me pareció más desafiante o importante?
    - ¿Cómo puedo aplicar este conocimiento en un proyecto real?
    - ¿Qué diferencias identifican entre metodologías ágiles y tradicionales?
    - ¿Qué importancia tienen los requisitos para el éxito de un proyecto de software?
    - ¿Cómo elegirías las técnicas de recolección de información según el tipo de cliente?
    - ¿Por qué es clave la trazabilidad de requisitos en proyectos grandes?
    - ¿Cuál es el papel de las herramientas de modelado y control de versiones en el análisis?

Materiales de formación: Computadores con conexión a Internet.

Material de apoyo:

Duración de la actividad: 20 horas directas y 3 horas indirectas.

3.2 Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje:

Descripción de la actividad: ¡Prepárate para una aventura de aprendizaje única con la actividad *"Misiones Secretas del Software"*! A través de retos creativos, dinámicos y colaborativos, descubrirás cómo se construyen los grandes sistemas de software desde sus primeras ideas hasta su ejecución final. Esta experiencia te permitirá conocer metodologías, roles, técnicas y herramientas esenciales para el desarrollo de soluciones reales. Es tu oportunidad para aprender haciendo, compartir con tus compañeros y desarrollar habilidades clave para tu futuro profesional. **¡Acepta la misión y transforma tu conocimiento!**

Ambiente requerido: Sala de Sistemas.

Estrategias o técnicas didácticas activas: "*Misiones Secretas del Software*"

🧠 ***Contexto Ficticio:***

Ha sido reclutado por la agencia "SoftMission" para formar parte de un equipo de élite encargado de salvar el proyecto más importante de una empresa tecnológica global. El sistema de software que están desarrollando está en peligro por una mala gestión de requisitos, falta de claridad en los roles, y

la confusión entre metodologías ágiles y tradicionales. Tu misión: identificar las fallas, proponer soluciones y construir el conocimiento necesario para evitar el colapso del sistema.

Dinámica: "Misión en 5 Estaciones"

Organiza al grupo en equipos de 4 personas. Cada estación será una *misión* que representa una categoría clave del proceso de desarrollo de software. Cada grupo rotará por todas las estaciones

1. **Estación 1 – "La Máquina del Tiempo del Desarrollo"**
   - Compara metodologías ágiles vs. tradicionales.
   - Crea una línea del tiempo visual con los conceptos, ventajas y desventajas.
   - Juego: unir tarjetas de metodología con sus características correctas.

2. **Estación 2 – "Escanea el Código Secreto"**
   - Analiza documentos y técnicas de recolección de requisitos.
   - Actividad: relacionar técnicas (focus group, entrevistas, encuestas) con situaciones prácticas simuladas.

3. **Estación 3 – "Agentes del Proyecto"**
   - Identifica y representa con dibujos o personajes los distintos roles del desarrollo (Director de proyecto,Analista de sistemas: Diseñador de software, Desarrollador, Probador, Responsable de calidad,Gestor de configuración, etc.).
   - Juego de roles: simulan una reunión inicial de levantamiento de requisitos.

4. **Estación 4 – "La Caja Negra del Requisito"**
   - Clasifican requisitos funcionales y no funcionales.
   - Actividad: escribir ejemplos reales y colocarlos en la caja correspondiente.
   - Construyen un mini ERS en papel con plantilla simplificada.

5. **Estación 5 – "UML y los Diagramas Perdidos"**
   - Juego de emparejar íconos de UML con su significado.
   - Tarea: crear un mini storyboard o caso de uso del sistema imaginario.

Cierre de la actividad:

Cada grupo comparte sus hallazgos, reflexiona sobre los errores comunes en la gestión de requisitos y propone una lección clave aprendida en cada estación. El instructor complementa con información técnica**.**

🎁 Entregable:

Cada equipo entrega un *"Mapa de Misión Completada"*, que es una infografía o presentación que resume lo aprendido en cada estación con conceptos, ejemplos y analogías.

