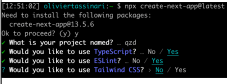
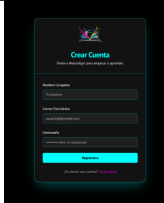
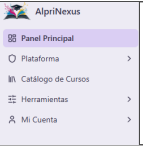
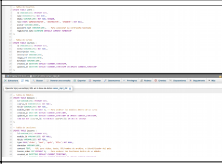
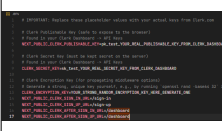

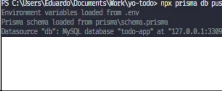




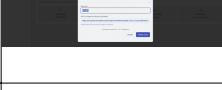
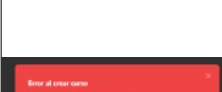
				Código: GFPI-F-147	
				Versión: 04	
PROCESO					
GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL					
NOMBRE DEL FORMATO					
FORMATO BITÁCORA DE SEGUIMIENTO ETAPA PRODUCTIVA					
CLASIFICACIÓN DE LA INFORMACIÓN					
Pública <input type="checkbox"/>		Pública Clasificada <input type="checkbox"/>		Pública Reservada <input checked="" type="checkbox"/>	
Nombre de la persona con rol de aprendiz		Tipo de documento		Número de identificación	
Karol Vanessa Nuñez Vera		C.C		1016594191	
		Teléfono de contacto		Correo electrónico institucional	
		3104519565		karol_vnunez@soy.sena.edu.co	
				vaneauaiskow58@gmail.com	
Número de grupo			Programa de formación		
2930435			Técnico en Control de la Seguridad Digital		
Nombre de la empresa, ente u organización donde está realizando la etapa productiva		NIT		Bitácora N°	
ALL PRINT GRAPHIC & MARKETING SAS - ALPRIGRAMA S.A.S		900167804		10	
				Período a reportar	
				Desde 31/05/2025 hasta 14/06/2025	
Nombre del Ente Coformador (jefe inmediato/responsable/supervisor)		Cargo del Ente Coformador		Teléfono de contacto	
Ángela Patricia Marroquín		Dra. Administrativa y Financiera.		3163539122	
				Correo electrónico	
				angela.marroquin@alprigrama.com	
Datos de la persona con rol de instructor de seguimiento					
Nombre de la persona con rol de instructor de seguimiento:		Mauricio Umbarilla Figueroa		Correo electrónico:	
				etapaproductivacdm@sena.edu.co	
Seleccione con una "X" el tipo de alternativa de etapa productiva que está realizando, teniendo en cuenta el subtipo al cual pertenece si es el caso:					
ALTERNATIVA DE ETAPA PRODUCTIVA		SUBTIPO DE ALTERNATIVA		ALTERNATIVA DE ETAPA PRODUCTIVA	
				SUBTIPO DE ALTERNATIVA	
PROYECTO PRODUCTIVO		Sena - Empresa		CONTRATO VÍNCULO FORMATIVO* *De acuerdo a la Resolución 623 de 2020, emanada por el Mintrabajo, pasantía se denomina vínculo formativo.	
		Sena Proveedor Sena			
		Producción de Centros			
		Proyecto productivo bajo enfoque empresarial o de investigación, desarrollo e innovación I+D+i			
		Proyectos Ruta Emprendedora SENA			
		Proyecto productivo prácticas en la economía popular y/o campesina			
		Proyecto productivo a través de convocatorias SENNOVA			
MONITORÍA		Monitoría (regular)		CONTRATO DE APRENDIZAJE	
		Monitoría a través de convocatorias SENNOVA			
		Otra			
				CONTRATO DE APRENDIZAJE	
				Contrato de aprendizaje (regular)	
				Contrato de aprendizaje a través de convocatorias SENNOVA	
				Contrato de aprendizaje por medio de prácticas en la economía popular y/o campesina	
				Vínculo laboral (regular)	
				Vínculo laboral por medio de prácticas en la economía popular y/o campesina	
DESCRIPCIÓN DE LA ACTIVIDAD (Ingrese cuantas filas sean necesarias)		FECHA DE INICIO		FECHA DE FIN	
				EVIDENCIA DE CUMPLIMIENTO (Indique si corresponde a un documento, proceso, producto, entregable u otro) En anexo puede fortalecer la evidencia si es el caso.	
				OBSERVACIONES, INASISTENCIAS, DIFICULTADES PRESENTADAS	
				Y/O COMENTARIOS REALIZADOS POR LAS PERSONAS CON ROL DE: APRENDIZ Y/O JEFE INMEDIATO	
1. Hoy fue un gran día al iniciar la fase de desarrollo real con Next.js. Se configuró el proyecto Next.js con TypeScript y App Router, estableciendo una base sólida para el desarrollo. a) Configuración de un nuevo proyecto Next.js (create-next-app) con TypeScript y App Router. b) Revisión de la estructura inicial de carpetas generada por Next.js.		31-may		31-may	
				Anexo 1.	
2. Descripción de Actividad: Se integraron ShadCN UI y Tailwind CSS para el diseño visual, y se consolidó la estructura de carpetas estándar. a) Configurar Tailwind CSS en el proyecto. b) Integrar ShadCN UI (instalación e init). c) Copia y personalización de algunos componentes básicos de ShadCN (ej. Button, Card). d) Creación de carpetas como src/contexts/ y prisma/.		1-jun		1-jun	
					
