ENUNCIADO PROYECTO DESARROLLO WEB:

INDICACIONES:

El proyecto en cuestión deberá ser entregado un fin de semana antes del examen final, la entrega deberá hacerse el **28 de OCTUBRE del año en curso**, deberá tener su proyecto funcionando listo para su presentación según la dinámica a definir previo a la entrega, el enunciado presentado sugiere un proceso de investigación para reforzar lo consignado durante el curso, con el objetivo de que desarrolle habilidades técnicas como profesional de sistemas.

1) Contexto y objetivo

Desarrollar el frontend de un Sistema de Nómina y Gestión de RR. HH. consumiendo una API REST ya existente. El resultado debe ser una SPA robusta, accesible, segura y desplegada en la nube junto con la API y la base de datos, con prácticas de calidad propias de un proyecto profesional.

2) Stack, lineamientos y arquitectura

2.1 Frontend

- React 18 + Vite + TypeScript
- Router: React Router 6
- Estado: React Query (server-state) + Context o Zustand (UI/local state)
- Validación: react-hook-form + Zod
- UI: TailwindCSS, diseño responsive, dark mode opcional
- Accesibilidad: WCAG 2.1 AA
- Autenticación JWT con guards de ruta por rol
- Subida de archivos + exportación de reportes
- Manejo de respuestas HTTP obligatorio en todas las operaciones (200, 201, 204, 400, 401, 403, 404, 422, 500+).

2.2 Integración con la API

- Cliente HTTP con fetch/axios
- Tipos TS generados desde OpenAPI o manualmente
- Seguridad con JWT, expiración y logout automático
- Manejo visible de códigos de error en todas las operaciones

2.3 Opciones de despliegue en la nube

- Frontend: Vercel / Netlify / Cloudflare Pages
- API: Railway / Render / Azure App Service / Fly.io
- Base de datos: Azure SQL Database / PlanetScale / Railway SQL

- Secretos: variables de entorno seguras
- HTTPS habilitado

2.4 Opción alternativa — Máquina Virtual de la Universidad (Windows + IIS)

- Frontend: publicar dist/ como sitio estático en IIS
- API: configurar como aplicación en IIS
- BD: usar gestor instalado (SQL Server / MySQL)
- Configuración IIS recomendada con HTTPS, CORS y variables de entorno

2.5 CI/CD y Calidad (opcional)

- GitHub Actions con build, lint, typecheck
- Deploy automático a preview y producción
- Logs, métricas y reporte Lighthouse

3) Módulos y funcionalidades del Frontend

Los módulos deben cubrir las funcionalidades principales del sistema, asegurando manejo explícito de los códigos de respuesta HTTP en cada operación y cumpliendo con criterios de accesibilidad WCAG 2.1 AA.

3.1 Autenticación y Roles

- Login con usuario/contraseña.
- Almacenamiento de token JWT en memoria/localStorage.
- Rutas protegidas según rol (Administrador, RRHH, Empleado).
- Códigos HTTP requeridos: 200, 401, 403.

3.2 Gestión de Empleados

- CRUD completo.
- Búsqueda avanzada, filtros, paginación.
- Validaciones estrictas en formularios.
- Códigos HTTP requeridos: 200, 201, 204, 400/422, 404, 500.

3.3 Expediente de Empleado

- Carga y descarga de documentos.
- Checklist de requisitos.
- Observaciones de RRHH.
- Códigos HTTP requeridos: 200, 413, 422, 404, 500.

3.4 Módulo de Nómina

- Listado de nóminas por periodo.

- Detalle por empleado.
- Filtros por rango de fechas y departamento.
- Códigos HTTP requeridos: 200, 404, 422.

3.5 Reportes y Exportación

- Reportes de nómina y expedientes.
- Exportación PDF/CSV/XLSX.
- Gráficas opcionales.
- Códigos HTTP requeridos: 200, 204, 500.