Materiales de formación: Computador con acceso a internet

Material de apoyo:

Duración de la actividad: 20 horas directas y 5 horas indirectas.

3.3 Actividades de apropiación:

Descripción de la actividad: ***¡Prepárate para sumergirte en una aventura de aprendizaje única con "El Tablero del Software: De la Idea al Código"!*** A través de esta actividad creativa y colaborativa, diseñarán un juego interactivo que simula el recorrido completo del desarrollo de software. Pondrás en práctica conceptos clave como metodologías, requisitos, modelado, trazabilidad, UML y control de versiones. Esta experiencia te permitirá aprender jugando, resolver retos reales y conectar teoría con práctica. ***¡Atrévete a construir, reflexionar y convertirte en un verdadero desarrollador estratégico!***

Ambiente requerido: Sala de sistemas

Estrategias o técnicas didácticas activas: "El Tablero del Software: ¡De la Idea al Código!"

Cada grupo (3 a 4 aprendices) diseñará y construirá un tablero de juego digital (usando herramientas como Genially, Canva, PowerPoint interactivo o Miro), el cual simulará un recorrido completo de un proyecto de software, desde la concepción hasta la entrega final. El tablero debe representar de forma dinámica e interactiva los pasos, desafíos, decisiones y herramientas utilizadas en cada etapa del ciclo de vida del desarrollo de software.

🧩 Contenido mínimo del tablero (por secciones/casillas):

1. **Inicio del Proyecto**

   📌 Idea del software

   📌 Roles: usuarios, stakeholders, analista, dueño del producto, equipo

   📌 Técnicas de recolección de información: entrevistas, encuestas, focus group, observación

2. **Recolección y análisis de requisitos**

   📌 Requisitos funcionales y no funcionales

   📌 Elicitación y análisis de requisitos

   📌 Técnicas de análisis

   📌 Estándar IEEE 830 (definir, ejemplificar)

   📌 Matriz de trazabilidad

3. **Diseño del software**

   📌 Plantillas ERS

   📌 Lenguajes de modelado: conceptos, tipos, ventajas y desventajas

   📌 Herramientas para captura de requisitos: diagramas de casos de uso, historias de usuario, storyboard

   📌 Generar el storyboard con el caso de estudio.

   📌 Modelado UML: elementos, relaciones, diagramas, clasificación

   📌 Herramientas de modelado: Adobe XD, Canvas, Figma

4. **Ciclo de vida del desarrollo del software**

   📌 Fases: análisis, diseño, implementación, pruebas, mantenimiento

   📌 Diagrama del ciclo de vida (en cascada, incremental, iterativo, ágil)

   📌 Comparación entre metodologías ágiles y tradicionales

5. **Gestión y seguimiento**

   📌 Priorización de requisitos

   📌 Control de versiones: concepto, herramientas (Git, GitHub), ventajas

   📌 Presentación de entregables

📌 Retroalimentación y mejora continua

🧨 Retos del tablero:

Cada casilla o sección debe tener una tarjeta de reto, que contenga:

📌 Una pregunta de aplicación o reflexión sobre el tema.

📌 Una prueba de conocimiento (pregunta tipo test o juego interactivo).

📌 Una actividad lúdica o de rol, como dramatizar una reunión Scrum, armar un diagrama UML o elegir la mejor herramienta de modelado.

📦 Producto Final:

📌 Tablero interactivo funcional (digital), con diseño atractivo y navegación clara.

📌 Mapa del recorrido del proyecto con casillas temáticas.

📌 Tarjetas con contenido reflexivo y dinámico.

📌 Presentación a los compañeros del curso.

📌 Reflexión individual: ¿Qué aprendiste?, ¿qué conceptos relacionaste?, ¿cómo lo aplicarías?

Materiales de formación: Computador con conexión a internet, Canva / Genially / PowerPoint interactivo, GitHub (para mostrar control de versiones básico), Miro / Figma para esquemas, Adobe XD para simular interfaces.

