Μηνάς Ελευθερίου ΑΜ: 4245, Ελευθέριος Κύρκος ΑΜ:4215

Μηχανή Αναζήτησης Τραγουδιών

GitHub repository link: https://github.com/MinasE98/song-search-engine.git

Η ανάκτηση δεδομένων είναι το έργο της εύρεσης σχετικών πληροφοριών από ένα μεγάλο σύνολο δεδομένων. Σε αυτό το έργο, χρησιμοποιήσαμε το Apache Lucene, το οποίο είναι μια δημοφιλής βιβλιοθήκη μηχανών αναζήτησης ανοιχτού κώδικα, για να δημιουργήσουμε μια απλή μηχανή αναζήτησης για ένα σύνολο δεδομένων τραγουδιών από το Spotify. Ο στόχος αυτού του έργου είναι να επιτρέψει στους χρήστες να αναζητήσουν τραγούδια με βάση τις λέξεις-κλειδιά στους στίχους και να εμφανίσουν τα αποτελέσματα σε μια φιλική προς τον χρήστη γραφική διεπαφή (GUI).

Σύνολο Δεδομένων:

Το σύνολο δεδομένων περιέχει πληροφορίες για περισσότερα από 44000 τραγούδια, πάνω από 600 καλλιτέχνες και περιλαμβάνει τα εξής πεδία:

1. artist: Ο καλλιτέχνης που ερμηνεύει το τραγούδι.
2. song: Ο τίτλος του τραγουδιού.
3. link: Ο σύνδεσμος προς το τραγούδι στο διαδίκτυο.
4. text: Οι στίχοι του τραγουδιού.

Το αρχείο είναι σε μορφή CSV.

Ανάλυση κειμένου και κατασκευή ευρετηρίου:

Για την δεύτερη φάση του project έχουμε επιλέξει όλα τα έγγραφα από τη συλλογή τραγουδιών που χρησιμοποιούμε για την εργασία μας. Τα έγγραφα προέρχονται από τη βάση δεδομένων του Spotify (πηγή: https://www.kaggle.com/datasets/notshrirang/spotify-million-song-dataset).

Το έργο υλοποιήθηκε χρησιμοποιώντας γλώσσα προγραμματισμού Java και τη βιβλιοθήκη Apache Lucene. Το σύνολο δεδομένων που χρησιμοποιήθηκε για αυτό το έργο ήταν ένα αρχείο CSV που περιείχε πληροφορίες σχετικά με τραγούδια του Spotify, συμπεριλαμβανομένου του καλλιτέχνη, του ονόματος τραγουδιού, του συνδέσμου και των στίχων. Το αρχείο CSV διαβάστηκε χρησιμοποιώντας τη βιβλιοθήκη OpenCSV και τα δεδομένα ευρετηριάστηκαν χρησιμοποιώντας την κλάση IndexWriter του Lucene. Η κλάση StandardAnalyzer χρησιμοποιήθηκε για τη δημιουργία διακριτικών των δεδομένων κειμένου και τη δημιουργία ενός ευρετηρίου των όρων. Η κύρια μέθοδος δημιουργεί ένα ευρετήριο χρησιμοποιώντας τη μέθοδο createIndex, η οποία διαβάζει τα δεδομένα τραγουδιού από το αρχείο CSV και το προσθέτει στο ευρετήριο χρησιμοποιώντας τη μέθοδο addDoc. Το ευρετήριο αποθηκεύεται σε έναν ByteBuffersDirectory, ο οποίος είναι ένας τύπος καταλόγου στη μνήμη που παρέχεται από τη Lucene.

