

SISTEMA DE RECOMENDACIÓN DE VIAJES

SISTEMAS DE INFORMACIÓN DE GESTIÓN Y BUSINESS INTELLIGENCE



memoria de la aplicación

ainoa santos fernández 71043760a

CURSO 2022-2023

Una captura de pantalla de un videojuego

Descripción generada automáticamente con confianza baja

ÍNDICE

[INTRODUCCIÓN. 2](#_Toc121930531)

[DESCRIPCIÓN DEL PROGRAMA. 3](#_Toc121930532)

[HERRAMIENTAS UTILIZADAS. 4](#_Toc121930533)

[Base de datos. 4](#_Toc121930534)

[Backend. 5](#_Toc121930535)

[Frontend. 5](#_Toc121930536)

[FUNCIONAMIENTO DE LA APLICACIÓN. 7](#_Toc121930537)

[Base de datos. 7](#_Toc121930538)

[Backend. 10](#_Toc121930539)

[Frontend. 10](#_Toc121930540)

[ALGORITMO DE RECOMENDACIÓN. 13](#_Toc121930541)

[Algoritmo de búsqueda. 13](#_Toc121930542)

[Algoritmo con orden establecido. 14](#_Toc121930543)

[Algoritmo por el estilo de viaje. 15](#_Toc121930544)

[Algoritmo colaborativo basado en modelos. 19](#_Toc121930545)

[ANÁLISIS DE RESULTADOS. 21](#_Toc121930546)

[DAFO. 29](#_Toc121930547)

[LÍNEAS FUTURAS. 30](#_Toc121930548)

[LECCIONES APRENDIDAS. 31](#_Toc121930549)

[GUÍA DE INSTALACIÓN DE LA APLICACIÓN. 32](#_Toc121930550)

[REFERENCIAS, BIBLIOGRAFÍA Y WEBGRAFÍA. 33](#_Toc121930551)

# INTRODUCCIÓN.

Después de una pandemia mundial, la gente que se ha tenido que privar de realizar viajes ya sea por salud o por trabajo, esperan con entusiasmo para aprovechar los pocos días que hay de vacaciones. Por este motivo, el hecho de que haya 2 o 3 días juntos libres significa buscar un sitio para visitar, y este es el verdadero objetivo de la aplicación, asesorar a las personas que no saben a donde ir para aprovechar esas vacaciones.

Primero, se expondrán los problemas que se intentan resolver con esta aplicación, segundo se explicarán las tecnologías utilizadas para este fin, y finalmente se mostrará la interfaz creada.

Además, se realizará un análisis del algoritmo de recomendación seguido, y se analizarán los resultados obtenidos.

Como final de esta memoria, se llevará a cabo un análisis crítico de las ventajas que esta aplicación conlleva y también las limitaciones sufridas y las líneas futuras por las cuales se puede seguir si se tuviera más tiempo.

Logotipo

Descripción generada automáticamente con confianza baja

# DESCRIPCIÓN DEL PROGRAMA.

A día de hoy, es muy habitual aprovechar para hacer turismo, ir a descansar o a visitar conocidos y es por ello que los viajes están a la orden del día. Además de ir de vacaciones por ocio o por trabajo, también puede ser por salud. Sea el motivo que sea los viajes son algo muy común en la vida de las personas. Pero llega un momento en el cual no se sabe a dónde ir, o que sitios no se han visitado recientemente. Para evitar que la gente se pierda sitios realmente bonitos y acordes con sus gustos, se ha desarrollado esta aplicación.

Esta aplicación permitirá buscar ciertas características que queremos en nuestro viaje y esta dará resultados afines a nuestra búsqueda. Así ahorrará mucho tiempo a personas que deciden irse de un día para otro, dando opciones de sitios concreto y las cosas mas destacables que ver allí.

