# **Prueba Técnica**

# (Nivel Intermedio)

## Enunciado.

La empresa lo ha contratado para que creen un proyecto de autenticación de usuarios el cual debe de estar creado con .Net 4.5 o .Net Core 6 o superior y este a su vez tendrá que estar integrado a un nuevo aplicativo, uno web que puede estar creado en Asp.Net MVC, Asp.Net Web Forms o Angular

## Primera Parte / Base de Datos

La base de datos debe tener su respectiva normalización, primary keys, foreign keys, etc.

1. Se debe crear la base de datos con el nombre “P1700”.
2. Se debe crear las siguientes tablas: Usuarios, Empleados, Perfil, Permisos y Tiendas.
   1. Cada Empleado tendrá únicamente asociado un Perfil.
   2. Cada Perfil, puede tener asociado varios Permisos.
3. Se debe crear por scripts la siguiente información:
   1. La tabla de Tiendas tendrá las siguientes:
      * Zapote, San Pedro, Heredia, San José.
   2. La tabla de Permisos tendrá los siguientes:
      * Registro de Empleados
      * Consulta de Empleados
      * Consulta de Perfiles
   3. La tabla de Perfiles
      * Administrador (Acceso a Registro de Empleados, Consulta de Empleados, Consulta de Perfiles,).
      * Encargado (Acceso únicamente a la pantalla de Registro de Empleados)
      * Auditoria (Acceso únicamente a las pantallas de Consulta de Empleados y Consulta de Perfiles)

1. Un empleado puede ser un supervisor por día, al igual que el empleado, es decir, se debe modificar la estructura de BD de la tabla Empleados y crear el Foreign Key respectivo que soporte que un empleado sea supervisor y tenga asociado diferentes empleados (Para este punto no se puede crear una tabla nueva).
   1. Una vez ejecutado el cambio, se debe modificar la información de la tabla empleados de forma que:
      1. Al menos 10 registros tengan 1 supervisor.
      2. Al menos 2 empleados deben tener el mismo supervisor.

Ejemplo:

|  |  |  |  |
| --- | --- | --- | --- |
| Supervisor | Empleado que Supervisa |  |  |
| Fabián Pérez Salazar | Gustavo Rojas Arias |  |  |
|  | Sofía Vargas Vargas |  |  |

**Justifique como lo hizo:**

**Se crea un campo nuevo (ID\_SUPERVISOR) en la tabla (EMPLEADOS) con una llave foránea referenciando a su misma tabla (EMPLEADOS).**

**Esto ayudará a que cualquier empleado pueda tener su supervisor.**

1. Se debe crear un Procedimiento Almacenado llamado ObtenerEmpleados con las siguientes consideraciones:
   1. La información para mostrar es el Nombre del Empleado, Teléfono, Fecha, Tipo de Empleado, Salario y el Nombre del Supervisor
   2. Debe utilizar como filtro el código de empleado, es decir, si se usa el filtro, se obtiene la información del empleado, caso contrario, se muestra todos los empleados.

**Justifique como lo hizo:**

**Se crea una vista que obtenga los empleados (EMPLEADOS\_VISTA) y se crea un procedimiento almacenado llamado (SP\_OBTENER\_EMPLEADOS) el cual contiene un IF que valida el parámetro recibido (@ID\_EMPLEADO) y dependiendo si viene un valor null o 0 realiza un select de la vista con todos los empleados, en caso contrario obtiene de la vista solo el registro del ID.**

1. Se debe crear un procedimiento almacenado llamado ObtenerCantidadEmpleadosSupervisados que agrupe la información de forma que muestre únicamente el Nombre del Supervisor y la Cantidad de Empleados que actualmente supervisa.

Ejemplo:

|  |  |  |  |
| --- | --- | --- | --- |
| Supervisor | Cantidad de Empleados Supervisados |  |  |
| Fabián Pérez Salazar | 2 |  |  |

1. Debe de presentar los siguientes scripts de creación de la BD con toda la información requerida para recrear la base de datos, tablas, procedimientos y datos necesarios para la ejecución del proyecto.
   1. 1.Crear-BD.sql (Incluye, BD, Tablas, Relaciones, Procedimientos Almacenados, etc, etc.
   2. 2.Datos.sql (Incluye los datos básicos para las pruebas del sistema)

## Segunda Parte

1. Se requiere hacer una solución de FrontEnd y Backend, para administrar los usuarios y empleados, con la cual se deben tener las siguientes consideraciones:
   1. Las actividades se deben realizar sin la utilización de generadores de códigos o plantillas automáticas.
   2. Cuando se hacen llamados desde el FrontEnd hacia el Backend, debe mostrarse un indicador que permita mostrar al usuario que debe esperar (Loading). (Opcional)

**NOTA: En caso de usar .Net Core 6 o 8 ASP.NET MVC, no es necesario separar el Backend y el Front End, ya que el mismo se ejecutaría en usa sola solución integral.**

