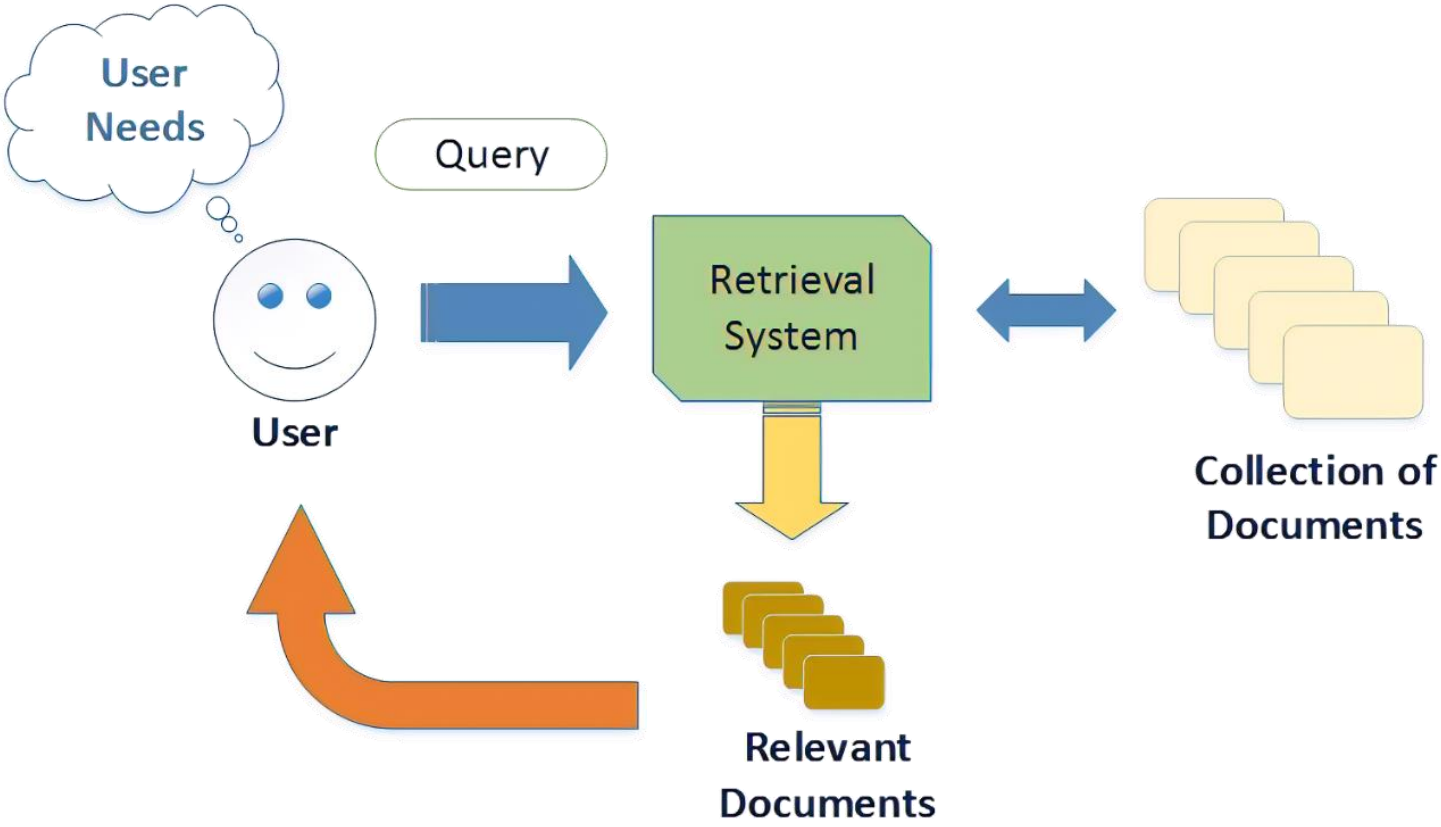
Ανάκτηση Πληροφορίας

Ονοματεπώνυμο : Καρρά Αντωνία

A.M. : 4075



Περιγραφή πρώτης φάσης

Τρόπος συλλογής των δεδομένων

Για την δημιουργία της συλλογής δεδομένων η οποία θα περιλαμβάνει πληροφορίες πεντακοσίων ταινιών , συνέλεξα τα δεδομένα τους κάνοντας scrapping από το IMDB και συγκεκριμένα από το παρακάτω link: top 500 Greatest movies of all time - IMDb, με την χρήση του εργαλείου Beautiful Soup, μια βιβλιοθήκη της Python, η οποία σου επιτρέπει να εξάγεις δεδομένα από οποιαδήποτε ιστοσελίδα. Ακόμη έκανα import και την βιβλιοθήκη request, η οποία μας δίνει την δυνατότητα να πραγματοποιήσουμε HTML requests στην ιστοσελίδα που μας ενδιαφέρει με την εντολή `page = requests.get(url)` , όπου url, το URL της ιστοσελίδας. Επειδή η ιστοσελίδα αποτελείται από πέντε σελίδες με εκατό ταινίες η κάθε μια, δημιούργησα πέντε διαφορετικά προγράμματα τα οποία σχηματίζουν ένα csv αρχείο το καθένα και στην συνέχεια με την βοήθεια ενός αρχείο merge_csv.py, γίνεται η ένωση των επιμέρους αρχείων σε ένα, το films_DataBase.csv