La aplicación tendrá cuatro búsquedas distintas, una es mediante las características elegidas por el propio usuario, y la otra es de manera aleatoria dentro de la comunidad autónoma seleccionada (método elegido por comunidad autónoma para acotar por zonas) y que se podrá ordenar mediante la característica que el usuario desee, otra es la búsqueda según el estilo de viaje, ya puede ser en busca de fiesta, de desconexión, de playa o simplemente por aprender de la cultura, y por último, habrá una búsqueda que esté en relación con la edad del usuario, de tal manera que el usuario introduzca su rango de edad y se busquen los sitios cercanos a ese modelo de persona (estos datos se deben obtener tras un gran aprendizaje automático, el cual en tan poco tiempo es imposible de obtener).

Personalmente, siempre me ha gustado organizar los viajes, conocer y documentarme que ver en ese lugar, incluso en qué sitios se come bien o se puede salir a descansar un rato. Sobre todo, me gusta cuando un viaje sale bien y se aprovecha el tiempo, tener esos bonitos recuerdos de sitios verdaderamente impresionantes que, si no mirase antes de ir, probablemente me los perdería. Este es el motivo principal por el cual he tenido esta idea y he escogido este tema.

# HERRAMIENTAS UTILIZADAS.

Para la realización del trabajo, el requisito era el uso de Neo4J para la realización de la base de datos de la aplicación. Para el desarrollo de la aplicación, se debe tener en cuenta que está formado por 3 aspectos importantes, la base de datos, el backend y el frontend.

## Base de datos.

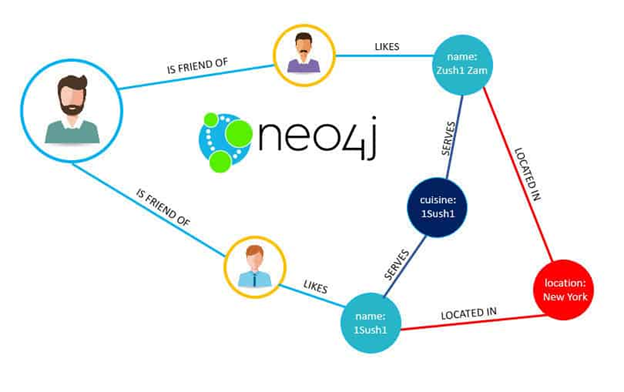
La base de datos es todo el conjunto de los datos que va a usar la aplicación para recomendar los lugares que visitar. En primer lugar, busqué por Kaggle (una herramienta dedicada al almacenamiento de datasets) en el cual no encontré ninguna base de datos que se adecuase a lo que se necesitaba para la aplicación.

Imagen que contiene Aplicación

Descripción generada automáticamente

Es por este motivo es por el cual se decidió generar los datos manualmente, buscando por páginas que recomienden sitios que visitar de cada una de las zonas de España, y dándole los valores que la gente haya opinado en esos foros.

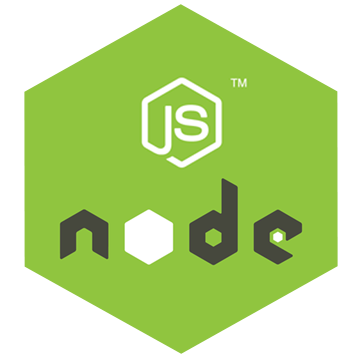
Paran crear y gestionar la base de datos, se ha usado Neo4J, la cual gestiona las bases de datos en forma de grafos y así hace mas sencilla la visualización de estos. Se crean nodos para cada tipo de datos y se relacionan entre si gracias a las relaciones, además tanto los nodos como las relaciones pueden tener propiedades.

Para manejar Neo4J, se debe empezar haciendo los cursos de iniciación, los cuales son básicos y altamente recomendables para entender como funciona y aprender también el lenguaje Cypher. 

## Backend.

El backend es la parte encargada de que la aplicación funcione y reaccione a la lógica de esta. Son el conjunto de acciones que deben ocurrir en la web cuando presionamos botones o elegimos.

Para crear esta parte se ha hecho uso del entorno nodeJS, el cual estructura el backend, y que da la funcionalidad con el código en JavaScript.



Además, se ha hecho uso de ExpressJS, que es un framework para recibir las consultas del frontend, y un driver de Neo4J usado para establecer la conexión del backend con la base de datos, y que permite así realizar peticiones a la base de datos.