## Backend. (.Net / .Net Core)

1. Se requiere hacer una solución con Visual Studio 2022, esta solución debe contener un proyecto Api (.Net o .Net Core) el cual será el Backend para administrar la información, de forma que se debe crear lo siguiente:
   1. Se requiere que esta solución se conecte la base de datos P1700, el tipo de conexión, librerías y arquitectura es de libre elección, sin embargo, la conexión a base de datos debe quedar centralizada y debe poder modificarse desde el web.config, appsetting.json, etc.
   2. Se debe crear el CRUD de la tabla empleados.
   3. Debe tener un Endpoint /User/Registrar que devuelve 201 si se creo o 400 si hubo error.
   4. Debe tener un Endpoint /User/Autenticar que devuelva 200 si autentico o 400 si hubo error, en la respuesta exitosa debe incluir Nombre de Empleado, Perfil y Permisos.
   5. Se debe incluir el uso de documentación Swagger, para poder acceder los métodos directamente del Api sin la necesidad de Postman (Opcional).

Url de referencia en caso de no conocerlo: <https://swagger.io/tools/swagger-ui/>

## Asp.Net WebForms o MVC o Angular

1. Se requiere hacer una solución con Visual Studio 2022. Esta solución debe contener un proyecto de ASP.net el cual será el back office para administrar la información.
2. La pantalla inicial debe tener el login por usuario y contraseña, dos botones, uno para ingresar al sistema y otro que le permita registrarse b como usuario, el registro debe realizarse mediante una ventana emergente (pop-up), solicitando la **cédula del usuario, nombre completo, contraseña, perfil y tienda**).
3. Una vez que exista el usuario debe poder autenticar, en caso de error mostrar mensaje con sweetalert o similar, en caso de éxito desplegar otra pantalla donde tendrá acceso a un menú con las siguientes pantallas:

* **Registro de Empleados (CRUD)**
  + 1. La pantalla principal debe mostrar la Lista de Empleados, dicha información se debe mostrar según la tienda asociada al usuario conectado, es decir, no se pueden ver Empleados de otras tiendas.
    2. Sobre la lista de empleados debe existir un botón que indique Nuevo, el cual permitirá mediante una ventana emergente (pop up) el registro del empleado, el mismo debe estar validado con manejo de errores y no permitir ingreso de datos inconsistentes.
    3. Sobre la lista de empleados, debe existir dos botones:
       - Editar, que permite editar la información reutilizando la misma vista de nuevo registro.
       - Eliminar, mostrando un mensaje de confirmación por parte de sweetAlert antes de proceder a borrar el registro de forma que le permita al usuario aceptar o cancelar la eliminación.
* **Consulta de Empleados: Muestra la información del Procedimiento ObtenerEmpleados**
* **Consulta de Permisos: Muestre la lista de Permisos**

1. **CONSIDERACIONES**
   1. Las actividades se deben realizar sin la utilización de generadores de códigos o plantillas automáticas.
   2. Implementar un sistema de control de acceso basado en roles para restringir ciertas funcionalidades a usuarios con permisos específicos. (Opcional)
      1. Las pantallas en el menú que se visualizarán serán únicamente las que el usuario tenga asociadas según el Perfil. (Opcional)
      2. No se puede ingresar al sistema por el url si no ha hecho sesión anteriormente, de hacerlo debe Redireccionar al Login (Pantalla Principal). (Opcional)
      3. Si se conoce la opción de menú por url, pero el usuario no tiene permiso para verla, no podrá acceder ya que no está autorizado, en este caso se debe redireccionar a una pantalla indicando que No está Autorizado para ver la pantalla. (Opcional)
      4. Agregar la funcionalidad de cierre de sesión para permitir que los usuarios cierren su sesión activa. (Opcional)

**Justifique como lo hizo:**

**<Incluir Justificación acá>**

## Tercera Parte (Opcional, se evalúa la investigación, en caso de no conocerlo)

1. Se requiere hacer una solución de Reporting Services con Visual Studio 2022 o superior.
2. La solución debe mostrar un reporte y consumir el procedimiento almacenado ObtenerEmpleados creado anteriormente.
3. Mostrar en el reporte la información agrupada por fechas, y para cada fecha, se debe mostrar la lista de empleados asignados.

**Justifique como lo hizo:**

**<Incluir Justificación acá>**

## Cuarta Parte (Opcional, se evalúa la investigación, en caso de no conocerlo)

1. Implementar autenticación basada en tokens, utilizando JSON Web Tokens (JWT) para garantizar la seguridad de las comunicaciones entre el cliente y el servidor.
2. Agregar validaciones de entrada para los Endpoints, como verificar que los campos requeridos estén presentes y tengan el formato adecuado.

## Entregable

A continuación, las instrucciones para la entrega:

1. Debe de enviar los proyectos en un .rar o mediante un Github público.
2. Debe tener todas las instrucciones para poder correr el proyecto y probarlo.
3. Debe entregar antes el lunes 30 de setiembre antes de las 8 am.