Εργασία Ανάκτησης Πληροφορίας 2023-2024

Φοιτητές: *Αλέξανδρος Σαββίδης (ΑΕΜ: 3494)*

Σωτήριος Λουκάς Καμπύλης (ΑΕΜ: 3805)

Περίληψη Εργασίας (Εισαγωγή):

Στόχος της εργασίας είναι η υλοποίηση μιας web-based εφαρμογής, η οποία δίνει την δυνατότητα αναζήτησης για πληροφορίες ομιλιών από βουλευτές του Κοινοβουλίου. Τα δεδομένα έχουν συγκεντρωθεί από τον ιστότοπο της Βουλής, καθώς οι ομιλίες είναι διαθέσιμες στον οποιοδήποτε, όπου περιλαμβάνει ένα σύνολο δεδομένων με 1.280.918 ομιλίες βουλευτών της Ελληνικής Βουλής (βέβαια χρησιμοποιήθηκε μόνο ένα υποσύνολο των ομιλιών). Στόχος της εφαρμογής είναι η εύρεση των σημαντικότερων λέξεων-κλειδιών (ανά ομιλία, βουλευτή και κόμμα) και πως αυτές αλλάζουν με την πάροδο του χρόνου. Επίσης, πρέπει να μπορούμε να ανιχνεύσουμε ομοιότητες ανά ζεύγη μεταξύ των μελών του κοινοβουλίου και λαμβάνοντας υπόψιν ολές τις ομιλίες, χρησιμοποιώντας την τεχνική LSI, να βρίσκουμε τις σημαντικότερες θεματικές περιοχές και να εκφράζουμε κάθε ομιλία ως διάνυσμα σε κάποιο πολυδιάστατο χώρο. Η εργασία υλοποιήθηκε στην προγραμματιστική γλώσσα **Python** και για τον front-end κομμάτι της εφαρμογής χρησιμοποιήθηκε η τεχνολογία **Flask**. Έχουν υλοποιηθεί αρκετά Python files τα οποία έχουν το δικό τους ρόλο στην εφαρμογή όπως: της επεξεργασίας των δεδομένων, της ομοιότητας των ερωτημάτων αλλά και της υλοποίησης της LSI τεχνικής. Επίσης, χρειάζεται να υλοποιηθούν κάποια requirements για την εκτέλεση του προγράμματος όπως διευκρινίζεται παρακάτω.

Προυποθέσεις:

Βασική προυπόθεση για να εκτελεστεί το πρόγραμμα είναι να κατεβάσουμε κάποια συγκεκριμένα modules. Συγκεκριμένα, στο αρχείο requirements.txt υπάρχουν όλα τα απαραίτητα modules που χρειάζεται να κατεβάσουμε. Στο command line, και με την εντολή ‘pip install -r requirements.txt’ και εφόσον είναι επιτυχής το download, η εφαρμογή είναι έτοιμη για εκτέλεση. Η εφαρμογή εκτελείται από το αρχείο app.py και όταν είμαστε έτοιμοι μεταβαίνουμε στην ιστοσελίδα ‘http://127.0.0.1:5000/’.

Αρχεία Python

App

Αποτελεί το βασικό αρχείο της εφαρμογής όπου εδώ εκτελείται το πρόγραμμα.Αυτός ο κώδικας Python δημιουργεί μια Flask web εφαρμογή που υλοποιεί την αναζήτηση και την εμφάνιση πληροφοριών που σχετίζονται με ομιλίες στην βουλή, κόμματα και ομιλητές. Για την υλοποίση αυτή, κάνουμε import απαραίτητα modules όπως το Flask για web functionality, query για querying data , και άλλα custom modules (data_processing, initialize) για επεξεργασία δεδομένων και initialization.

Flask Routes:

1. /(Root Route): Χειρίζεται αιτήματα GET και POST. Το GET αποδίδει το index.html, ενώ το POST επεξεργάζεται το ερώτημα αναζήτησης που παρέχεται από τον χρήστη.
2. /result: Επεξεργάζεται το ερώτημα αναζήτησης και αποδίδει μια σελίδα αποτελεσμάτων που εμφανίζει σχετικές πληροφορίες.
3. /sitting, /speaker, /party: Αυτά τα routes εμφανίζουν πληροφορίες σχετικά με έναν συγκεκριμένο ομιλητή, πολιτικό κόμμα ή μία συνεδρίαση.