Imagen que contiene Logotipo

Descripción generada automáticamente

## Frontend.

El frontend es la parte encargada de interactuar con el usuario, la parte del cliente que contiene la interfaz gráfica. Esta parte realiza las peticiones al backend que busca los datos en la base de datos, y así hace posible realizar las operaciones para mostrar los resultados de la búsqueda en la interfaz.

Para crear esta parte se ha usado el lenguaje JavaScript, que realiza las operaciones para el correcto funcionamiento, VueJS, que permite la creación de aplicaciones web con estructuras de la interfaz.



Además Axios, que es una librería que realiza las peticiones HTTP desde el cliente al backend.



# FUNCIONAMIENTO DE LA APLICACIÓN.

El usuario final de esta aplicación interactuará directamente con el frontend, el cual contiene los elementos para componer la interfaz gráfica de usuario. Desde esta interfaz dependiendo de la acción que realice el usuario, el frontend solicitará cierta información al backend, la cual la solicitará a la base de datos y de ahí se la devolverá al frontend para poder mostrarla por pantalla.

Para usar la aplicación lo primero que hay que tener en cuenta es tener la aplicación de Neo4J, con los datos necesarios cargados desde la base de datos, después se debe activar el backend y por último el frontend.

## Base de datos.

En primer lugar, se ha de crear un nuevo proyecto, y en ese proyecto crear una nueva base de datos, y crear una base de datos local, introducir nombre y contraseña, e introducir en la carpeta import de esa base de datos el archivo .csv. Finalmente iniciar la la base y abrir una terminal de esta.

La base de datos tiene 5 nodos distintos:

* Sitios: son los nodos principales, donde están las características de cada sitio, como son el nombre, la provincia, la calificación de monumentos, naturaleza, fiesta y comida, y los lugares de interés que ver allí.
* Comunidad: son los nodos que contienen el nombre de las comunidades a la que pertenece cada sitio, mediante la relación PERTENECE\_A.
* RuralUrbano: son dos nodos, uno es Rural y otro es Urbano, y habla de esa característica de cada sitio, se relaciona mediante la relación ES\_RURAL\_O\_URBANO.
* CostaInterior: son dos nodos, uno es Costa y otro es Interior, y habla de esa característica de cada sitio, se relaciona mediante la relación ES\_COSTA\_O\_INTERIOR.
* DescansoTurismo: son dos nodos, uno es Descanso y otro es Turismo, y habla de esa característica de cada sitio, se relaciona mediante la relación ES\_DESCANSO\_O\_TURISMO.

Los nodos principales serían los sitios, ósea los nodos por los cuales se hacen las búsquedas y donde se guarda la información que daría la aplicación cumpliendo su objetivo, recomendar sitios que ir a visitar.

Los atributos de estos nodos son los siguientes:

* Nombre: nombre del sitio que visitar.
* Comunidad: comunidad a la que pertenece.
* Provincia: provincia a la que pertenece.
* Monumentos: calificación de 0-10 que se le da al lugar dependiendo de sus monumentos.
* Naturaleza: calificación de 0-10 que se le da al lugar dependiendo de su naturaleza.
* Fiesta: calificación de 0-10 que se le da al lugar dependiendo de su fiesta.
* Comida: calificación de 0-10 que se le da al lugar dependiendo de su comida.
* QueVer: tres sitios de interés más destacados.
* ModeloA: afinidad con el modelo de persona de ese rango de edad (16-24).
* ModeloB: afinidad con el modelo de persona de ese rango de edad (24-32).
* ModeloC: afinidad con el modelo de persona de ese rango de edad (32-40).
* ModeloD: afinidad con el modelo de persona de ese rango de edad (40-55).
* ModeloE: afinidad con el modelo de persona de ese rango de edad (+55).

Al crear yo misma la base de datos, tuve que pasarla del propio Excel, a un archivo .csv, pero el propio Excel separa los datos mediante el uso de punto y coma (“ ; “), mientras que el Neo4J necesita que los datos estén separados por comas (“ , “), entonces me tocó reemplazar todos los punto y coma por comas para que me funcionara de manera correcta.