3. Aunque no fue un día laboral, ya que era festivo, se realizaron ciertas acciones. Se implementó una autenticación simulada con AuthContext para gestionar roles y sesiones, y se crearon las páginas principales protegiendo las rutas con AppLayout. a) Creación de AuthContext para simular inicio de sesión y registro con roles. b) Almacenamiento de información de sesión simulada en localStorage. c) Creación de las páginas clave en src/app/. d) Implementación de AppLayout (src/app/(app)/layout.tsx) para proteger rutas y redirigir a /login.		2-jun		2-jun	
					

<p>4. Se desarrolló una barra de navegación lateral dinámica (Sidebar) que muestra ítems de menú según el rol del usuario.</p> <p>a) Desarrollo del componente Sidebar (ShadCN).</p> <p>b) Implementación de la función getNavItemsForRole en src/lib/nav-items.ts.</p> <p>c) Asegurar que la visibilidad de elementos de navegación y el acceso a páginas se controlen por rol.</p>	3-jun	3-jun		La Sidebar que cambia según el rol es superinteresante. Ver cómo la función getNavItemsForRole adapta la interfaz es un ejemplo claro de código flexible y responsivo a los usuarios.
<p>5. Se definió el schema.prisma para estructurar la base de datos con modelos (User, Course, Module, etc.) y sus relaciones.</p> <p>a) Definición de modelos en prisma/schema.prisma (User, Course, Module, Lesson, etc.).</p> <p>b) Definición de relaciones cruciales entre modelos (uno-a-muchos, muchos-a-muchos con UserLessonProgress).</p> <p>c) Uso de atributos de Prisma como @id, @default(uuid()), @default(now()), @updatedAt, @relation, @unique.</p>	4-jun	4-jun		Definir el schema.prisma es un paso gigantesco. Me costó un poco al principio entender todas las relaciones (@relation, uuid(), etc.), pero ahora veo el poder de tener una "fuente de verdad" tan clara para la base de datos.
<p>6. Se conectó la aplicación a MySQL configurando DATABASE_URL y se generó el Prisma Client para consultas tipadas.</p> <p>a) Configuración de DATABASE_URL en el archivo .env para apuntar a MySQL.</p> <p>b) Generación del Prisma Client con npx prisma generate.</p> <p>c) Creación de la instancia import prisma from '@lib/prisma' para realizar consultas.</p>	5-jun	5-jun		Conectar a MySQL con DATABASE_URL y generar el Prisma Client fue un momento "¡eureka!". Ya no son datos de mentira, son datos de verdad. La posibilidad de hacer consultas tipadas es algo que valoro mucho para evitar errores.
<p>7. Se sincronizó el esquema de Prisma con MySQL y se refactorizaron los endpoints de Usuarios y Anuncios para usar la base de datos real a través de Prisma.</p> <p>a) Ejecución de npx prisma db push para aplicar cambios a la base de datos.</p> <p>b) Refactorización de endpoints API para Usuarios (findMany, findUnique, create, update, delete) usando Prisma.</p> <p>c) Refactorización de endpoints API para Anuncios (incluyendo author).</p>	6-jun	6-jun		Reemplazar los .json por las operaciones con Prisma para Usuarios y Anuncios fue un poco tedioso al principio, pero el resultado es una aplicación que se siente mucho más robusta porque interactúa con datos persistentes.
<p>8. Se refactorizaron los endpoints de Cursos para manejar relaciones anidadas con Módulos y Lecciones usando las capacidades de escritura anidadas de Prisma.</p> <p>a) Refactorización de operaciones CRUD para Cursos, manejando relaciones anidadas.</p> <p>b) Uso de instructor: { connect: { id: instructorId } } al crear/actualizar cursos.</p> <p>c) Uso de include para obtener datos del instructor y el número de módulos al leer cursos.</p> <p>d) Manejo de la escritura anidada de Prisma para módulos y lecciones en la edición de cursos.</p>	7-jun	7-jun		Refactorizar los endpoints de Cursos con sus relaciones anidadas fue el mayor reto hasta ahora. Entender connect, include y las operaciones de escritura anidada de Prisma (como upsert) es crucial para manejar datos complejos. Siento que dominé una parte difícil aquí.
9. Domingo no laboral.	8-jun	8-jun	N/A	Día no laboral.