Query

Αυτό το αρχείο επικεντρώνεται στον χειρισμό ερωτημάτων και στην επιστροφή σχετικών πληροφοριών σχετικά με τις κοινοβουλευτικές διαδικασίες με βάση ομοιότητες του παρεχόμενου ερωτήματος.

Συναρτήσεις:

1. get_sittings(): Ανακτά τα 5 πιο παρόμοια έγγραφα (συνεδριάσεις) σε ένα δεδομένο query. Επιστρέφει λεπτομέρειες σχετικά με τις συνεδριάσεις, το όνομα του ομιλιτή, το πολιτικό κόμμα του ομιλητή και των πιο συχνών λέξεων που χρησιμοποιούνται στην ομιλία.
2. get_sitting_info(): Παρέχει πληροφορίες σχετικά με την συνεδρίαση που προσδιορίζεται από το ID της. Επιστρέφει λεπτομέρειες όπως το όνομα του ομιλιτή, το περιεχόμενο της ομιλίας και των πιο συχνών λέξεων που χρησιμοποιούνται στην ομιλία.
3. get_sittings_by_speaker(): Ανακτά όλες τις συνεδρίες από ένα συγκεκριμένο ηχείο. Επιστρέφει τα IDs, το πολιτικό κόμμα του ομιλητή για κάθε συνεδρίαση και τις πιο συχνές λέξεις που χρησιμοποιούνται στην ομιλία.
4. get_sittings_by_party(): Ανακτά 5 συνεδριάσεις από ένα συγκεκριμένο κόμμα, καθεμία από ένα διαφορετικό μέλος αυτού του κόμματος. Εμφανίζει τα IDs κάθε συνεδρίασης, το όνομα του ομιλητή για κάθε συνεδρίαση και τις πιο συχνές λέξεις (ετικέτες) που χρησιμοποιούνται στην ομιλία.

Data Processing

Αυτό το αρχείο εστιάζει στο preprocessing και handling κειμένου γραμμένο στην ελληνική γλώσσα.

Συναρτήσεις:

1. punctuation_removal(Data): Καταργεί τα σημεία στίξης και αντικαθιστά τα περισσότερα special characters στο κείμενο εισαγωγεί. Τέλος, επιστρέφει μία λίστα λέξεων από το επεξεργασμένο κείμενο.
2. Stop_word_removal(preprocessed_data,sop_words_array): Καταργεί τα stop words από μία λίστα προεπεξεργασμένων

λέξεων και επιστρέφει ένα string μετά την αφαίρεση των stop words.

3. Stemming(preprocessed_data): Κάνει stems τις λέξεις από τα preprocessed data χρησιμοποιώντας ένα greek stemmer. Επιστρέφει μία λίστα από stemmed words
4. preprocess(Data, stop_words_array): Καλεί τις συναρτήσεις stop words removal και punctuation removal εξίσου. Επιστρέφει ένα preprocessed string.
5. process(Data, stop_words_array): Χρησιμοποιεί τα παραπάνω functions για την προεπεξεργασία των input speech data. Επιστρέφει επεξεργασμένα data μαζί με τους πιο συνηθισμένους όρους στην ομιλία.

Συνολικά, το αρχείο αυτό περιέχει ένα σύνολο από functions αφιερωμένα στην προετοιμασία και τον ‘καθαρισμό΄ των δεδομένων κειμένου χρησιμοποιώντας την κατάργηση των stop words και εκτέλεση stemmer για περαιτέρω ανάλυση.

Initialize

Αυτό το αρχείο, υλοποιεί την ανάγνωση και την επεξεργασία αρχείων CSV που περιέχουν δεδομένα ομιλίας και σχετικά metadata.

Συναρτήσεις:

1.readCSV(): Διαβάζει ένα αρχείο CSV και φιλτράρει άσχετα δεδομένα, επιστρέφοντας ένα DataFrame χωρίς ομιλίες που σχετίζονται με ένα συγκεκριμένο πολιτικό κόμμα και μια λίστα με stop words από ένα separate file.