Para pasar los datos de la base de datos a grafos de conocimiento para que puedan ser manejados por Neo4J creé una sentencia en CYPHER para tomar los datos, donde va creando los nodos, sus atributos y sus relaciones. Crea el nodo sitios, con todos los atributos mencionados, y relaciona con los nodos RuralUrbano, CostaInterior, DescansoTurismo con las relaciones anteriores, y con los nodos Comunidad con la relación de PERTENECE\_A.

Esta query en primer lugar accede al archivo .csv, se indica que va a llamar line a cada línea del archivo, y se establece un límite en 200 para facilitar la búsqueda. Después se crean los nodos Sitios con todos sus atributos, además con cada FOREACH se comprueba que los datos no sean nulos y si no lo son creamos las relaciones necesarias. Además, se usa MERGE y no CREATE con la intención de no poder tener datos duplicados.

LOAD CSV WITH HEADERS FROM 'file:///BaseDeDatosViajes.csv' AS line WITH line LIMIT 200

WITH line WHERE line.Nombre IS NOT NULL AND line.ModeloE IS NOT NULL

MERGE (s:Sitios { nombre:line.Nombre, comunidad:(line.ComunidadAutonoma),

provincia: (line.Provincia), monumentos:toInteger(line.Monumentos),

naturaleza:toInteger(line.Naturaleza), fiesta:toInteger(line.Fiesta),

comida:toInteger(line.Comida), queVer:line.QueVer, modeloA:(line.ModeloA),

modeloB:(line.ModeloB), modeloC:(line.ModeloC),

modeloD:(line.ModeloD), modeloE:(line.ModeloE)})

FOREACH (n IN (CASE WHEN line.ComunidadAutonoma IS NULL THEN [] ELSE [1] END)

| MERGE (c:comunidad {nombre:line.ComunidadAutonoma})

MERGE (s)-[:PERTENECE\_A]->(c)

FOREACH (m IN (CASE WHEN line.RuralUrbano IS NULL THEN [] ELSE [1] END)

| MERGE (r:ruralUrbano {nombre:line.RuralUrbano})

MERGE (s)-[:ES\_RURAL\_O\_URBANO]->(r)

)

FOREACH (m IN (CASE WHEN line.CostaInterior IS NULL THEN [] ELSE [1] END)

| MERGE (i:costaInterior {nombre:line.CostaInterior})

MERGE (s)-[:ES\_COSTA\_O\_INTERIOR]->(i)

)

FOREACH (m IN (CASE WHEN line.DescansoTurismo IS NULL THEN [] ELSE [1] END)

| MERGE (d:descansoTurismo {nombre:line.DescansoTurismo})

MERGE (s)-[:ES\_DESCANSO\_O\_TURISMO]->(d)

)

)

Otras querys que hay que tener en cuenta para la realización de eta práctica son la que elimina todo el grafo (para errores o simplemente meter otra base de datos distinta.

MATCH (n) DETACH DELETE n

Y por último, esta es la query que nos muestre todo el grafo creado.

MATCH (n) RETURN n

Fuegos artificiales de colores

Descripción generada automáticamente con confianza baja

## Backend.

El backend es el encargado de mediar entre la base de datos y la interfaz gráfica o frontend, además procesa las peticiones que recibe del frontend, solicita a la base de datos y obtiene esos datos y los devuelve al frontend para mostrarlos por pantalla.