3.6 Dashboard por Rol

- KPIs diferenciados por rol (Admin, RRHH, Empleado).
- Códigos HTTP requeridos: 200, 401, 403.

3.7 Usabilidad, Accesibilidad y Experiencia

Además de diseño responsive, notificaciones y dark mode, el sistema debe cumplir con WCAG 2.1 AA:

- Perceptible: texto alternativo en imágenes, contraste mínimo 4.5:1, indicadores visuales.
- Operable: navegación con teclado, focus visible, sin límites de tiempo estrictos.
- Comprensible: formularios con etiquetas, mensajes claros, consistencia en diseño.
- Robusto: uso de roles ARIA, estructura HTML5 semántica, soporte lectores de pantalla.

4) Requisitos no funcionales

- ✓ Rendimiento: LCP < 2.5s, bundle inicial < 300KB
- ✓ Accesibilidad: nivel AA
- ✓ Seguridad: JWT, validaciones, manejo estandarizado de errores HTTP
- ✓ Mantenibilidad: TS estricto
- ✓ Confiabilidad: reintentos y mensajes de error claros

5) Pruebas y calidad técnica (OPCIONAL — Recuperación de puntos)

- ✓ Unitarias con Vitest.
- ✓ Integración UI con React Testing Library.
- ✓ E2E con Playwright (mínimo 5 flujos).
- ✓ CI/CD con GitHub Actions.
- ✓ Lighthouse report.

Nota: si se implementa correctamente, puede recuperar puntos perdidos en otras secciones.

6) Criterios de aceptación (DoD)

- ✓ Autenticación funcional con manejo de 401/403 y redirección.
- ✓ CRUD de empleados con validaciones y manejo de 400/422.
- ✓ Expedientes con carga/descarga y control de 413, 422, 500.
- ✓ Nómina listada y detallada con manejo de 404, 422.
- ✓ Reportes exportables con manejo de 204 y 500.
- ✓ Dashboard diferenciado por rol.
- ✓ Accesibilidad implementada según WCAG 2.1 AA.
- ✓ Frontend, API y BD accesibles desde URLs o IIS.
- ✓ Documento formal UMG entregado.
- ✓ Sección 5 opcional puede compensar fallos.

7) Entregables obligatorios

- ✓ Repositorio GitHub con el frontend.
- ✓ Aplicación en producción (nube o IIS en VM).
- ✓ Manual de Usuario (Front).
- ✓ Manual Técnico (Front).
- ✓ Documento formal UMG con anexos: enlaces a GitHub, frontend publicado, backend, BD, evidencias de pruebas (si existen), demo en video.

El esquema presentado en este enunciado es solamente una guía base, según su análisis podrá agregar los módulos que considere necesarios o modificar los que considere, así como los reportes o algunas otras funcionalidades que considere prudentes en beneficio de la herramienta desarrollada.

Evaluación

Rubrica para evaluar proyecto				
Criterio a calificar	**Principiante 20%**	**Competente 60%**	**Muy competente 100%**	**Punteo**
Funcionalidad 60%	El sistema no realiza las funcionalidades esperadadas, no tiene login, no tiene reportes PDF, no se conecta a base de datos, no maneja errores.	El sistema realiza la funcionalidad forma aceptable, tiene login, genera reportes en PDF, utiliza base de datos, manejo de errores basico.	El sistema realiza la funcionalidad solicitada de forma eficiente, genera reportes PDF, tiene login con encriptacion, se conecta a una base de datos utilizando procedimientos almacenados, manejo de errores de forma eficiente.	9
DERCAS (documentación) 40%	La documentación es deficiente, no cumple con el minimo requerido.	La documentación es aceptable, contiene de forma básica lo requerido. Utiliza git de forma aceptable.	La documentación es detallada y eficiente. Contiene todos los elementos requeridos.	6
Total 100%				15

Fecha de entrega: 28 de octubre 2025