2.init(): Αρχικοποιεί την επεξεργασία δεδομένων καλώντας την readCSV(), στη συνέχεια επαναλαμβάνει τις ομιλίες, τις επεξεργάζεται και τις οργανώνει σε πολλά dictionaries και data structures για αποτελεσματική αναζήτηση.

Και οι δύο συναρτήσεις συνεργάζονται για να προετοιμάσουν και να οργανώσουν το σύνολο δεδομένων για επεξεργασία ερωτημάτων, συμπεριλαμβανομένου του φιλτραρίσματος περιττών δεδομένων και της εξαγωγής σχετικών πληροφοριών για ανάλυση και ανάκτηση.

KeyWord

Σκοπός του αρχείου είναι να εξάγει σχετικές key words που λέγονται από μέλη και κόμματα στο κοινοβούλιο, ταξινομώντας τις κατά ημερομηνία και δημιουργώντας ξεχωριστά αρχεία για μέλη και κόμματα.

Συναρτήσεις:

1. find_KeyWords(): Αυτή η συνάρτηση ανακτά τα speech data και τα stop words χρησιμοποιώντας συναρτήσεις από τα initialize και data_processing αρχεία. Στη συνέχεια επεξεργάζεται τις ομιλίες, δημιουργώντας δύο dictionaries (date_dict_member και date_dict_party) με βάση τις ημερομηνίες, τα μέλη και τα κόμματα.

Για κάθε ομιλία, ελέγχει τη διάρκεια της ομιλίας και επαληθεύει την ύπαρξη έγκυρων πληροφοριών ονόματος και κόμματος.

Συγκεντρώνει τα words που λέγονται από μέλη και κόμματα με βάση την ημερομηνία συνεδρίασης, σχηματίζοντας ένα dictionary structure.

Επαναλαμβάνεται μέσω των επεξεργασμένων δεδομένων και δημιουργεί δύο text files: το ένα περιέχει τους 15 πιο συχνούς όρους που εκφωνούνται από κάθε μέλος ταξινομημένο κατά την ημερομηνία συνεδρίασης και το άλλο περιέχει τους 15 πιο συχνούς όρους που εκφωνούνται από κάθε μέρος ταξινομημένο κατά την ημερομηνία συνεδρίασης.

LSI

Αυτό το αρχείο, υλοποιεί και εφαρμόζει την Latent Semantic Indexing (LSI) σε δεδομένα από κοινοβουλευτικές ομιλίες για εξαγωγη θεμάτων.

Initialization: Διαβάζει ένα CSV αρχείο και προετοιμάζει τα δεδομένα ομιλίας για επεξεργασία, αφαιρώντας τα stop words και ενδιάμεσες λέξεις.

LSI Implementation: Χρησιμοποιώντας το TfidfVetorizer και το TruncatedSVD (LSI) από τη βιβλιοθήκη scikit-learn:

Μετατρέπει τις επεξεργασμένες ομιλίες σε διανύσματα TF-IDF.

Εκτελεί LSI (TruncatedSVD) για να μειώσει τις διαστάσεις και να εξάγει topics. Η παράμετρος n_components καθορίζει τον αριθμό των topics που εξάγονται.

Στη συνέχεια εκτυπώνονται τα θέματα μαζί με τους πέντε πιο σχετικούς όρους για κάθε topic. Τέλος, γράφει το dominant topic για κάθε document σε ένα file με το όνομα doc_topics.txt.

Apriori Rules

Αυτό το αρχείο, δημιουργεί κανόνες συσχέτισης (assocation rules) με βάση τις λέξεις που χρησιμοποιούνται στις ομιλίες στην Βουλή.

Functionality: Δημιουργεί association rules χρησιμοποιώντας τη μέθοδο Apriori.

Παράμετροι input:

1. start_year: Δεδομένα από φέτος και μετά.
2. end_year: Δεδομένα από φέτος και πριν.
3. pref_speaker: Δεδομένα μόνο από τις ομιλίες αυτού του μέλους.
4. pref_party: Δεδομένα μόνο από τις ομιλίες αυτού του κόμματος.
5. pref_word: Θα σχηματιστούν rules που περιέχουν αυτήν τη λέξη.
6. user_useless_tags: Λέξεις που δεν θα χρησιμοποιηθούν στα rules.
7. minimal_support: Ελάχιστη υποστήριξη που πρέπει να έχουν οι rules.