Para crearlo se usa NodeJS con lenguaje JavaScript, con drivers de express y neo4j. Las funciones del backend están todas en la misma clase de la carpeta backend. Las peticiones para obtener datos (devuelven los datos solicitados) son:

* buscandoSitios: devuelve los sitios que cumplen los requisitos introducidos por el usuario desde la primera parte de búsqueda.
* recomendandoSitios: devuelve los sitios que pertenecen a la comunidad autónoma elegida por el usuario, y ordenados por la característica seleccionada.
* avgFiesta: devuelve la media de la calificación del apartado de fiesta para todo el conjunto.
* busquedaFestividad: devuelve los sitios que están mas cercanos a celebrar festividades, como son aquellos que tienen mejor valoración de fiesta que la media de todo el conjunto, y que además es urbano.
* avgNaturaleza: devuelve la media de la calificación del apartado de naturaleza para todo el conjunto.
* busquedaNaturaleza: devuelve los sitios que están mas cercanos a la desconexión, como son aquellos que tienen mejor valoración de naturaleza que la media de todo el conjunto, y que además sea rural y descanso.
* busquedaPlaya: devuelve los sitios que están en la costa, ordenados por la valoración de la naturaleza.
* avgMonumento: devuelve la media de la calificación del apartado de monumentos para todo el conjunto.
* busquedaCultural: devuelve los sitios que están mas cercanos a la cultura, como son aquellos que tienen mejor valoración de monumentos que la media de todo el conjunto, y que además sea urbano.
* buscandoSitiosRango: devuelve los sitios que son mas afines al modelo de persona según el rango de edad, al que pertenece el usuario.

## Frontend.

El frontend es la parte de la aplicación que el usuario percibe y con la cual él interactúa, esta interfaz se crea mediante VueJS, para dar la estética, y se apoya en la parte lógica programada con JavaScript, y con la ayuda de axios para enviar y recibir peticiones entre Backend y Frontend, mediante peticiones de HTTP.

Como la aplicación debe tener una interfaz sencilla e intuitiva, seleccioné un color claro que personalmente da sensaciones de tranquilidad, y decidí no meter mas colores aparte de los neutros, para no crear una interfaz engorrosa.

La primera página que se ve cuando se entra en la aplicación se basa en un header y una imagen central con un botón que haría avanzar hacia la siguiente pantalla para realizar la búsqueda.

Texto

Descripción generada automáticamente

Al pasar a la siguiente pantalla tras pulsar el ir a la aplicación, podemos ver una pantalla con el mismo header y un footer. La parte principal de la página se subdivide en cuatro partes, una es la búsqueda donde se necesita elegir una comunidad autónoma y se pueden meter otros filtros, y para llevar a cabo esta, se debe presionar Buscar sitios. Otra es la búsqueda seleccionando la comunidad autónoma y mediante que atributo podemos ordenarlo. Otra búsqueda es sobre el estilo concreto de viaje ya sea por fiestas, desconexión, playa, o cultura. Y por último una búsqueda según el rango de edad, que busca la afinidad con los modelos de usuarios que están en la base de datos.

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

Si al hacer una búsqueda no metemos los requisitos mínimos saldrá una pantalla en la cual diga que la búsqueda es incompleta. En el primer caso se debe meter mínimo la comunidad, en el segundo caso la comunidad y el orden, y en el tercer caso es obligatorio rellenar el estilo.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Además, en el footer hay una parte destinada a la ayuda, en caso de no saber cual es el funcionamiento de la aplicación presionando en AYUDA podríamos ver un pequeño resumen. Y otra parte del footer es el apartado de contacto, en el cual hay un enlace que lleva a enviar un correo para poner en contacto, decir las dudas o comentar los errores, desde el bocadillo de la derecha.

Texto

Descripción generada automáticamente

# ALGORITMO DE RECOMENDACIÓN.

## Algoritmo de búsqueda.

En este primer caso, el usuario introduce lo que busca desde la interfaz y así el backend hace la consulta a la base de datos, se va construyendo una query, donde lo mínimo es introducir la comunidad autónoma, y si se selecciona algo de la columna de costa o interior se hace la búsqueda para que se cumpla esa condición, al igual que si esto pasa con la columna de rural o urbano, o descanso o turismo. Además, si se añade un entero en monumentos, naturaleza, fiesta o comida, se busca que el sitio tenga una calificación mayor que la introducida.

