Documento de Objetivos y Funcionalidades Principales

1. **Objetivos del Proyecto**

   Objetivo General

   Desarrollar una herramienta web que permita monitorear e identificar relaciones entre entidades públicas y proveedores con posibles conflictos de interés y tráfico de influencias en los procesos de contratación pública en Perú, de esta forma ayudando a combatir la corrupción en el sector público.

   Objetivos Específicos

   1. **Integración y Gestión Dinámica de Datos**: Desarrollar un sistema que permita buscar en tiempo real entidades a través de la API del OSCE. El usuario podrá solicitar un análisis en profundidad en la que se extrae datos específicos sobre contratos y entidades relacionadas con dicha entidad, y se almacenará temporalmente en **Neo4j.**
   2. **Desarrollo de Algoritmos de Detección de Influencias:** Implementar algoritmos de redes que en base a ciertos criterios puedan identificar patrones de tráfico de influencias y conexiones sospechosas basándose por ejemplo en la frecuencia y peso de las conexiones.
   3. **Visualización de Redes de Relaciones:** Brindar una interfaz gráfica que permita al usuario navegar en la red, visualizar los puntos críticos (casos de tráfico de influencias) y explorar en los demás nodos y conexiones.

2. **Funcionalidades Principales (MVP)**
   1. **Búsqueda en Tiempo Real e Integración con la API del OSCE**

      Nexet permite que el usuario realice búsquedas en tiempo real de entidades usando la API de OSCE. Cuando el usuario dese un análisis en profundidad entonces el sistema obtendrá datos adicionales sobre contratos y relaciones vinculadas a la entidad seleccionada y estos datos serán almacenados temporalmente en **Neo4j.**

Subfunciones:

1. Consultar la **API** del **OSCE** para obtener datos en tiempo real sobre entidades.
2. Brindar resultados básicos de búsqueda al usuario sin persistencia en **Neo4j**.
3. Cuando el usuario seleccione la opción de análisis, entonces se obtendrá y almacenará en **Neo4j** los contratos y relaciones vinculadas para la consulta activa.

2. **Construcción y Gestión Dinámica de Base de Datos en Neo4j**

Almacenar temporalmente en **Neo4j** los datos relacionados a la consulta activa del usuario, formando una red que nos sirve para el análisis detallado. Entonces cada vez que el usuario inicie una nueva consulta de análisis, el sistema deberá refrescar los datos en **Neo4j**, eliminando los datos anteriores.

Subfunciones:

1. Crear nodos y relaciones en **Neo4j** solo para los datos específicos del análisis seleccionado por el usuario.
2. Al iniciar una nueva consulta entonces debemos eliminar los datos de la consulta anterior y de esta forma evitar el saturamiento de la base de datos con el volcado de datos anterior.
3. Optimizar la gestión de datos en **Neo4j** para el soporte de gestión dinámica.

3. **Implementación de Algoritmos de Detección de Influencias**

Usar algoritmos de análisis de redes para identificar patrones de tráfico de influencias y relaciones anómalas y para ello podemos basarnos en la frecuencia de contratación que será representado como el peso de las conexiones.

Subfunciones:

1. Determinar el peso de las conexiones basándonos en la frecuencia de interacción o contratación.

2. Implementar algoritmos en **NetworkX** para identificar nodos y relaciones sospechosas.
3. Proporcionar alertas o notificaciones al encontrar relaciones sospechosas.

4. Visualización Interactiva de Redes en el Frontend

Proporcionar al usuario una interfaz gráfica interactiva que permita visualizar las relaciones de la entidad seleccionada, detallando los nodos y conexiones en la red.

Subfunciones:

1. Usando **D3.js** mostrar nodos (entidades) y conexiones visualizando un grafo.
2. Proporcionar una opción de exploración detallada para cada nodo, mostrando información adicional.
3. Permitir que el usuario filtre la visualización según criterios específicos, como conexiones sospechosas, por ejemplo.