Implementation: Διαβάζει και επεξεργάζεται δεδομένα μίας ομιλίας με βάση κριτήρια που καθορίζονται από τον χρήστη. Εξάγει 50 πιο συχνές λέξεις από κάθε ομιλία, εξαιρουμένων ορισμένων προκαθορισμένων useless tags και καθορισμένων από τον χρήστη "user_useless_tags". Δημιουργεί ένα dataframe όπου κάθε σειρά αντιπροσωπεύει μια ομιλία και οι στήλες αντιπροσωπεύουν την παρουσία/απουσία των 50 πιο συχνών λέξεων. Χρησιμοποιεί τον αλγόριθμο Apriori για τη δημιουργία association rules με βάση τις εμφανίσεις λέξεων. Γράφει τα association σε ένα αρχείο με το όνομα Rules.txt.

Query Similarity

Αυτό το αρχείο, υλοποιεί μία συνάρτηση υπολογισμού της ομοιότητας document-query χρησιμοποιώντας TF-IDF. Λαμβάνει ένα dictionary words, που αντιπροσωπεύει τη συχνότητα όρων σε documents και queries, και επιστρέφει τις πέντε πιο σχετικές ομιλίες μαζί με την αντίστοιχη βαθμολογία ομοιότητάς τους. Προσδιορίζει documents σχετικά με το query ελέγχοντας εάν κάποια λέξη στο ερώτημα υπάρχει στο dictionary words. Υπολογίζει τις τιμές TF-IDF για το ερώτημα και τα έγγραφα χρησιμοποιώντας τον τύπο που παρέχεται. Υπολογίζει την ομοιότητα συνημιτόνου μεταξύ του ερωτήματος και κάθε εγγράφου με βάση τις αναπαραστάσεις του TF-IDF. Ταξινομεί τα έγγραφα με βάση τις βαθμολογίες ομοιότητάς τους και επιστρέφει τις πέντε πρώτες πιο σχετικές ομιλίες.

Top k Similar

Αυτό το αρχείο, βρίσκει τους top-k similar ομιλητές με βάση τις ομιλίες τους. Διαβάζει speech data και προετοιμάζει dictionaries για την αποθήκευση ονομάτων μελών, κομμάτων και εγγράφων. Επεξεργάζεται τις ομιλίες, δημιουργώντας dictionaries για μέλη, κόμματα και έγγραφα ενώ συνδυάζει ομιλίες από το ίδιο μέλος. Υπολογίζει τα διανύσματα TF-IDF για τα έγγραφα και υπολογίζει τον πίνακα ομοιότητας συνημιτόνου με βάση αυτά τα διανύσματα. Καθορίζει τα top k similar ζεύγη ομιλητών με βάση τις βαθμολογίες ομοιότητας συνημιτόνου τους. Γράφει τα top k similar ζεύγη ομιλητών μαζί με τους βαθμούς ομοιότητάς τους σε ένα file.