Texto

Descripción generada automáticamente

## Algoritmo con orden establecido.

En este caso, el usuario introduce la comunidad de la que busca los sitios, y selecciona la forma en la que ordena los datos.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

## Algoritmo por el estilo de viaje.

En este caso, el algoritmo tiene 4 partes:

**Festividad:** este algoritmo realiza una búsqueda la cual muestra los resultados de aquellos lugares que estén más próximos a las festividades, para ello se obtiene la media de la puntuación del apartado de fiesta, y se escogen solo aquellos lugares que tengan una puntuación superior. Además de ello, se entiende que otro requisito para ello sea que el sitio debe ser urbano. Para el orden de los resultados, se colocan de mejor a peor puntuación de fiesta.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

**Desconexión:** este algoritmo realiza una búsqueda la cual muestra los resultados de aquellos lugares que estén más próximos a la desconexión, para ello se obtiene la media de la puntuación del apartado de naturaleza, y se escogen solo aquellos lugares que tengan una puntuación superior. Además de ello, se entiende que otro requisito para ello sea que el sitio debe ser rural, y también de descanso. Para el orden de los resultados, se colocan de mejor a peor puntuación de naturaleza.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

**Playa:** este algoritmo realiza una búsqueda en la cual solo se muestran aquellos resultados que sean de costa. Para el orden de los resultados, se colocan de mejor a peor puntuación de naturaleza.

Texto

Descripción generada automáticamente

**Cultural:** este algoritmo realiza una búsqueda la cual muestra los resultados de aquellos lugares que estén más próximos a la cultura, para ello se obtiene la media de la puntuación del apartado de monumentos, y se escogen solo aquellos lugares que tengan una puntuación superior. Además de ello, se entiende que otro requisito para ello sea que el sitio debe ser urbano.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

## Algoritmo colaborativo basado en modelos.

Este algoritmo se basa en la cercanía y afinidad de usuarios con características semejantes al usuario actual. Para establecer esta semejanza se han elaborado modelos de usuario con afinidad para cada uno de los lugares. Por ejemplo, si tenemos un usuario de 20 años, buscará en el modelo del usuario que esté en ese rango de edad la afinidad con todos los lugares de la base de datos, y mostrará los resultados acordes a la afinidad con cada uno de ellos.

El caso visto desde un ejemplo sencillo se basaría en una tabla con los lugares y los modelos de usuario como esta:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Mod usuario 1 | Mod usuario 2 | Mod usuario 3 | Mod usuario 4 |
| Lugar 1 | 1 | 4 | 5 | 4 |
| Lugar 2 | 9 | 8 | 5 | 3 |
| Lugar 3 | 4 | 7 | 2 | 3 |
| Lugar 4 | 6 | 3 | 4 | 1 |

En este caso, el usuario 1 representa los usuarios en el rango de edad entre 16 y 20 años, el usuario 2 entre 20 y 30, el usuario 3 entre 30 y 50 y el usuario 4 de 50 en adelante, los lugares son cada uno de los lugares de la base de datos y los números de por medio son la afinidad entre el modelo de usuario y el lugar.

Si entra a buscar una recomendación un usuario de 23 años, los sitios recomendados serán los lugares en orden Lugar 2, Lugar 3, Lugar 1 y Lugar 4.

Para aplicar este algoritmo a esta aplicación se ampliarían los lugares al total de lugares de la base de datos y los modelos de usuario serían mas detallados, con rangos de edad menores, o simplemente con mas características a tener en cuenta para saber a que modelo pertenece el usuario actual.

Se debe tener en cuenta de que para elaborar esta parte de la tabla se debe llevar a cabo un aprendizaje automático profundo con la colaboración de personas de todos los rangos y que opinen sobre cuanta afinidad tienen con cada uno de los lugares. Después de tener todos los datos se debería sacar una media de cada rango para cada lugar concreto. Esto todo necesita mucho tiempo, muchas personas que colaboren para dar su opinión, y muchos mas programas para almacenar y sacar los resultados de la agrupación de estos datos. Como para el tiempo que se tiene para esta asignatura esto no es posible, se entiende que en una aplicación de uso real se tendría que realizar todo ese estudio, pero para este caso vamos a rellenar la tabla de una manera aleatoria para poder usar el algoritmo.