Ύστερα, για την εξαγωγή των δεδομένων αναζήτησα τον HTML κώδικα της ιστοσελίδας και τα κατάλληλα elements των οποίων τα arguments θέλω να βρω. Τα arguments περιέχουν την πληροφορία που θα χρειαστώ για την δημιουργία του corpus.

Πιο συγκεκριμένα, τα arguments αυτά αποτελούν και τα πεδία της συλλογής των δεδομένων που θέλω να δημιουργήσω και τα οποία είναι τα ακόλουθα:

- Title
- Plot
- Year
- Genre
- Runtime

Τα παραπάνω fields για την κάθε ταινία βρέθηκαν με την βοήθεια της συνάρτησης find**()** του Beautiful Soup, ψάχνοντας τα κατάλληλα elements και class που περιέχουν την πληροφορία που μας ενδιαφέρει. Για παράδειγμα η παρακάτω εντολή

```
runtime = film.find('span',class_='runtime').get_text().replace('\n','')
```

ψάχνει στον HTML κώδικα το element = ‘span’ με το όνομα της κλάσης class_='runtime', το οποίο μας εμφανίζει την διάρκεια όλων των ταινιών που έχουμε στην συλλογή μας. Με ανάλογο τρόπο βρίσκουμε και τα υπόλοιπα πεδία. Το συγκεκριμένο παράδειγμα μας τυπώνει

```
Title:Pulp Fiction
runtime:154 min
Title:Casablanca
runtime:102 min
Title:The third man
runtime:104 min
Title:Star Wars: Episode 5 - The Empire Counterattack
runtime:124 min
Title:Schindler's List
runtime:195 min
Title:Revelation now!
runtime:147 min
Title:The good guys
runtime:145 min
Title:Singing in the rain
runtime:103 min
```

Αφού λοιπόν εξήγαμε τα δεδομένα πρέπει να δημιουργήσουμε ένα csv αρχείο, στο οποίο θα αποθηκεύουμε τα πεδία και τις τιμές τους. Αυτό θα πραγματοποιηθεί με την βοήθεια της βιβλιοθήκης pandas,όπου και θα δημιουργήσουμε ένα dataframe που θα το μετατρέψουμε σε ένα csv αρχείο. Αυτό επιτυγχάνεται με τις ακόλουθες γραμμές κώδικα:

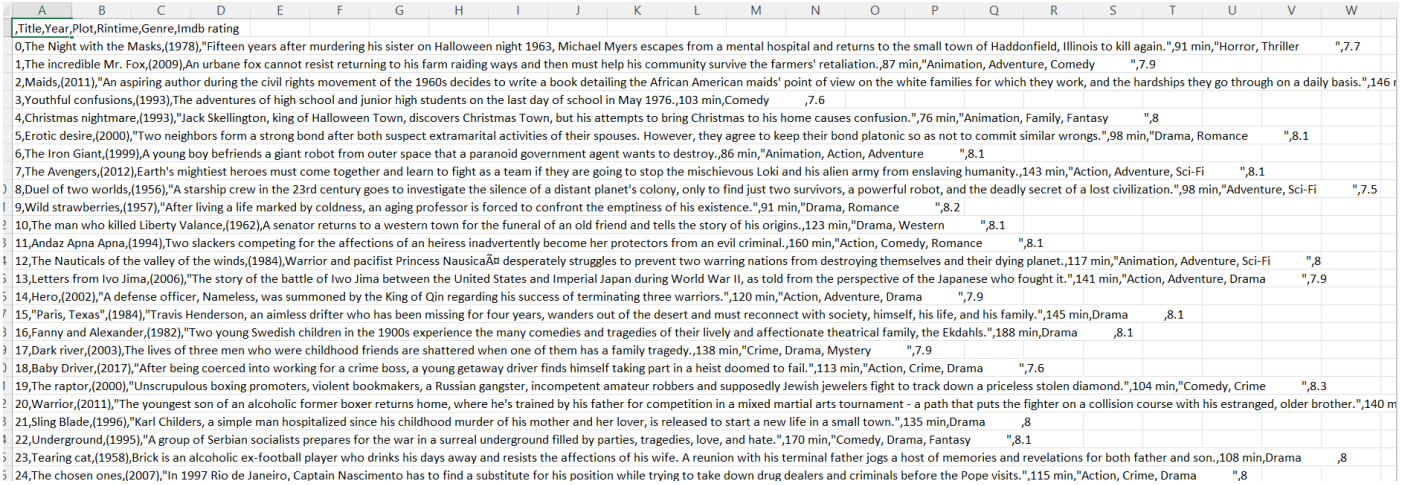
```
df = pd.DataFrame(fields_film,columns = ['Title','Year','Plot','Rintime','Genre','Imdb rating'])
df.to_csv('films_DataBase.csv')
```