Material de apoyo:

- https://view.genially.com/6692b79f697166f0f86da8ca/interactive-content-programacion-de-aplicaciones-y-servicios-para-la-nube-modulo-2
  ***Importancia de la Especificación de Requisitos*- Aporta los fundamentos necesarios para representar correctamente los requisitos y su trazabilidad en el juego.**

- https://view.genially.com/66e434d7a0c4bea4ee972deb
  ***Validación de Requisitos – Criterios de Validación y Pruebas*- Refuerza el enfoque en evaluación y ajuste de requisitos como parte del recorrido del juego.**

- https://view.genially.com/67c5b6161bccb557c1dbc5d2
  ***Despliegue Eficiente de Aplicaciones en la Nube*- Puede integrarse en el tramo final del juego para simular la fase de entrega, pruebas y publicación.**

Evidencias de aprendizaje: Conocimiento: Evaluación de conocimiento de ciclo de la vida del software herramienta para captura de requisitos, roles, requisitos funcionales y no funcionales y las metodologías ágiles y tradicionales.

Desempeño: tablero digital, mapa del recorrido del proyecto, tarjetas con contenidos reflexivo

Producto: Documento de la interpretación de los requerimientos asignados (Formato de legalización de requerimientos)

Instrumentos de evaluación: Cuestionario, Lista de chequeo

Duración de la actividad: 35 horas directas y 7 horas indirectas.

3.4 Actividades de Transferencia el Conocimiento:

Descripción de la actividad: En esta actividad te convertirás en un consultor experto que deberá salvar un proyecto de software que fracasó por errores comunes. Trabajarás con tu equipo para aplicar metodologías, técnicas de recolección de requisitos, diagramas UML, herramientas de modelado y control de versiones, transformando un caos en una solución profesional. **¡Atrévete a tomar decisiones reales, aplicar tus conocimientos y demostrar tu ingenio para llevar este proyecto al éxito! ¿Estás listo para este reto?** 💻🚀

Ambiente requerido: Sala de Sistemas

Estrategias o técnicas didácticas activas: 🧠 **Actividad de Transferencia: "Rescatando el Proyecto SofTech"**

Contexto del caso de estudio:

La empresa ficticia SofTech Solutions ha sido contratada por una clínica veterinaria que necesita un sistema de gestión integral. Sin embargo, el proyecto anterior fue un fracaso por falta de análisis de requisitos, mala comunicación con stakeholders, y una mala elección de metodología. Tu misión como consultor de rescate del proyecto es rediseñar todo el proceso de desarrollo de software utilizando buenas prácticas.

🎯 Desarrollo de la actividad:

1. Formación de equipos (3 a 4 personas).
2. Lectura del caso: Se entrega un documento con los antecedentes del fallo, incluyendo errores en la recolección de información, roles mal definidos y una plantilla ERS mal estructurada.

3. Retos del equipo:

   📌 Elegir la metodología adecuada (ágil o tradicional) justificando por qué.

   📌 Aplicar técnicas de elicitación de requisitos para definir qué necesita realmente la clínica.

   📌 Usar herramientas como diagramas de casos de uso, historias de usuario o storyboard para representar el sistema.

   📌 Elaborar una mini plantilla ERS alineada con el estándar IEEE 830.

   📌 Identificar requisitos funcionales y no funcionales, con su trazabilidad.

   📌 Representar el ciclo de vida del software y crear un diagrama UML básico.

   📌 Describir cómo usarían Adobe XD o Canva para prototipar y GitHub o GitLab para control de versiones.

Producto final:

- Informe de rescate del proyecto.
- Mapa visual del proceso propuesto.
- Presentación creativa del caso (infografía interactiva para socializar con sus compañeros).

Materiales de formación: Computadores con conectividad a Internet.