<p>10. Se creó un endpoint simulado para subida de imágenes y se mejoró el manejo de estados de carga y error en el frontend con mensajes de la API.</p> <p>a) Creación del endpoint simulado /api/upload/image.</p> <p>b) Ajuste de funciones fetch en el frontend para consumir datos de Prisma.</p> <p>c) Implementación de useState para isLoading y error en los componentes.</p> <p>d) Mejora de los mensajes de error mostrados al usuario.</p>	9-jun	9-jun		Mantener el endpoint de subida de imágenes simulado (/api/upload/image) ha sido una buena estrategia para no bloquear el avance. También, mejorar los estados de carga (isLoading) y los mensajes de error del frontend es vital para una buena experiencia de usuario.
<p>11. Se actualizaron formularios para enviar IDs correctos en las relaciones, se mejoraron diálogos y se implementaron verificaciones de permisos en el frontend.</p> <p>a) Ajuste de formularios para enviar IDs correctos para las relaciones (ej. authorId, instructorId).</p> <p>b) Mejora de diálogos de confirmación (ej. para eliminar usuarios o anuncios).</p> <p>c) Implementación de verificaciones en el frontend para la lógica de autorización.</p>	10-jun	10-jun		Es importante que los formularios envíen los IDs correctos para las relaciones. La implementación de verificaciones de permisos en el frontend (como que solo el instructor pueda editar su curso) es un buen "complemento" a la seguridad del backend y me hace pensar más en la experiencia del usuario.
<p>12. Se implementó la selección de tema en el perfil del usuario, guardándose en localStorage, y se aseguró la consistencia visual.</p> <p>a) Implementación de la selección de tema en la página de perfil.</p> <p>b) Configuración de variables CSS (colores HSL) en globals.css para los temas.</p> <p>c) Ajuste de tailwind.config.ts para usar estas variables.</p> <p>d) Aplicación de la clase CSS del tema al <html> usando un script en useEffect.</p> <p>e) Asegurar un diseño profesional y limpio con ShadCN y Tailwind.</p>	11-jun	11-jun		Implementar la selección de temas y ver cómo los colores HSL en globals.css y tailwind.config.ts cambian toda la interfaz es muy satisfactorio. Le da un aspecto más profesional y personalizable.
<p>13. Se realizó una depuración intensiva para resolver el error 400 al crear cursos, mejorando la validación y verificando la disponibilidad de user.id.</p> <p>a) Añadir console.log detallados en el endpoint POST /api/courses para inspeccionar req.body.</p> <p>b) Mejorar la validación en el frontend (uso de .trim()).</p> <p>c) Añadir verificaciones para asegurar que user.id esté disponible antes de enviar la solicitud.</p> <p>d) Realizar pruebas de creación de cursos para validar las correcciones.</p>	12-jun	12-jun		El error 400 al crear cursos fue frustrante, pero me enseñó la importancia de console.log detallados y de la validación exhaustiva en el frontend (.trim()), asegurarme de que user.id esté disponible). Resolverlo me dio mucha confianza.
<p>14. Se realizaron correcciones menores, como la importación de buttonVariants, y se ajustó el manejo de datos entre frontend y backend (ej. formato de audience para anuncios).</p> <p>a) Solución del error donde buttonVariants no estaba importado en la página de usuarios.</p> <p>b) Ajustes en cómo se manejan y esperan los datos entre el frontend y el backend.</p> <p>c) Modificación del formato del campo audience para anuncios.</p> <p>d) Revisión de otras pequeñas inconsistencias o errores.</p>	13-jun	13-jun		Pequeños detalles como importar buttonVariants o ajustar el formato de audience muestran que los "fines" son tan importantes como los "grandes inicios". Cada corrección hace que la aplicación sea más sólida.
<p>15. Se realizó una revisión exhaustiva de todas las funcionalidades, la conexión a la base de datos y la aplicación de roles, consolidando el proyecto.</p> <p>a) Revisión general de todas las páginas y funcionalidades.</p> <p>b) Verificación de la persistencia de datos y el funcionamiento del CRUD.</p> <p>c) Confirmación de la aplicación correcta de la lógica de roles (simulada pero con impacto real en la UI).</p>	14-jun	14-jun		Siento que esta revisión general fue un gran cierre. Pude confirmar que todo está conectado y funcionando como debería. Ver la persistencia del CRUD y cómo los roles, aunque simulados, afectan la UI, me dio una gran perspectiva de lo que hemos logrado.