Και μας τυπώνει:

```
C:\Users\tania\Desktop\University\Epiloghs\Information_Retreival>python scrape.py
                        Title    Year  ...                                  Genre Imdb rating
0                 Citizen Kane  (1941)  ...                  Drama, Mystery                 8.3
1                The Godfather  (1972)  ...                    Crime, Drama                 9.2
2              The Wizard of Oz  (1939)  ...     Adventure, Family, Fantasy                 8.1
3   Last release: Rita Hayworth  (1994)  ...                           Drama                 9.3
4                  Pulp Fiction  (1994)  ...                    Crime, Drama                 8.9
..                          ...     ...  ...                             ...             ...
95    Los Angeles: Confidential  (1997)  ...          Crime, Drama, Mystery                 8.3
96    Mad Max: The Road to Rage  (2015)  ...      Action, Adventure, Sci-Fi                 8.1
97                    Pinocchio  (1940)  ...  Animation, Adventure, Comedy                 7.5
98               The 39 stairs  (1935)  ...       Crime, Mystery, Thriller                 7.6
99         Rome, Fortified City  (1945)  ...           Drama, Thriller, War                   8

[100 rows x 6 columns]
```

Στιγμιότυπο από το csv αρχείο



,Title,Year,Plot,Rintime,Genre,Imdb rating
0,The Night with the Masks,(1978),"Fifteen years after murdering his sister on Halloween night 1963, Michael Myers escapes from a mental hospital and returns to the small town of Haddonfield, Illinois to kill again.",91 min,"Horror, Thriller ",7.7
1,The incredible Mr. Fox,(2009),An urbane fox cannot resist returning to his farm raiding ways and then must help his community survive the farmers' retaliation.,87 min,"Animation, Adventure, Comedy ",7.9
2,Maids,(2011),"An aspiring author during the civil rights movement of the 1960s decides to write a book detailing the African American maids' point of view on the white families for which they work, and the hardships they go through on a daily basis.",146 r
3,Youthful confusions,(1993),The adventures of high school and junior high students on the last day of school in May 1976.,103 min,Comedy ,7.6
4,Christmas nightmare,(1993),"Jack Skellington, king of Halloween Town, discovers Christmas Town, but his attempts to bring Christmas to his home causes confusion.",76 min,"Animation, Family, Fantasy ",8
5,Erotic desire,(2000),"Two neighbors form a strong bond after both suspect extramarital activities of their spouses. However, they agree to keep their bond platonic so as not to commit similar wrongs.",98 min,"Drama, Romance ",8.1
6,The Iron Giant,(1999),A young boy befriends a giant robot from outer space that a paranoid government agent wants to destroy.,86 min,"Animation, Action, Adventure ",8.1
7,The Avengers,(2012),Earth's mightiest heroes must come together and learn to fight as a team if they are going to stop the mischievous Loki and his alien army from enslaving humanity.,143 min,"Action, Adventure, Sci-Fi ",8.1
8,Duel of two worlds,(1956),"A starship crew in the 23rd century goes to investigate the silence of a distant planet's colony, only to find just two survivors, a powerful robot, and the deadly secret of a lost civilization.",98 min,"Adventure, Sci-Fi ",7.5
9,Wild strawberries,(1957),"After living a life marked by coldness, an aging professor is forced to confront the emptiness of his existence.",91 min,"Drama, Romance ",8.2
10,The man who killed Liberty Valance,(1962),A senator returns to a western town for the funeral of an old friend and tells the story of his origins.,123 min,"Drama, Western ",8.1
11,Andaz Apna Apna,(1994),Two slackers competing for the affections of an heiress inadvertently become her protectors from an evil criminal.,160 min,"Action, Comedy, Romance ",8.1
12,The Nauticals of the valley of the winds,(1984),Warrior and pacifist Princess NausicaÃ¤ desperately struggles to prevent two warring nations from destroying themselves and their dying planet.,117 min,"Animation, Adventure, Sci-Fi ",8
13,Letters from Ivo Jima,(2006),"The story of the battle of Iwo Jima between the United States and Imperial Japan during World War II, as told from the perspective of the Japanese who fought it.",141 min,"Action, Adventure, Drama ",7.9
14,Hero,(2002),"A defense officer, Nameless, was summoned by the King of Qin regarding his success of terminating three warriors.",120 min,"Action, Adventure, Drama ",7.9
15,"Paris, Texas",(1984),"Travis Henderson, an aimless drifter who has been missing for four years, wanders out of the desert and must reconnect with society, himself, his life, and his family.",145 min,Drama ,8.1
16,Fanny and Alexander,(1982),"Two young Swedish children in the 1900s experience the many comedies and tragedies of their lively and affectionate theatrical family, the Ekdahls.",188 min,Drama ,8.1
17,Dark river,(2003),The lives of three men who were childhood friends are shattered when one of them has a family tragedy.,138 min,"Crime, Drama, Mystery ",7.9
18,Baby Driver,(2017),"After being coerced into working for a crime boss, a young getaway driver finds himself taking part in a heist doomed to fail.",113 min,"Action, Crime, Drama ",7.6
19,The raptor,(2000),"Unscrupulous boxing promoters, violent bookmakers, a Russian gangster, incompetent amateur robbers and supposedly Jewish jewelers fight to track down a priceless stolen diamond.",104 min,"Comedy, Crime ",8.3
20,Warrior,(2011),"The youngest son of an alcoholic former boxer returns home, where he's trained by his father for competition in a mixed martial arts tournament - a path that puts the fighter on a collision course with his estranged, older brother.",140 m
21,Sling Blade,(1996),"Karl Childers, a simple man hospitalized since his childhood murder of his mother and her lover, is released to start a new life in a small town.",135 min,Drama ,8
22,Underground,(1995),"A group of Serbian socialists prepares for the war in a surreal underground filled by parties, tragedies, love, and hate.",170 min,"Comedy, Drama, Fantasy ",8.1
23,Tearing cat,(1958),Brick is an alcoholic ex-football player who drinks his days away and resists the affections of his wife. A reunion with his terminal father jogs a host of memories and revelations for both father and son.,108 min,Drama ,8
24,The chosen ones,(2007),"In 1997 Rio de Janeiro, Captain Nascimento has to find a substitute for his position while trying to take down drug dealers and criminals before the Pope visits.",115 min,"Action, Crime, Drama ",8