Αναζήτηση:

Η λειτουργία αναζήτησης υλοποιήθηκε χρησιμοποιώντας τις κλάσεις QueryParser και IndexSearcher του Lucene. Η μέθοδος actionPerformed καλείται όταν ο χρήστης κάνει κλικ στο κουμπί αναζήτησης και εκτελεί την αναζήτηση χρησιμοποιώντας την κλάση QueryParser από το Lucene. Ο χρήστης μπορεί να εισαγάγει ένα ερώτημα αναζήτησης σε ένα πλαίσιο κειμένου και τα αποτελέσματα εμφανίζονται σε μια περιοχή κειμένου κάτω από το πλαίσιο ιστορικού αναζήτησης. Ο χρήστης μπορεί επίσης να επιλέξει το πεδίο για αναζήτηση από ένα αναπτυσσόμενο μενού, συμπεριλαμβανομένου του καλλιτέχνη, του

ονόματος τραγουδιού, του συνδέσμου ή των στίχων. Το ιστορικό αναζήτησης εμφανίζεται στην αριστερή πλευρά του GUI, επιτρέποντας στον χρήστη να δει προηγούμενες αναζητήσεις.

Για να βελτιωθεί η εμπειρία χρήστη, προστέθηκε ένα κουμπί "Επόμενη σελίδα", επιτρέποντας στον χρήστη να δει περισσότερα αποτελέσματα εάν είναι περισσότερα από δέκα. Το κουμπί αυξάνει έναν μετρητή και τα αποτελέσματα ενημερώνονται με βάση τον τρέχοντα αριθμό σελίδας.

Παρουσίαση Αποτελεσμάτων:

Η μηχανή αναζήτησης που προκύπτει παρέχει έναν απλό και διαισθητικό τρόπο στους χρήστες να αναζητούν τραγούδια του Spotify με βάση τις λέξεις-κλειδιά στους στίχους. Η γραφική διεπαφή χρήστη διευκολύνει την εισαγωγή ερωτημάτων αναζήτησης και την προβολή των αποτελεσμάτων και υλοποιείται χρησιμοποιώντας Java Swing. Η λειτουργία ιστορικού αναζήτησης επιτρέπει στους χρήστες να έχουν γρήγορη πρόσβαση στις προηγούμενες αναζητήσεις τους, καθιστώντας πιο εύκολη την τελειοποίηση των ερωτημάτων αναζήτησής τους.

Ημιτελής Λειτουργίες:

Ωστόσο, υπάρχουν ορισμένα ημιτελή μέρη στον κώδικα που χρειάζονται περαιτέρω εξήγηση.

Πρώτον, ο κώδικας περιλαμβάνει μια μερική υλοποίηση μιας δυνατότητας ιστορικού αναζήτησης. Διατηρεί μια λίστα ερωτημάτων αναζήτησης στη μεταβλητή searchHistory και τα ερωτήματα εμφανίζονται στην περιοχή κειμένου searchHistoryBox. Ωστόσο, ο κώδικας δεν χειρίζεται τη διατήρηση του ιστορικού αναζήτησης σε πολλαπλές εκτελέσεις προγραμμάτων. Η προσθήκη λειτουργικότητας για αποθήκευση και φόρτωση του ιστορικού αναζήτησης προς/από ένα αρχείο ή μια βάση δεδομένων θα βελτίωνε τη χρηστικότητα της μηχανής αναζήτησης.

Δεύτερον, ο κώδικας περιλαμβάνει ένα nextButton ActionListener που είναι υπεύθυνο για την εμφάνιση της επόμενης σελίδας των αποτελεσμάτων αναζήτησης. Ωστόσο, η υλοποίηση δεν είναι ολοκληρωμένη, καθώς ανακτά μόνο το επόμενο σύνολο αποτελεσμάτων με βάση τον τρέχοντα αριθμό σελίδας. Δεν χειρίζεται την περίπτωση που ο χρήστης φτάνει στο τέλος των αποτελεσμάτων ή παρέχει έναν τρόπο πλοήγησης στις προηγούμενες σελίδες. Η ολοκλήρωση της δυνατότητας σελιδοποίησης με την εφαρμογή πλοήγησης σε προηγούμενες σελίδες και τον χειρισμό του τέλους των αποτελεσμάτων θα βελτιώσει την εμπειρία του χρήστη.