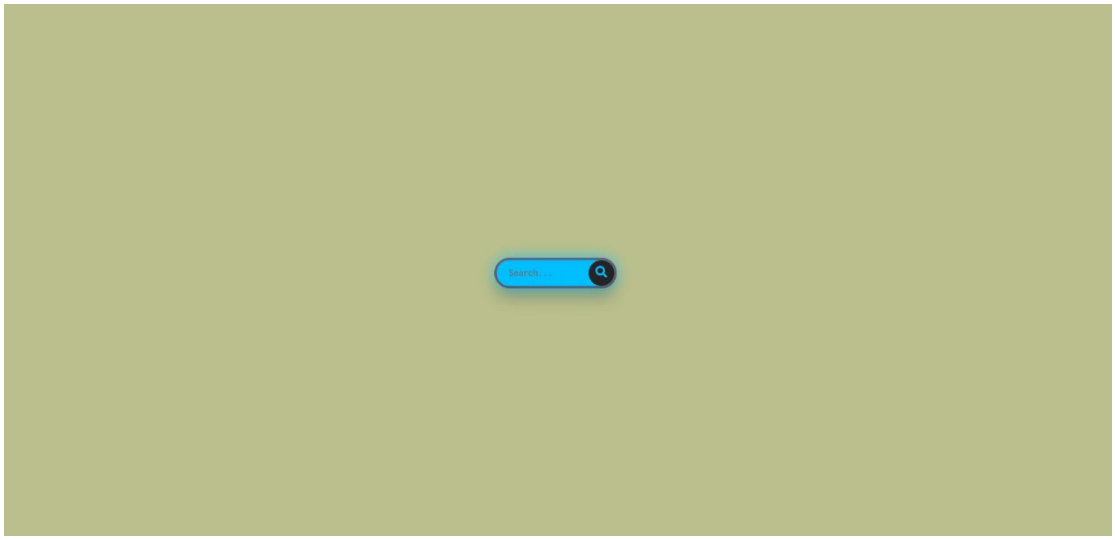
ΕΝΔΕΙΚΤΙΚΑ ΠΑΡΑΔΕΙΓΜΑΤΑ ΤΡΟΠΟΥ ΛΕΙΤΟΥΡΓΙΑΣ

Παρακάτω παρουσιάζουμε μερικά παραδείγματα (προφανώς όχι όλα γιατί υπάρχουν πολλές διαφορετικές περιπτώσεις) για το πως να ξεκινήσετε την εργασία.