Περιγραφή του σχεδιασμού του συστήματος

Αρχικά ο στόχος του συστήματος που πρόκειται να σχεδιάσουμε είναι η ανάκτηση της πληροφορίας, η οποία θα είναι και η απάντηση στο ερώτημα κάποιου χρήστη. Με τον όρο ανάκτηση πληροφορίας εννοούμε την ανάκτηση εγγράφων που περιέχουν πληροφορία, η οποία χρειάζεται να είναι όσο πιο συναφής δύναται με την ανάγκη πληροφόρησης του χρήστη, ώστε να ολοκληρωθεί το ερώτημα του.

Πιο συγκεκριμένα, εδώ κάνει την εμφάνιση της η Lucene μια open-source βιβλιοθήκη λογισμικού υλοποιημένη σε Java. Η Lucene σου επιτρέπει να προσθέσεις δυνατότητες αναζήτησης σε μια εφαρμογή, καθώς μπορεί να ευρετηριοποιησει και να αναζητά οποιασδήποτε μορφής δεδομένα, από τα οποία εξάγεις από κάποιο text. Είναι ένα χρήσιμο εργαλείο, αφού δεν την ενδιαφέρει η πηγή των δεδομένων, το format τους ή ακόμα και η γλώσσα τους, αρκεί να μπορεί να εξάγει κάποιο κείμενο από εκεί.

- **Indexing :** Η διαδικασία του indexing θα μας βοηθήσει να διαχειριστούμε μεγάλο αριθμό αρχείων, αλλά και μεγάλα σε μέγεθος, πιο γρηγορά από το αν γινόταν σειριακά η διαδικασία αναζήτησης της πληροφορίας. Το αποτέλεσμα αυτής της διαδικασίας ονομάζεται index, μια ειδικά διαμορφωμένη δομή δεδομένων αποθηκευμένη στο file system ως μια ομάδα από index files . Το indexing ακολουθεί τα παρακάτω βήματα.
    1) **Aquire content**: Η Lucene δεν παρέχει κάποια λειτουργική υποστήριξη στο συγκεκριμένο βήμα. Αντίθετα, υποστηρίζεται εξ' ολοκλήρου από την εφαρμογή ή από ξεχωριστά τμήματα του λογισμικού. Υπάρχουν ορισμένοι open source crawlers που μπορούν να φανούν χρήσιμοι στην υλοποίηση του βήματος αυτού, όπως Solr, Nutch, OpenNLPκλπ.