Material de apoyo:

Programa: Programación de Software

- https://view.genially.com/679839afac4b87277dac1741

  ***Diseño y Desarrollo de Software con Programación Orientada a Objetos*- Aporta estructura al rediseño del sistema, especialmente cuando el proyecto falló por falta de cohesión en la lógica de objetos.**

- https://view.genially.com/66971fb73037eb3b9d3a494c/interactive-content-programacion-de-software-modulo-2

  ***Estándares IEEE: Concepto y Características*- Brinda criterios técnicos para estructurar el proyecto con estándares reconocidos que ayudan a evitar errores comunes en el desarrollo.**

Evidencias de aprendizaje: Desempeño: Mini plantilla ERS alineada con el estándar IEEE 830. mapa visual del proceso propuesto.

Producto: Informe de rescate del proyecto, Exposición de infografía interactiva

Instrumentos de evaluación: Lista chequeo

Duración de la actividad: 40 horas directas y 8 horas indirectas.

¡Felicitaciones, aprendices, por culminar la **Fase Análisis** 🎉 Han demostrado un gran compromiso y habilidades excepcionales en el **ESTABLECIMIENTO DE REQUISITOS DE LA SOLUCIÓN DE SOFTWARE** . Su dedicación y esfuerzo los convierten en profesionales cada vez más capacitados. 🚀 Cada paso que dan fortalece su preparación y los acerca a nuevas oportunidades en el mundo de la tecnología. Sigan creciendo, explorando y perfeccionando sus habilidades, porque el aprendizaje continuo es la clave del éxito.

4. PLANTEAMIENTO DE EVIDENCIAS DE APRENDIZAJE PARA LA EVALUACIÓN EN EL PROCESO FORMATIVO.

Fase del proyecto formativo	Actividad del proyecto formativo	Actividad de Aprendizaje	Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
ANALISI	Interpretar informe técnico de diseño de la solución propuesta del proyecto formativo.	Interpretar los requisitos del software de acuerdo con los requerimientos del cliente.	**Conocimiento:** Evaluación de conocimiento de ciclo de la vida del software herramienta para captura de requisitos, roles, requisitos funcionales y no funcionales y las metodologías ágiles y tradicionales. **Desempeño:** tablero digital,	Determina las técnicas de elicitación de acuerdo con las necesidades del cliente. Aplica los instrumentos de recolección de información según las técnicas seleccionadas. Utiliza los estándares de	**Conocimiento:** Técnica: evaluación de conocimiento. Instrumento: Cuestionario **Desempeño** Técnica (observación directa,) Instrumento; (Lista de de chequeo) **Producto**

			mapa del recorrido del proyecto, tarjetas con contenidos reflexivo Mini plantilla ERS alineada con el estándar IEEE 830. mapa visual del proceso propuesto. **Producto:** Documento de la interpretación de los requerimientos asignados (Formato de legalización de requerimientos) Informe de rescate del proyecto, Exposición de infografía interactiva	especificación de requisitos de acuerdo con las características del proyecto. Documenta la especificación de requisitos según las necesidades del cliente..	Técnica: Valoración de producto Instrumento (Lista de chequeo)

5. GLOSARIO DE TÉRMINOS

🔍 **Algoritmo:** Secuencia finita de pasos definidos y no ambiguos que permiten resolver un problema o realizar una tarea.

🔍 **Ambiente de desarrollo:** Conjunto de herramientas que permiten escribir, compilar, ejecutar y depurar programas (por ejemplo, Visual Studio Code, NetBeans, Eclipse).

🔍 **Aprendizaje autónomo:** Proceso mediante el cual el estudiante dirige su propio aprendizaje, gestionando el tiempo y los recursos para alcanzar los objetivos.

🔍 **Aprendizaje significativo:** Tipo de aprendizaje en el que el estudiante relaciona el nuevo conocimiento con conocimientos previos, generando comprensión.

🔍 **Argumento:** Valor que se pasa a una función o procedimiento cuando se llama.