Texto

Descripción generada automáticamente

# ANÁLISIS DE RESULTADOS.

Vamos a ver 8 ejemplos distintos de búsquedas:

PRIMER EJEMPLO:

Vamos a introducir como comunidad autónoma Castilla y León, y vamos a seleccionar la opción de Rural y Descanso, y usaremos el primer tipo de búsqueda:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

El resultado sería este:

Imagen que contiene Calendario

Descripción generada automáticamente

SEGUNDO EJEMPLO:

Vamos a introducir ahora como comunidad autónoma Andalucía, con la opción de costa y que tenga mas de un 6 en naturaleza, y usaremos el primer tipo de búsqueda.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

El resultado sería este:

Texto

Descripción generada automáticamente con confianza baja

TERCER EJEMPLO:

Vamos a introducir ahora como comunidad autónoma Galicia, con la opción ordenar por Nombre del sitio, y usaré el segundo tipo de búsqueda.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Este sería el resultado:

Una captura de pantalla de una computadora

Descripción generada automáticamente

CUARTO EJEMPLO:

Vamos a introducir ahora como comunidad autónoma País Vasco, con la opción ordenar por Comida, y usaré el segundo tipo de búsqueda.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Este sería el resultado:

Imagen que contiene Tabla

Descripción generada automáticamente

QUINTO EJEMPLO:

Vamos a introducir ahora como tipo de viaje Festividad, y usaré el tercer tipo de búsqueda.

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

Este sería el resultado:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

SEXTO EJEMPLO:

Vamos a introducir ahora como tipo de viaje Cultural, y usaré el tercer tipo de búsqueda.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Este sería el resultado:

Imagen que contiene Calendario

Descripción generada automáticamente

SEPTIMO EJEMPLO:

Vamos a introducir ahora como rango de edad Entre 16 y 24, para comparar con el modelo de usuario al que pertenece por el rango de edad, y usaré el cuarto tipo de búsqueda.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Este sería el resultado:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

OCTAVO EJEMPLO:

Vamos a introducir ahora como rango de edad Entre 40 y 55, para comparar con el modelo de usuario al que pertenece por el rango de edad, y usaré el cuarto tipo de búsqueda.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Este sería el resultado:

Imagen que contiene Tabla

Descripción generada automáticamente

# DAFO.

El análisis DAFO se basa en buscar las Debilidades, Amenazas, Fortalezas y Oportunidades.

|  |  |
| --- | --- |
| Debilidades  -La base de datos no es demasiado grande, por ello la recomendación puede ser un poco escueta. | Amenazas  -La aplicación es sencilla, y es fácil de implementar. |
| Fortalezas  -La interfaz es sencilla y bastante intuitiva.  -Los algoritmos son sencillos y fáciles de implementar.  -Distintos tipos de recomendación, rápidos. | Oportunidades  -La base de datos es fácil de ampliar, ya que es sencilla. |

**Debilidades:**

* La base de datos no es lo suficientemente grande para que la recomendación sea mas aleatoria y con más probabilidad de ofrecer lo que el usuario pueda realmente buscar. Además, al ser tan escasa podrían repetirse las recomendaciones, o llegar a un punto que si se meten muchos atributos podría no llegar a recomendar nada.

**Amenazas:**

* La aplicación no es complicada, y es fácil de implementar es por ello por lo que podría implementarla cualquier persona, y hacerle competencia. Además, de no ser muy innovadora.

**Fortalezas:**

* La interfaz es sencilla y bastante intuitiva, eso hace que cualquier persona sin conocimientos previos pueda hacer uso de ello.
* Los algoritmos son sencillos y fáciles de implementar, al ser fáciles es poco probable que fallen siempre y cuando el resto de los componentes funcione correctamente.
* Distintos tipos de recomendación hacen que el usuario si no sabe muy bien que buscar pueda encontrar una solución, y además que esta sea rápida.

**Oportunidades:**

* La base de datos sería fácil de ampliar, ya que al fin y al cabo solo habría que buscar y scrapear distintas páginas donde den su valoración y sacar los valores.