2) **Build Document**: Μετάφραση του content σε units/documents. Το κάθε document αποτελείται από διάφορα, ξεχωριστά fields(πεδία) που περιέχουν κάποια τιμή. Τα fields μπορούν να καταταχθούν σε δυο κατηγορίες με την πρώτη να είναι (είτε) indexes (είτε όχι). Σε αυτή την περίπτωση οι τιμές του δεν αναλύονται από τον Analyzer. Η δεύτερη κατηγορία περιέχει fields τα οποία θεωρούνται είτε stored είτε όχι και αυτά είναι τα πεδία που πρέπει να εμφανίζονται στον χρήστη αναλυτικά. Επομένως θα δημιουργήσουμε μια μέθοδο η οποία θα δημιουργεί document και θα προσθέτει σε αυτό τα καινούρια fields. Στην συγκεκριμένη μηχανή αναζήτησης, επέλεξα και τα πέντε πεδία να αναλύονται και να αποθηκεύονται. Επιπροσθέτως, δημιουργώ και αποθηκεύω τα πεδία `"Year"` και `"Imdb Rating"`,ως **`SortedDocValuesField`** και **`StoredField`** , ιδιότητα με την οποία, είναι δυνατόν να επιτευχθεί κατά την διάρκεια της αναζήτησης, η εμφάνιση των αποτελεσμάτων μια ερώτησης είτε ταξινομημένα ως προς την χρονολογία που κυκλοφόρησε η ταινία, είτε ως προς την βαθμολογία της εκάστοτε ταινίας.

```
public Document getDocument(String title,String year,String plot,String runtime,String
genre,String imdb_rating)
	{
		Document document = new Document();
		document.add(new TextField("Title",title,Field.Store.YES));
		document.add(new TextField("Year",year.toString(),Field.Store.YES));
		document.add(new TextField("Plot",plot,Field.Store.YES));
		document.add(new TextField("Runtime",runtime.toString(),Field.Store.YES));
		document.add(new TextField("Genre",genre,Field.Store.YES));
		document.add(new TextField("Imdb Rating",imdb_rating.toString(),Field.Store.YES));
		document.add(new SortedDocValuesField ("Year",new BytesRef(year)));
		document.add(new StoredField("Year",year));
		document.add(new SortedDocValuesField ("Imdb Rating",new BytesRef(imdb_rating)));
		document.add(new StoredField("Imdb Rating",imdb_rating));

		return document;
	}
```

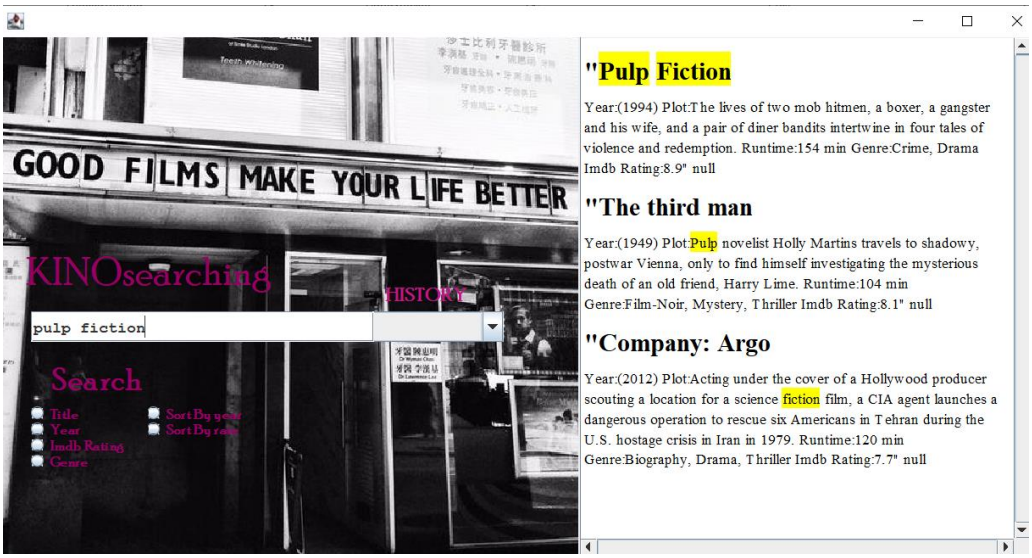
3) **Analyze Document and Index Document**: Η ευρετηριοποίηση ενός document δεν γίνεται αμέσως. Χρειάζεται πρώτα η διάσπαση του text σε μια σειρά από ανεξάρτητα μέρη, τα tokens. Το βήμα αυτό αποφασίζει πως θα χωριστούν τα textual fields σε ακολουθία από tokens. Πιο συγκεκριμένα έγινε η χρήση του StandardAnalyzer(), ο οποίος αφαιρεί τα stop words και μετατρέπει τα γράμματα του κάθε token σε μικρά. Για παράδειγμα, αν ο χρήστης δώσει μια ερώτηση με την μορφή “PuLp FicTion” ο StandardAnalyzer() θα το την αναλύσει και θα την μετατρέψει στην μορφή “pulp fiction”. Ακόμη, είναι η

