Para enfrentar este desafío en una hackathon usando JavaScript, junto con tecnologías de procesamiento de lenguaje natural (NLP), podríamos tomar los siguientes pasos:

**1. Definición y Preparación del Set de Datos**

Antes de empezar a codificar, es crucial definir y preparar el set de datos que utilizarás:

* **Fuente de datos**: Identificar y acceder a bases de datos que contengan relatos y registros relevantes para el conflicto armado.
* **Anonimización**: Es esencial anonimizar la información para proteger la identidad de las personas implicadas. Esto puede implicar remover o enmascarar nombres, ubicaciones específicas y cualquier otro dato personal identificable.
* **Etiquetado de datos**: Manualmente o con ayuda de expertos, etiquetar un conjunto de datos que servirá para entrenar el modelo. Los datos deben ser clasificados en categorías como "relacionado al conflicto armado" y "no relacionado".

**2. Herramientas y Librerías de JavaScript para NLP**

Para el procesamiento de lenguaje natural en JavaScript, puedes utilizar librerías como:

* **Natural**: Ofrece herramientas para lenguajes específicos, tokenización, clasificación, entre otros.
* **Compromise**: Una librería que permite manipular y analizar texto en inglés de forma sencilla.
* **Node-nlp**: Librería para entrenar modelos NLP con soporte para varios idiomas, incluyendo clasificación de texto.

**3. Procesamiento de Texto**

Una vez que tienes tu set de datos preparado y tus herramientas listas, el siguiente paso es el procesamiento de texto:

* **Tokenización**: Dividir el texto en oraciones o palabras para facilitar su análisis.
* **Limpieza de datos**: Eliminar ruido en el texto como signos de puntuación innecesarios, palabras de parada, etc.
* **Vectorización**: Transformar el texto en vectores numéricos que puedan ser procesados por modelos de machine learning.

**4. Implementación del Modelo de Clasificación**

Para la clasificación, se pueden implementar modelos simples o complejos dependiendo de la precisión requerida y la disponibilidad de datos:

* **Modelos Simples**: Como regresión logística o Naive Bayes, que son fáciles de implementar y pueden servir como línea base.
* **Modelos más complejos**: Redes neuronales o modelos de aprendizaje profundo que pueden aprender características más complejas del texto.
* **Entrenamiento del modelo**: Usar los datos etiquetados para entrenar el modelo.
* **Evaluación**: Validar el modelo con un set de datos de prueba para verificar su precisión.

**5. Desarrollo de una Interfaz de Usuario**

Para que la herramienta sea accesible a los investigadores o al público, desarrollar una interfaz de usuario simple:

* **Front-end**: Puede ser desarrollado usando HTML, CSS y JavaScript. Frameworks como React o Angular pueden ser útiles para construir interfaces más complejas.
* **Backend**: Node.js puede ser usado para manejar la lógica del servidor, incluyendo las solicitudes de procesamiento de NLP y las respuestas a la interfaz de usuario.

**6. Integración y Despliegue**

Finalmente, integra todas las piezas:

* **APIs**: Desarrollar APIs para interactuar entre el front-end y el back-end.
* **Despliegue**: Utilizar plataformas como Heroku, AWS o Google Cloud para desplegar la aplicación de manera que sea accesible globalmente.

Esta es una hoja de ruta básica para enfrentar el desafío de la hackathon utilizando JavaScript y tecnologías relacionadas. La clave está en comenzar con un prototipo simple y luego iterar rápidamente basado en el feedback y los resultados obtenidos.

### 1. Preparación de Datos

Dado que no podemos trabajar con datos reales de personas desaparecidas aquí, podemos simular este proceso con un conjunto de datos ficticio o utilizar un dataset público similar para fines de demostración. Podrías preparar algunos datos en un archivo CSV o JSON para comenzar.

### 2. Backend: API en Node.js

Instalaremos las librerías necesarias y prepararemos el API para manejar solicitudes y ejecutar el procesamiento de NLP.

### 3. Frontend

Desarrollaremos una interfaz sencilla usando HTML, CSS y JavaScript (o algún framework como React si prefieres).

# **Desafío de la hackathon**

La Unidad de Búsqueda de Personas dadas por Desaparecidas (UBPD) es una entidad del Estado colombiano de carácter **humanitario** y **extrajudicial**que, dentro del Sistema Integral para la Paz, dirige, coordina y contribuye a la búsqueda de personas dadas por desaparecidas en razón y en contexto del conflicto armado.