🔍 **Array (Arreglo):** Estructura de datos que almacena múltiples valores del mismo tipo en una sola variable indexada.

🔍 **Compilador:** Programa que traduce código fuente escrito en un lenguaje de programación a un lenguaje máquina.

🔍 **Condicional:** Estructura de control que permite ejecutar distintas instrucciones dependiendo de si se cumple una condición lógica (por ejemplo, if, else).

🔍 **Constante:** Valor que no cambia durante la ejecución de un programa.

🔍 **Debugging:** Proceso de encontrar y corregir errores en el código fuente de un programa.

🔍 **Estructura de control:** Instrucciones que alteran el flujo de ejecución del programa, como condicionales y bucles.

🔍 **Estructura secuencial:** Modelo de programación donde las instrucciones se ejecutan una tras otra en orden.

🔍 **Evaluación diagnóstica:** Instrumento que permite identificar los conocimientos previos del estudiante antes de comenzar una unidad de aprendizaje.

🔍 **Función:** Bloque de código que realiza una tarea específica y que puede ser reutilizado.

🔍 **Identificador:** Nombre que se utiliza para identificar variables, funciones, arreglos, etc., en un programa.

🔍 **Inicialización:** Proceso de asignar un valor inicial a una variable.

🔍 **Input/Output (I/O):** Términos que se refieren a la entrada y salida de datos en un programa.

🔍 **Iteración:** Repetición de un bloque de código un número determinado de veces o hasta que se cumpla una condición.

🔍 **Lenguaje de programación:** Lenguaje formal que permite escribir instrucciones que una computadora puede ejecutar.

🔍 **Lógica de programación:** Conjunto de principios que permiten desarrollar algoritmos y resolver problemas mediante programación.

🔍 **Operador:** Símbolo que representa una operación (por ejemplo, +, -, *, ==).

🔍 **Parámetro:** Variable que se define como parte de la declaración de una función y que recibe valores cuando esta es llamada.

🔍 **Pseudocódigo:** Descripción de un algoritmo utilizando una notación similar a un lenguaje de programación, pero más informal.

🔍 **Sintaxis:** Conjunto de reglas que definen la estructura correcta de las instrucciones en un lenguaje de programación.

🔍 **Tipo de dato:** Clasificación de los datos que indica qué tipo de valores puede almacenar una variable (por ejemplo, int, float, string).

🔍 **Variable:** Espacio de memoria identificado por un nombre, que puede contener un valor que puede cambiar durante la ejecución del programa.

6. REFERENTES BIBLIOGRÁFICOS

- Campus Virtual SENA. (2020). *Guía para la elaboración de actividades de aprendizaje*. Servicio Nacional de Aprendizaje SENA.
- Deitel, H. M., & Deitel, P. J. (2012). *Java: Cómo programar* (9.ª ed.). Pearson Education.
- Forouzan, B. A., & Gilberg, R. F. (2010). *Estructuras de datos: Un enfoque orientado a objetos con Java*. Cengage Learning.

- González Duque, J. (2005). *Lógica computacional: Fundamentos y aplicaciones*. McGraw-Hill.
- Joyanes Aguilar, L. (2012). *Fundamentos de programación: Algoritmos, estructura de datos y orientación a objetos*. McGraw-Hill.
- Pressman, R. S. (2010). *Ingeniería de software: Un enfoque práctico* (7.ª ed.). McGraw-Hill.
- SENA. (2021). *Material didáctico del programa de formación – Programación de Software*. Servicio Nacional de Aprendizaje.

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	ING. ARGEMIRO DE JESÚS OLIVARES MG. MOSQUERA TEHERAN WATCIRA ING. NEIRA GARCIA YEYMI ALEXANDRA	INSTRUCTOR INSTRUCTOR INSTRUCTOR	SENATIC	01/04/2025

8. CONTROL DE CAMBIOS (diligenciar únicamente si realiza ajustes a la guía)

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					