απαραίτητη η δημιουργία του ευρετηρίου στην οποία θα αποθηκεύουμε τα documents. Αυτό πραγματοποιείται με την εντολή

```
dir = FSDirectory.open(Paths.get(indexDirectoryPath));
```

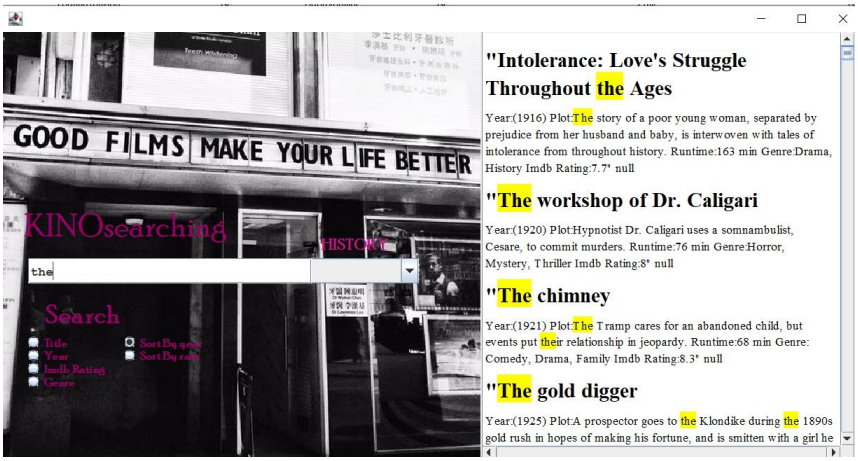
Τέλος, θα δημιουργήσουμε το αντικείμενο **IndexWriter writer** και με την μέθοδο **addDocument()** που μας παρέχει θα προσθέτουμε στον directory το document.

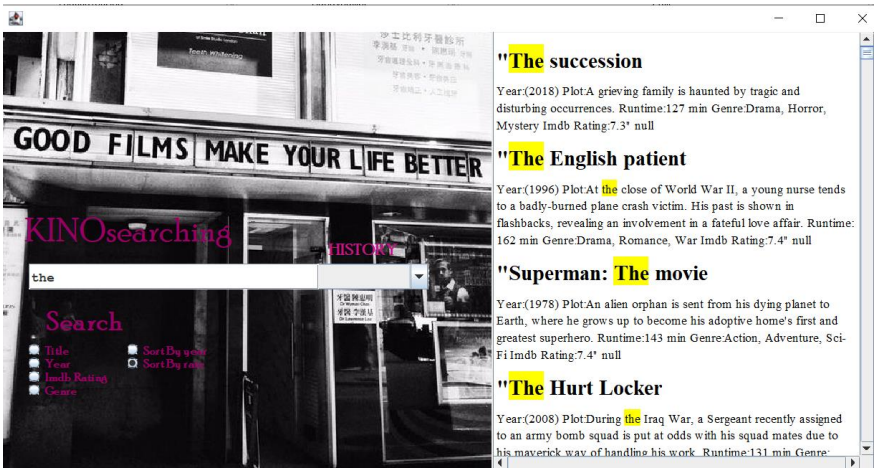
- **<u>Components of searching:</u>** **Searching** ονομάζεται η διαδικασία η οποία αναζητά μια λέξη μέσα στο ευρετήριο(index) και προσπαθεί να βρει τα documents που εμφανίζουν την συγκεκριμένη λέξη και αποτελείται από τα παρακάτω στοιχεία.
    1) **Search User Interface**: Το user interface είναι στην ουσία αυτό που βλέπει ο χρήστης στον web browser για παράδειγμα, όταν έρχεται σε επαφή με την μηχανή αναζήτησης. Είναι απαραίτητο να διατηρήσουμε την δομή του όσο πιο απλή γίνεται, το οποίο πρέπει να είναι πάντα ορατός τον χρήστη. Ακόμη η παρουσίαση των αποτελεσμάτων είναι πολύ σημαντική, όπως το να εμφανίζονται σε έντονη γραμματοσειρά οι επεξηγήσεις που αναφέρονται σε κάποια αστοχία σε σχέση με την αναζήτηση. Το user interface δημιουργήθηκε με την βοήθεια του **WindowBuilder**(), στο οποίο προστέθηκαν οι εξής λειτουργίες: Ένα κουμπί “Search”, με το οποίο ο χρήστης θα μπορεί να κάνει αναζήτηση, καθώς και επιπρόσθετα κουμπιά όπως “Title”, “Year”,”Genre”,”Imdb Rating”, τα οποία αν επιλεγούν η αναζήτηση της ερώτησης θα γίνει στα πεδία αυτά. Για παράδειγμα αν ο χρήστης κάνει την ερώτηση “Pulp Fiction” χωρίς την επιλογή “Title”, τα αποτελέσματα θα είναι αυτά.