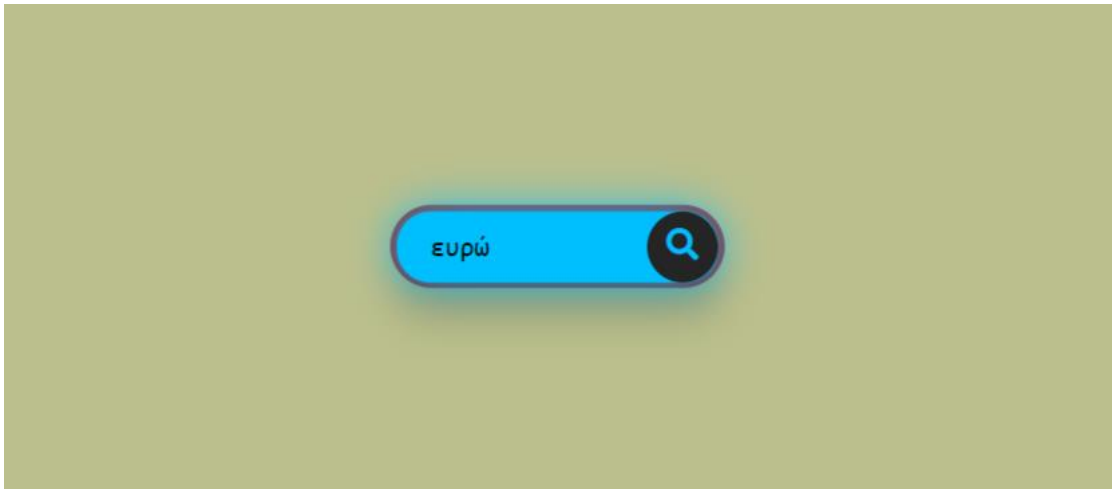
Αρχικά, κάνουμε κλικ στον υπερσύνδεσμο που βλέπουμε στην εικόνα παρακάτω έτσι ώστε να μπορούμε να ανοίξουμε την εφαρμογή μας.

```
 * Serving Flask app 'app.py'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

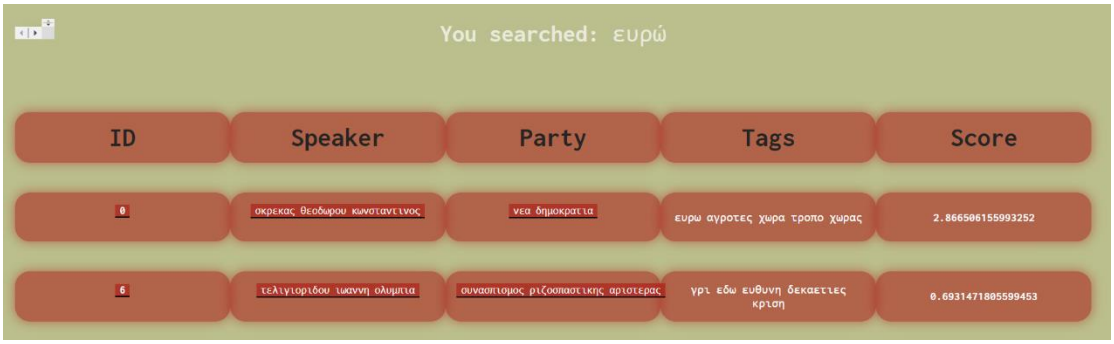
Έπειτα, βλέπουμε ένα πεδίο αναζήτησης (το οποίο έχει διαμορφωθεί κατάλληλα χρωματικά κτλ. με την βοήθεια της *HTML/CSS*). Μέσα εκεί δοκιμάζουμε να πληκτρολογήσουμε κατάλληλες [***ΠΡΟΣΟΧΗ***: Σε λέξεις οι οποίες δν πληρούν τις προυποθέσεις της εκφώνησης (π.χ. αγγλικά), πετάει *Internal Server Error* που είναι και η αναμενόμενη συμπεριφορά!].



Ας δοκιμάσουμε την λέξη: *“ευρώ”*, όπως φαίνεται και στην παρακάτω εικόνα.



Ύστερα, θα παρατηρήσουμε (σύμφωνα με το csv που έχουμε),τις ομιλίες των υπουργών που εμφανίζεται η συγκεκριμένη λέξη!

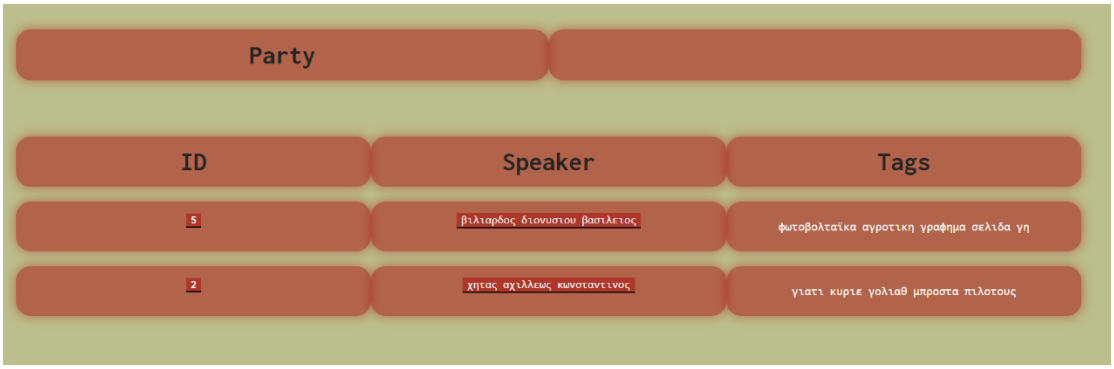


Μάλιστα, το Score αναφέρεται στην ομοιότητα μεταξύ των ομιλιών. Ακόμα, άμα θέλουμε μπορούμε να πλοηγηθούμε μέσα στο περιβάλλον της εφαρμογής κάνοντας κλικ στους υπερσυνδέσμους που έχουμε φτιάξει (στα ID, Speaker & Party). Για παράδειγμα, πατώντας στο Party = "*νεα δημοκρατια*" θα δούμε όλους τους υπουργούς (μέσα στο csv μας) της Νέας Δημοκρατίας!

Το ίδιο μπορούμε να κάνουμε με οποιαδήποτε ***ΣΩΣΤΗ*** λέξη μέσα από τις ομιλίες έτσι ώστε να μπορέσουμε να κάνουμε την αναζήτηση μας (απλώς δώσαμε ένα ενδεικτικό παράδειγμα)! Μάλιστα, λέξη θεωρείται οτιδήποτε αναφέρεται ως keyword στην εκφώνηση.

2. Ανά ομιλία, ανά βουλευτή και ανά κόμμα, θα πρέπει να βρούμε τις σημαντικότερες λέξεις-κλειδιά (keywords) και πως αυτές αλλάζουν στο χρόνο.

Ένα άλλο παράδειγμα (με βάση το κόμμα) είναι αν ψάξουμε βουλευτές του κόμματος: "ελληνικη λυση - κυριακος βελοπουλος" θα μας βγάλει τους παρακάτω 2 βουλευτές.



Μάλιστα,πατώντας το ID βγάζει όλη την ομιλία του εκάστοτε βουλευτή (μαζί με χρήσιμες πληροφορίες για αυτήν π.χ. ημερομηνία κτλ.).