PROYECTO GESTIÓN APRENDICES SENA

Permite gestionar la información de los aprendices de formación titulada que se encuentran vinculados a los programas de formación mediante las fichas de identificación, ademas permite gestionar los llamados de atención generados para cada aprendiz, de acuerdo a las faltas cometidas.
El sistema tiene control de ingreso por usuario y rol, autenticación de 2 factores con token, generación de reportes o informes, exportación de archivos a excel y envío de email al aprendiz desde la api.
El sistema esta desarrollado asi: capa servidor Api REST en node js y express, capa cliente HTML, CSS y Vanilla Javascript

---

Informe de Diseño de Software: Proyecto Gestión Aprendices SENA

1. Introducción

El presente informe detalla el diseño de software propuesto para el **"Proyecto Gestión Aprendices SENA"**. Este sistema busca centralizar y optimizar la administración de la información de los aprendices de formación titulada, incluyendo sus datos de identificación y el registro de llamados de atención por faltas cometidas. Se ha diseñado pensando en la eficiencia operativa, la seguridad de la información y la facilidad de uso para los usuarios.

---

2. Alcance del Sistema

El sistema "Gestión Aprendices SENA" abarcará las siguientes funcionalidades clave:

- **Gestión de Información de Aprendices:**
- Registro, consulta, edición y eliminación de datos de aprendices (fichas de identificación).
- Vinculación de aprendices a programas de formación a través de "fichas de identificación".
- **Gestión de Llamados de Atención:**
- Registro de llamados de atención para cada aprendiz, detallando la falta cometida.
- Consulta del historial de llamados de atención por aprendiz.
- **Control de Acceso y Seguridad:**
- **Control de Ingreso por Usuario y Rol:** Diferentes niveles de acceso al sistema según el rol del usuario.
- **Autenticación de Dos Factores (2FA) con Token:** Mayor seguridad al iniciar sesión.

- **Generación de Reportes e Informes:**
- Creación de reportes personalizables sobre aprendices, programas de formación y llamados de atención.
- Exportación de informes a formato **Excel**.
- **Comunicación:**
- **Envío de Emails:** Funcionalidad para enviar notificaciones o información relevante a los aprendices directamente desde la API.

---

3. Arquitectura del Sistema

El sistema seguirá una arquitectura de tres capas, aprovechando las tecnologías especificadas:

- **Capa de Presentación (Cliente):** Responsable de la interfaz de usuario.
- **Capa de Lógica de Negocio (Servidor/API REST):** Maneja la lógica central de la aplicación, el procesamiento de datos y la comunicación con la base de datos.
- **Capa de Datos:** Almacena y gestiona toda la información del sistema.

3.1. Componentes Principales

- **Cliente (Frontend):**
- Desarrollado con **HTML, CSS y Vanilla JavaScript**.
- Interactuará con la API REST del servidor para enviar y recibir datos.
- Responsable de la renderización de la interfaz de usuario y la experiencia del usuario.
- **Servidor (Backend / API REST):**
- Desarrollado en **Node.js** con el framework **Express**.
- Expondrá endpoints RESTful para todas las operaciones de gestión (CRUD de aprendices, llamados de atención, autenticación, reportes, envío de emails).
- Gestionará la lógica de autenticación (incluido 2FA), autorización por roles y validación de datos.
- Se comunicará con la base de datos.
- **Base de Datos:**
- Un Sistema Gestor de Bases de Datos Relacional (SGBDR) como PostgreSQL o MySQL, o NoSQL como MongoDB, dependiendo de la naturaleza específica de los datos y las necesidades de escalabilidad futuras. Se recomienda un SGBDR para la gestión de datos estructurados de aprendices y llamados de atención.

3.2. Tecnologías Propuestas (según enunciado)

- **Backend:**
- **Node.js:** Entorno de ejecución de JavaScript del lado del servidor.
- **Express.js:** Framework web minimalista y flexible para Node.js, ideal para construir APIs REST.

- **Librerías para 2FA:** Paquetes como speakeasy o otplib para la generación y verificación de tokens TOTP/HOTP.
- **Librerías para Exportación a Excel:** Ej. exceljs o xlsx para la generación de archivos .xlsx.
- **Librerías para Envío de Email:** Ej. Nodemailer para la gestión de envío de correos electrónicos.
- **Librerías para Conexión a DB:** Dependerá de la base de datos seleccionada (ej. sequelize para SQL, mongoose para MongoDB).
- **Frontend:**
- **HTML5:** Estructura de las páginas web.
- **CSS3:** Estilos y presentación visual.
- **Vanilla JavaScript:** Lógica de interactividad del lado del cliente, manejo de eventos y comunicación con la API.

---

4. Módulos del Sistema

4.1. Módulo de Autenticación y Autorización

- **Descripción:** Gestiona el acceso al sistema y los permisos de los usuarios.
- **Funcionalidades:**
- **Inicio de Sesión:** Validación de credenciales de usuario (nombre de usuario/email y contraseña).
- **Autenticación de Dos Factores (2FA):** Después de la autenticación inicial, se solicitará un token generado por una aplicación de autenticación (ej. Google Authenticator) o enviado por otro medio (ej. email/SMS) para verificar la identidad del usuario.
- **Roles de Usuario:** Asignación de roles (ej. Administrador, Instructor, Coordinador) con permisos diferenciados para acceder a funcionalidades específicas.
- **Gestión de Sesiones:** Mantenimiento seguro de la sesión del usuario (ej. mediante JWT - JSON Web Tokens).