Ενώ, αν επιλέξει το κουμπί “Title”, τα αποτελέσματα θα είναι αυτά.

Ακόμη, έχουν προστεθεί και οι επιλογές **“Sort By year”** και **“Sort By rate”**, οι οποίες εμφανίζουν τα αποτελέσματα του χρήστη ταξινομημένα είτε με την χρονολογική σειρά, είτε με την βαθμολογία τους κατά αύξουσα διάταξη.

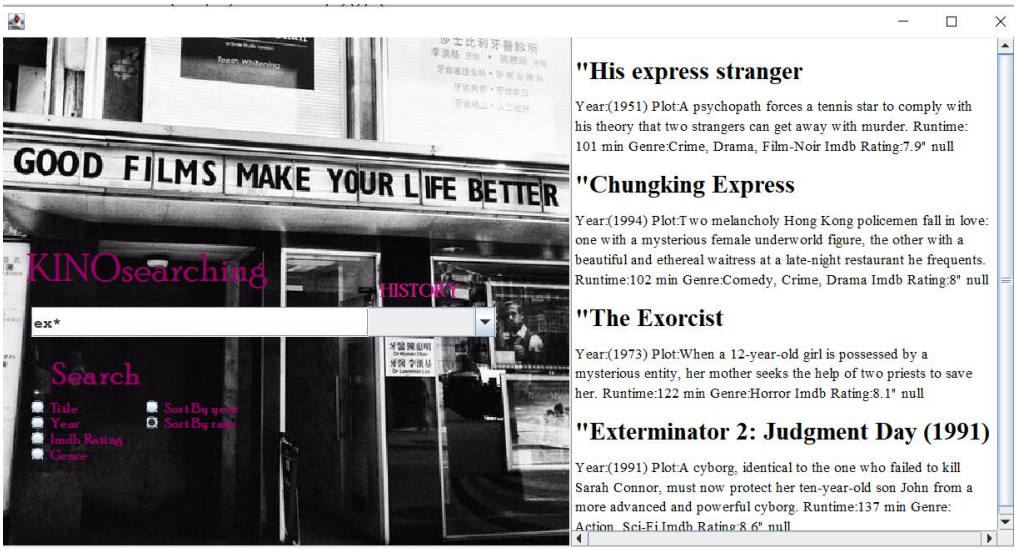




Ακόμη, έχει προστεθεί και το πεδίο History το οποίο διατηρεί, όλες τις προηγούμενες ερωτήσεις που έχει υποβάλει νωρίτερα ο χρήστης.

Επιπλέον, όπως φαίνεται και στα παραπάνω στιγμιότυπα έχει προστεθεί και η ιδιότητα της υπογράμμισης των λέξεων που περιέχονται στην ερώτηση του χρήστη. Αυτό υλοποιείται με την συνάρτηση **highlight(JEditorPane editorPane, String query)**, όπως φαίνεται και στον κώδικα.

Η τελευταίο προτέρημα της συγκεκριμένης μηχανής που προσφέρεται στον χρήστη είναι να αναζητήσει μια λέξη που περιέχεται σε κάποιο από τα πεδία σαν πρώτο ή δεύτερο συνθετικό της λέξης που δεν θυμάται.(Στο συγκεκριμένο στιγμιότυπο φαίνεται και η ταξινόμηση με βάση την βαθμολογία)



Το συγκεκριμένο στιγμιότυπο επιτεύχθηκε με την παρακάτω εντολή

`queryParser.setAllowLeadingWildcard(true)`😉

2) **Build Query**: Αφού ο χρήστης υποβάλλει την αίτηση αναζήτησης, συχνά ως αποτέλεσμα σε HTML μορφή ή ως Ajax αίτημα, υποβάλλεται από τον browser στον server μας. Γι' αυτό τον λόγο πρέπει να μεταφράσουμε το αίτημα σε ένα Query αντικείμενο της μηχανής αναζήτησης. Η Lucene παρέχει ένα εργαλείο , το οποίο ονομάζεται MultiFieldQueryParser και έχει την ικανότητα να παίρνει ως όρισμα πολλαπλά πεδία, δηλαδή τα πεδία που μας ενδιαφέρει να γίνει το searching.

```
if(field.length==0) {
		queryParser = new MultiFieldQueryParser(new
String[]{"Title","Year","Plot","Runtime","Genre","Imdb Rating","Drama"}
, analyzer);

}else {
		queryParser = new MultiFieldQueryParser(field, analyzer);
		}
```

