Proyecto de Programación I. Moogle!

Nombre: Carlos Mario Chang Jardínez

Grupo: 113

El proyecto Moogle! consiste en un sistema de recuperación de información que basa su funcionamiento central en el algoritmo TF-IDF y en la implementación de modelos vectoriales para garantizar búsquedas más eficientes y acertadas. Se emplearon un conjunto de lenguajes de programación para lograr la mejor vinculación entre los elementos que conforman el código. Como principal motor en la creación y el trabajo con los algoritmos, se empleó el lenguaje C#, mientras que la interfaz gráfica se encuentra codificada en una plataforma web.

Para garantizar una edición sencilla y la legibilidad del código desarrollado, se utilizaron en el proyecto las clases:

✓ Build, que se encarga de hacer todos los cálculos pertinentes antes de realizar la búsqueda. Encontramos los métodos:

• HacerTF\_BuscarSnipets: se encarga de hacer el TF de todas las palabras y buscar sus respectivos snipets en todos los documentos.

• Rellenar: se encarga de que el arreglo de la cantidad de palabras no empiece en null sino en 0 para que no de errores.

• PicarString: se encarga de eliminar los espacios, y los operadores gramáticos(comas, puntos, exclamaciones, etc)

• RetornarSnipet: se encarga de devolver el snipet de cada palabra.

• Calcular TF: Calcula el TF de cada palabra.

• HacerIDF: Hace el IDF de cada palabra en la base de datos.

• CargarSinonimos: Carga una gran cantidad de palabras con sus sinónimos.

• HacerLista: Hace una lista con todas las palabras de la base de datos sin repetir.

✓ Moogle, se encarga de hacer todos los cálculos pertinentes durante la búsqueda. Encontramos los métodos:

• PicarString: Se encarga de depurar el query y separar las palabras de los operadores que se introdujeron (excepto del de cercanía)

• Sugerir: Si una palabra no se encontró en la base de datos sugiere la palabra de la base de datos que más se parece a la que escribió.

• HacerTFIDF\_Sistema: Se encarga de calcular el TF-IDF de cada documento.

• BuscarCercanía: Si el query contiene el operador cercanía y la base de datos contiene a ambas palabras este método busca la menor distancia entre esas palabras entre todos los documentos y esto se tendrá en cuenta luego para hacer el TF-IDF de cada documento.

• CargarSnipets: Devuelve los snipets de cada documento que tenga puntuación TF-IDF distinta de 0 teniendo en cuenta que esos snipets contengan la palabra más rara del query.

• DevolverBúsqueda: Con todo lo anterior ya habremos obtenido lo necesario para con este método devolver un objeto que contiene el nombre del documento, el snipet y su TF-IDF ordenado de mayor a menor.