Τρίτο, μια ημιτελής πτυχή του παρεχόμενου κώδικα είναι η απουσία λειτουργίας επισήμανσης. Στα συστήματα ανάκτησης πληροφοριών, η επισήμανση των αντιστοιχισμένων όρων στα αποτελέσματα αναζήτησης μπορεί να βελτιώσει σημαντικά την εμπειρία του χρήστη παρέχοντας οπτικές ενδείξεις για τη συνάφεια των αποτελεσμάτων. Ενώ ο κώδικας ανακτά και εμφανίζει με επιτυχία τα αποτελέσματα αναζήτησης, δεν περιλαμβάνει τη λειτουργία επισήμανσης των αντιστοιχισμένων όρων στα εμφανιζόμενα αποτελέσματα.

Η εφαρμογή της δυνατότητας επισήμανσης θα συνεπαγόταν την τροποποίηση της εμφάνισης των αποτελεσμάτων αναζήτησης στο resultBox JTextArea για να τονιστούν οι αντιστοιχισμένοι όροι. Αυτό μπορεί να επιτευχθεί με την αντικατάσταση ή τη μορφοποίηση των αντιστοιχισμένων όρων με διαφορετικό χρώμα ή στυλ γραμματοσειράς. Ωστόσο, ο τρέχων κώδικας δεν περιέχει μεθόδους για την εκτέλεση αυτής της επισήμανσης. Για να ενσωματώσει τη λειτουργία επισήμανσης, ο κώδικας θα μπορούσε να χρησιμοποιήσει τις κλάσεις επισήμανσης του Lucene, όπως το Highlighter και το

SimpleHTMLFormatter. Αυτές οι κλάσεις παρέχουν μεθόδους για τη δημιουργία επισημασμένων αποσπασμάτων κειμένου, υποδεικνύοντας πού βρέθηκαν οι όροι αναζήτησης. Μετά από πολλές προσπάθειες οι λέξεις οι οποίες επισημαίνονταν στα αποτελέσματα ήταν λάθος, δηλαδή δεν ήταν η λέξη που χρησιμοποιήσουμε για την αναζήτηση. Ως εκ τούτου αποφασίσαμε να αφαιρέσουμε αυτές τις γραμμές κώδικα ώστε το project που παρουσιάζουμε να μην εμφανίζει λάθος και μη αναμενόμενα αποτελέσματα.

Τέλος, ο κώδικας δεν περιλαμβάνει χειρισμό σφαλμάτων ή διαχείριση εξαιρέσεων. Σενάρια σφαλμάτων όπως δεν βρέθηκε αρχείο, μη έγκυρα ερωτήματα ή εξαιρέσεις κατά τη δημιουργία ευρετηρίου/αναζήτησης δεν αντιμετωπίζονται σωστά. Η προσθήκη κατάλληλου κώδικα διαχείρισης σφαλμάτων και διαχείρισης εξαιρέσεων, μαζί με σημαντικά μηνύματα σφάλματος ή προτροπές χρήστη, θα έκανε τη μηχανή αναζήτησης πιο εύρωστη και φιλική προς το χρήστη.

Συνολικά, το έργο μας για τη μηχανή αναζήτησης τραγουδιών καταδεικνύει τις δυνατότητες του Apache Lucene στη δημιουργία αποτελεσματικών και φιλικών προς το χρήστη εφαρμογών αναζήτησης. Προσφέρει στους χρήστες έναν απλό και διαισθητικό τρόπο να ανακαλύψουν τραγούδια με βάση τους στίχους τους, δίνοντάς τους τη δυνατότητα να εξερευνήσουν και να απολαύσουν την τεράστια συλλογή μουσικής στη βάση δεδομένων Spotify. Με περαιτέρω βελτιώσεις, η μηχανή αναζήτησης θα μπορούσε να χρησιμεύσει ως πολύτιμο εργαλείο για τους λάτρεις της μουσικής, τους ερευνητές και οποιονδήποτε επιθυμεί να εμβαθύνει στον κόσμο των στίχων τραγουδιών.