Έπειτα η Query Parser οντότητα έχει μια parse() μέθοδο, την

public ***Query*** *parse* ***(String query)*** *throws* ***ParseException;***

Αν αποτύχει η ανάλυση της έκφρασης εμφανίζεται ένα ***ParseException***, το οποίο ουσιαστικά είναι ένα μήνυμα που εξηγεί τον λόγο που απέτυχε η ανάλυση. Ύστερα από την διαδικασία του **QueryParser**, τα αποτελέσματα περιέχονται στην δομή **Query**, τα οποία ταξινομούνται με την σειρά που αιτήθηκαν

3) **IndexSearcher**: Αρχικά δημιουργούμε ένα αντικείμενο **DirectoryReader reader**, με το οποίο θα διαβάσουμε το αρχείο ευρετηρίου που έχει δημιουργηθεί προηγουμένως μέσα στον directory. Έπειτα, σχηματίζουμε μια οντότητα **IndexSearcher writer = new IndexWriter(reader)**, η οποία ανοίγει το search index και στην συνέχεια με search μεθόδους παρουσιάζει τα αποτελέσματα της αναζήτησης. Τότε αυτό που επιστρέφεται είναι μια κλάση TopDocs που παρουσιάζει τα καλύτερα matches και το χρησιμοποιούμε για να παρουσιάσουμε τα αποτελέσματα στον χρήστη.
4) **TopDocs:** Όπως αναφέραμε και παραπάνω είναι μια κλάση που επιστρέφει τα μέγιστα hits. Το attribute **TopDocs.totalHit** επιστρέφει των αριθμό των documents που ταίριαξαν με την αναζήτηση του χρήστη ταξινομημένα σε φθίνουσα σειρά.
5) **ScoreDocs**: Το **TopDocs.scoreDocs** είναι ένας πίνακας που περιλαμβάνει τον απαιτούμενο αριθμό των καλύτερων matches. Κάθε **ScoreDocs** οντότητα διατηρεί ένα float score και ένα int doc, το ID του document, το οποίο μπορεί να χρησιμοποιηθεί για την ανάκτηση των αποθηκευμένων πεδίων του συγκεκριμένου document καλώντας **IndexSearcher.document(doc**). Τέλος, αν καλέσουμε **TopDocs.getMaxScore**(), θα μας επιστραφεί το καλύτερο score απ' όλα τα matches.

Όλη η συνάρτηση που πραγματοποιεί την διαδικασία του searching.

```
public List<String> searchIndex(String queryS, int maxHits,String[] field, String sortField) throws
ParseException, IOException {
		displayResults = new ArrayList();
		DirectoryReader reader = DirectoryReader.open(dir);
		searcher = new IndexSearcher(reader); //reader points to index folder
		if(field.length==0) {
			queryParser = new MultiFieldQueryParser(new String[]
			{"Title","Year","Plot","Runtime","Genre","Imdb Rating","Drama"} , analyzer);

		}else {
			queryParser = new MultiFieldQueryParser(field, analyzer);
		}
		TopDocs hits = null;
		//create a query object by parsing the search expression through QueryParse
		query = queryParser.parse(queryS);
		try {
			if(sortField=="") {
				hits = searcher.search(query, maxHits);
			}else if(sortField=="Year" || sortField=="Imdb Rating") {
				Sort sort = new Sort(new SortField(sortField, SortField.Type.STRING));
			hits = searcher.search(query, maxHits,sort);

		}
	//make the search which will return the TopDocs
	//TopDocs points to the top N search result which matches the search criteria
			for(ScoreDoc scoreDoc : hits.scoreDocs){
				Document doc = searcher.doc(scoreDoc.doc);
				displayResults.add("Title:"+doc.get("Title")+"\n");
				displayResults.add("Year:"+doc.get("Year")+"\n");
				displayResults.add("Plot:"+doc.get("Plot")+"\n");
				displayResults.add("Runtime:"+doc.get("Runtime")+"\n");
				displayResults.add("Genre:"+doc.get("Genre")+"\n");
				displayResults.add("Imdb Rating:"+doc.get("Imdb Rating")+"\n");
				displayResults.add(doc.get("Drama")+"\n");
			 }
	} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		reader.close();
		return displayResults;
```