# LÍNEAS FUTURAS.

Una opción de desarrollo de esta aplicación sería ir incorporando mas opciones de estilos, que incluya otro tipo de viajes que la gente suela buscar para así llegar a mas usuarios, o al menos dar soluciones más concretas.

Otra posible opción sería ampliar el rango de sitios que se incluye, actualmente la base de datos solo incluye sitios de España peninsular, por eso podría ampliarse a las islas o a otros países.

Otra opción posible podría ser incluir otros atributos como puede ser el precio del lugar, o la lejanía respecto a un punto concreto.

También se podría agregar un usuario que almacene aquellos sitios que se deseen guardar, o darle a like, y así tener un historial donde almacenar sitios que se tengan ganas de ir a visitar.

Y por último, se podría llevar a cabo el aprendizaje automático para rellenar la afinidad de cada modelo de usuarios con los lugares, teniendo en cuenta las opiniones de personas reales para tener una versión mas real de la afinidad.

# LECCIONES APRENDIDAS.

En esta asignatura me di cuenta de que he podido hacer una aplicación y aprender el funcionamiento de distintos frameworks, además de conseguir enlazar una base de datos de tal manera que se puedan usar esos datos para llegar a hacer un sistema recomendador, con distintos tipos de algoritmos de búsquedas.

Respecto a haber aprendido cosas de forma autodidacta, es algo que al fin y al cabo te acaban enseñando que se vuelve necesario, que no siempre vas a tener alguien que te diga haz esto y aquello de esta manera, sino que tengas que hacer algo y tengas que buscar como llegar a implementar la solución sin ayuda de un profesional en el campo. Además, gracias a esta asignatura he aprendido lo que son los grafos de conocimiento y como manejarlos, y que consultas hay que hacer para obtener los datos de estos grafos.

Otro punto a favor es que me he ido planteando semana por semana alcanzar un hito, y así es como he conseguido acabar la aplicación pronto y sin demorarme en el tiempo, por que me he autoexigido tener las cosas solucionadas para una semana concreta.

Este trabajo ha sido un trabajo que ha necesitado mucho tiempo, y mucha investigación del uso de aplicaciones, herramientas y lenguajes probablemente nuevos para la mayoría, pero también ayuda a ampliar el horizonte de manejo de esas aplicaciones y lenguajes.

# GUÍA DE INSTALACIÓN DE LA APLICACIÓN.

Para usar la aplicación lo primero que debemos tener es, la aplicación de Neo4J, crear un proyecto, una base de datos y una base de datos local, meter en archivo .csv y arrancarla, después de esto, abrir la terminal y usar la query para cargar los datos en el Neo4J.

Después de esto, se debe descargar el repositorio de GitHub donde están la carpeta backend y frontend, ambas se deben arrancar y cada una en una terminal distinta.

En primer lugar, vamos a ejecutar el backend, para ello en una terminal abriremos la carpeta (cd backend), después se ejecutará el comando npm install, y finalmente ejecutaremos el comando nodemon app.js, o node app.js. Habremos iniciado el backend cuando veamos en esa terminal un mensaje que ponga App funcionando.

En segundo lugar, vamos a ejecutar el frontend, para ello en otra terminal abriremos la carpeta (cd frontend), después se ejecutará el comando npm install, y finalmente ejecutaremos el comando npm run serve. EL frontend estará listo cuando carguen todas las dependencias y aparezca un mensaje con App running at:…

Con esto quedaría la aplicación lista para usar.

# REFERENCIAS, BIBLIOGRAFÍA Y WEBGRAFÍA.

Kaggle: <https://www.kaggle.com/datasets>

Curso Neo4J: <https://graphacademy.neo4j.com/?ref=redirect>

Node.js: <https://nodejs.org/es/>

Express: <https://expressjs.com/>

Driver Neo4J: <https://neo4j.com/docs/api/javascript-driver/current/>

Vue.js: <https://vuejs.org/>

Axios: <https://github.com/axios/axios>

Neo4J: <https://neo4j.com/download/>

GitHub: <https://github.com/asant05/SIBI-Viajes>