Decreto 055 de 2015, por el cual se reglamenta la afiliación de estudiantes al Sistema General de Riesgos Laborales y se dictan otras disposiciones

Este espacio debe ser siempre diligenciado.

Artículo 11. Obligaciones de la institución de educación. Corresponde a las instituciones de educación a las que pertenezcan los estudiantes, que deban ser afiliados al Sistema General de Riesgos Laborales de conformidad con el presente decreto:

1. Revisar periódicamente que el estudiante en práctica desarrolle labores relacionadas exclusivamente con su programa de formación o educación, que ameritaron su afiliación al Sistema General de Riesgos Laborales.
2. Verificar que el espacio de práctica cuente con los elementos de protección personal apropiados según el riesgo ocupacional.

¿La persona con rol de aprendiz se encuentra afiliado a la ARL?	Indique el nivel de riesgo actual	¿El nivel de riesgo de la ARL corresponde a las actividades que desarrolla la persona con rol de aprendiz en la empresa?	SI	¿La persona con rol de aprendiz cuenta con los elementos de protección personal (EPP), requeridos para desarrollar su etapa productiva?	SI
SI	Nivel 1				

Persona con rol de aprendiz: recuerde diligenciar completamente el formato bitácora y entregarlo o subirlo al espacio asignado para este

Firma de la persona con rol de aprendiz

14/06/2025
Fecha entrega bitácora

Firma de la persona con rol de instructor de seguimiento

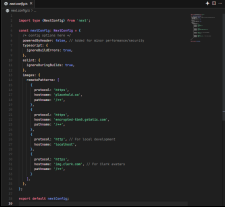
Firma de la persona con rol de jefe inmediato

Nota: Los datos proporcionados serán tratados de acuerdo con la Política de Tratamiento de Datos Personales del SENA y a la Ley 1581 de 2012

Anexo:

Es opcional relacionar evidencia fotográfica de las actividades desarrolladas

(No aplica documentos de la empresa u otros aspectos sensibles)



Anexo 1.