La Unidad de Búsqueda enfrenta un desafío crítico en el manejo de grandes volúmenes de información textual relacionada con la desaparición forzada y otros crímenes derivados del conflicto armado en Colombia. Esta información es clave para la búsqueda de personas desaparecidas, la identificación de patrones y la generación de datos que puedan acelerar los procesos de justicia y reparación. Sin embargo, gran parte de los datos disponibles provienen de relatos escritos por víctimas, testigos y otras fuentes, lo que dificulta su sistematización y uso eficiente. La falta de una herramienta que permita extraer, categorizar y analizar automáticamente esta información representa un obstáculo importante para la misión de la unidad. Además, existe la necesidad de integrar y consolidar datos de diversas fuentes, como relatos, tablas y referencias geográficas, para construir herramientas avanzadas que permitan consultas complejas y detalladas.

El reto de innovación se enfoca en aprovechar técnicas de procesamiento de lenguaje natural (NLP), inteligencia artificial y modelos de aprendizaje profundo para extraer información clave de los relatos, generar clasificaciones y, en su nivel más avanzado, implementar modelos LLM multimodales para ofrecer una herramienta de consulta por prompt, que permita a los usuarios acceder a información precisa sobre personas desaparecidas y hechos asociados al conflicto.

**Intermedio: 2.** **Analítica y generación de categorías a partir de información del relato.**

**Objetivo:** Desarrollar un modelo de clasificación binaria o múltiple que permita a través del relato identificar aquellos casos que hacen referencia al conflicto armado que por ende serían competencia de la UBPD determinar su suerte y paradero.

* **Requisitos:**
  + Definir el Set de datos para la implementación.
  + Realizar un proceso de anonimización.
  + Realizar un proceso de etiquetado para los datos de entrenamiento.
  + Implementar modelos de clasificación binaria que permita categorizar los registros y determinar la competencia de la UBPD.

Propuesta:

Excelente, comencemos con los primeros pasos en Python para manejar tus datos. Vamos a estructurar el código en varias partes:

1. **Carga de los Datos**: Leeremos los textos desde un archivo o directamente desde una lista si los tienes en ese formato.
2. **Anonimización**: Aplicaremos funciones para anonimizar nombres, lugares y fechas.
3. **Preparación de Datos**: Incluiremos procesamiento básico de texto para prepararlo para el modelo.

**Paso 1: Carga de los Datos**

Primero, necesitaremos cargar los datos en Python. Si tienes los relatos en un archivo (como un archivo .txt o .csv), puedes cargarlos usando pandas. Si no, podemos empezar con una lista simple en Python.

Por ejemplo, aquí está cómo podríamos cargar los datos desde una lista:

python

Copiar código

relatos = [

"Entre los años 2000 y 2002, hubo un bombardeo en la zona de San Juan, en la que murieron varios miembros de un grupo armado ilegal. Debido a esto, dicho grupo amenazó y desplazó forzadamente a J.E.P. y su familia (y probablemente a más familias religiosas) al casco urbano de Pueblito, argumentando que en la zona se había infiltrado un militar haciéndose pasar por líder religioso, quien presuntamente había recogido información que facilitó este ataque en el que murieron un comandante y varios combatientes. Posteriormente al desplazamiento, J.E.P. fue contactado para que volviera a la zona. Viajó con su esposa y, en un sitio conocido como 'El Mirador', hombres armados del grupo ilegal se lo llevaron y le dijeron a la esposa que se fuera de Pueblito. Desde entonces, no se supo más de él. Desapareció el 26 de marzo de 2007."

]

**Paso 2: Anonimización**

Podemos crear una función para anonimizar los relatos. Aquí un ejemplo básico de cómo podríamos hacerlo:

python

Copiar código

def anonimizar\_relato(relato):

# Reemplazar nombres específicos y lugares

relato = relato.replace('San Juan', 'Localidad 1')

relato = relato.replace('J.E.P.', 'Persona 1')

relato = relato.replace('Pueblito', 'Localidad 2')

relato = relato.replace('El Mirador', 'Lugar 1')

# Aquí puedes añadir más reglas según sea necesario

return relato

# Aplicar la anonimización

relatos\_anonimizados = [anonimizar\_relato(relato) for relato in relatos]

**Paso 3: Preparación de Datos**

Para preparar los datos para el modelo, podemos empezar con la tokenización y eliminación de palabras comunes (stop words). Esto se puede hacer utilizando la biblioteca nltk:

python

Copiar código

import nltk

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

# Descargar las palabras stop de nltk

nltk.download('punkt')

nltk.download('stopwords')

stop\_words = set(stopwords.words('spanish'))

def procesar\_texto(texto):

palabras = word\_tokenize(texto, language='spanish')

palabras\_filtradas = [palabra for palabra en palabras if not palabra.lower() in stop\_words]

return ' '.join(palabras\_filtradas)

# Aplicar procesamiento de texto

relatos\_procesados = [procesar\_texto(relato) for relato in relatos\_anonimizados]

Este es un buen punto de partida. Podemos ajustar o expandir estas funciones según tus necesidades específicas. ¿Qué te parece este enfoque? ¿Te gustaría proceder con la implementación de algún modelo específico ahora o hay algo más en la preparación de datos que te gustaría ajustar?