4.2. Módulo de Gestión de Aprendices

- **Descripción:** Permite el manejo completo de la información de los aprendices.
- **Funcionalidades:**
- **CRUD de Aprendices:**
- **Crear:** Registrar nuevos aprendices con sus datos de identificación (nombre completo, documento de identidad, programa de formación, ficha de identificación, información de contacto, etc.).
- **Consultar:** Buscar y visualizar la información de aprendices individualmente o por listados (filtrados por ficha, programa, etc.).
- **Actualizar:** Modificar los datos existentes de un aprendiz.
- **Eliminar:** Dar de baja a un aprendiz del sistema (considerar eliminación lógica para mantener historial).

- **Asociación a Fichas/Programas:** Vincular a cada aprendiz a una o varias fichas de identificación y programas de formación.

4.3. Módulo de Gestión de Llamados de Atención

- **Descripción:** Permite registrar y consultar las faltas disciplinarias de los aprendices.
- **Funcionalidades:**
- **Registro de Llamado de Atención:**
- Asociación del llamado a un aprendiz específico.
- Fecha y hora del llamado.
- Descripción detallada de la falta cometida.
- Tipo de falta (ej. leve, grave, gravísima).
- Nombre del instructor o personal que generó el llamado.
- Posibilidad de adjuntar evidencias (opcional).
- **Consulta de Historial:** Visualizar todos los llamados de atención asociados a un aprendiz.

4.4. Módulo de Reportes e Informes

- **Descripción:** Genera visualizaciones y documentos a partir de los datos del sistema.
- **Funcionalidades:**
- **Generación de Reportes Personalizables:**
- Listados de aprendices por programa, ficha, estado.
- Reportes de llamados de atención (por tipo de falta, por aprendiz, por instructor).
- Estadísticas básicas (ej. número de aprendices por programa, faltas más comunes).
- **Exportación a Excel:** Descarga de los datos de los reportes en un formato de hoja de cálculo (.xlsx) para análisis externo.

4.5. Módulo de Comunicación (API de Email)

- **Descripción:** Permite el envío de notificaciones por correo electrónico a los aprendices.
- **Funcionalidades:**
- **Envío Programado/Manual de Emails:** La API expondrá un endpoint para enviar emails, que podría ser usado para:
- Notificar a un aprendiz sobre un nuevo llamado de atención.
- Enviar recordatorios o comunicados generales.
- Confirmación de registro de nuevos aprendices.

---

5. Interfaz de Usuario (UI) y Experiencia de Usuario (UX)

- **Diseño Limpio e Intuitivo:** La interfaz de usuario, al ser desarrollada con HTML, CSS y Vanilla JavaScript, deberá ser diseñada para ser clara, organizada y fácil de navegar.
- **Formularios Sencillos:** Formularios bien estructurados y con validaciones en tiempo real para el registro de aprendices y llamados de atención.
- **Tablas Filtrables y Paginadas:** Para la consulta de listados de aprendices y llamados de atención, permitiendo ordenar, buscar y paginar los resultados.
- **Confirmaciones Visuales:** Mensajes claros de éxito o error para las operaciones realizadas.
- **Adaptabilidad:** Aunque no se especificó un diseño responsive, es recomendable que la interfaz tenga cierta adaptabilidad para funcionar bien en diferentes tamaños de pantalla, facilitando el acceso desde distintos dispositivos.

---

6. Consideraciones Técnicas Adicionales

- **Seguridad:**
- **Protección de Datos Sensibles:** Cifrado de contraseñas de usuarios en la base de datos.
- **Validación de Entradas:** Filtrar y validar todas las entradas de usuario para prevenir inyecciones SQL, XSS, etc.
- **Control de Sesiones:** Gestión segura de tokens de autenticación (JWT) y manejo de la revocación de sesiones.
- **Auditoría:** Posiblemente un registro de logs de acciones importantes realizadas por los usuarios para fines de auditoría.
- **Manejo de Errores:** Implementación de un manejo de errores robusto tanto en el backend como en el frontend, proporcionando mensajes informativos al usuario y registrando los errores internamente.
- **Rendimiento:** Optimización de consultas a la base de datos y de la lógica de negocio para asegurar tiempos de respuesta rápidos.
- **Escalabilidad:** El diseño de la API RESTful facilitará la escalabilidad horizontal del backend si el número de usuarios o la carga aumentan significativamente.
- **Documentación de API:** Es crucial mantener una documentación clara y actualizada de los endpoints de la API (ej. usando Swagger/OpenAPI) para facilitar el desarrollo del frontend y futuras integraciones.

---

7. Próximos Pasos

1. **Definición Detallada de Requisitos:** Profundizar en los requisitos específicos de cada funcionalidad, especialmente en los tipos de datos de los aprendices y las faltas.

2. **Modelado de Base de Datos:** Diseñar el esquema de la base de datos con sus tablas, relaciones e índices.
3. **Diseño de API (Endpoints):** Definir todos los endpoints RESTful, métodos HTTP, estructuras de request/response y códigos de estado.
4. **Prototipado de Interfaz de Usuario:** Crear wireframes y mockups del frontend para visualizar el flujo y la experiencia del usuario.
5. **Desarrollo Iterativo:** Implementar el sistema por fases, priorizando las funcionalidades clave y realizando pruebas continuas.
