View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by University of Thessaly Institutional Repository

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ,
ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

Σύστημα Παρακολούθησης Ακαδημαϊκών Δημοσιεύσεων

Academic Publications Tracker

Διπλωματική Εργασία

Παύλος Κάλλης

Επιβλέποντες Καθηγητές: Δημήτριος Κατσαρός
Λέκτορας Π.Θ. ΤΜΗΥΤΔ

Βάβαλης Εμμανουήλ
Καθηγητής Π.Θ. ΤΜΗΥΤΔ

ΒΟΛΟΣ 2013

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Ευχαριστίες

Με την περάτωση της παρούσας Διπλωματικής Εργασίας, θα ήθελα να ευχαριστήσω τον επιβλέποντα μου κ. Σπύρο Λάλη για την καθοδήγηση που μου παρείχε κατά τη διάρκεια εκπόνησής της καθώς και για την γενικότερη εμπιστοσύνη που έχει επιδείξει στο πρόσωπό μου.

Επίσης, ευχαριστώ τους φίλους και συμφοιτητές μου για την αμέριστη υποστήριξή τους και για όλες τις αξέχαστες στιγμές που περάσαμε μαζί κατά τη διάρκεια των σπουδών μου.

Τέλος, οφείλω ένα μεγάλο ευχαριστώ στην οικογένειά μου Γιώργο, Παρασκευή και Ελένη που βρίσκονται πάντα δίπλα μου όλα αυτά τα χρόνια και με στηρίζουν σε κάθε μου απόφαση.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Περίληψη

Αντικείμενο της παρούσας διπλωματικής είναι η υλοποίηση ενός συστήματος παρακολούθησης ακαδημαϊκών δημοσιεύσεων, που βρίσκονται σε ψηφιακές βιβλιοθήκες. Το σύστημα ειδοποιεί τους χρήστες όταν εντοπίσει νέες δημοσιεύσεις και προβάλλει στατιστικά για αυτές. Αρχικά, αναπτύχθηκε ένα σύστημα εντοπισμού δημοσιεύσεων, το οποίο χρησιμοποιώντας τεχνικές crawling αναζητά δημοσιεύσεις σε ψηφιακές βιβλιοθήκες(dblp, ieeexplore), με βάση κριτήρια που εισάγει ο χρήστης. Το συγκεκριμένο σύστημα είναι εύκολα επεκτάσιμο, προσφέροντας τη δυνατότητα σε τρίτους, να γράψουν το δικό τους crawler module για άλλες ψηφιακές βιβλιοθήκες. Επιπλέον, αναπτύχθηκε ένα σύστημα ειδοποίησης, το οποίο δίνει τη δυνατότητα στο χρήστη να ενημερώνεται για νέες δημοσιεύσεις μέσω skype και e-mail. Τέλος, αναπτύχθηκε ένα γραφικό περιβάλλον, το οποίο προβάλλει στατιστικά σχετικά με τις δημοσιεύσεις και είναι παραμετροποιήσιμο με βάση τις ανάγκες του χρήστη.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Abstract

The subject of the present thesis is the implementation of an academic publications tracker system. The system searches for publications, notifies the users about them and shows relevant statistics. At first, a publication detection system was implemented, which based on predefined user filters uses crawling techniques to search for publications in digital libraries(dblp, ieeeexplore). Its crawling logic can be easily extended by writing custom crawlers for other digital libraries. Furthermore, a notification system was implemented, which gives the ability to the user to be notified for new publications through skype and e-mail. Finally, a graphical user interface was developed, which displays statistics about the publications and is easily configurable by the user.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Περιεχόμενα

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 1 Εισαγωγή

1.1 Πλαίσιο της Διπλωματικής Εργασίας

Για κάθε ερευνητικό ίδρυμα είναι σημαντικό να καταγράφονται οι ερευνητικές του επιδόσεις, καθώς και οι επιδόσεις του κάθε ερευνητή που είναι μέλος αυτού. Με την καταγραφή και παρακολούθηση των επιδόσεων κάθε ιδρύματος, μπορεί να αξιολογηθεί η ποιότητα και η ποσότητα της παρεχόμενης έρευνας και να βγουν χρήσιμα συμπεράσματα. Επιπλέον, σε ένα ακαδημαϊκό ίδρυμα όπως είναι το πανεπιστήμιο, παρουσιάζεται η ανάγκη ενημέρωσης των μελών του για το ερευνητικό έργο που επιτελείται. Για τους σκοπούς της παρούσας διπλωματικής, επιλέχθηκε η ενημέρωση των μελών του μέσω μια οθόνης, η οποία θα μπορεί να εμφανίζει στατιστικά για το ερευνητικό έργο κάθε τμήματος και των μελών του. Η οθόνη αυτή θα μπορούσε να αναρτηθεί σε κάποιο πίνακα ανακοινώσεων του τμήματος, έτσι ώστε να είναι εύκολα προσβάσιμη. Επιπρόσθετα, για να μπορούν τα μέλη να ενημερώνονται για νέες δημοσιεύσεις, θα ήταν χρήσιμη η ενημέρωση μέσω e-mail ή μηνυμάτων skype κάθε φορά που εντοπίζεται μια νέα δημοσίευση από το τμήμα. Για τους σκοπούς αυτούς, επιλέχθηκε να υλοποιηθεί ένα σύστημα το οποίο θα υλοποιεί τη λειτουργικότητα που περιγράφτηκε προηγουμένως.

1.2 Αντίστοιχες εργασίες

Στην παρούσα διπλωματική εστιάζουμε σε ένα σύστημα που θα παρακολουθεί ακαδημαϊκές δημοσιεύσεις για ένα τμήμα. Βρήκαμε αντίστοιχες εφαρμογές που λύνουν τα ίδια προβλήματα. Οι κυριότερες από αυτές, είναι το Google Scholar και το Microsoft Academic Search. Και τα δύο συστήματα αποτελούν ολοκληρωμένες λύσεις παρακολούθησης ερευνητών και δημοσιεύσεων, δεν είναι όμως εύκολα προσαρμόσιμα για τις ανάγκες ενός πανεπιστημίου. Ακολουθεί μια σύντομη περιγραφή των χαρακτηριστικών της κάθε εφαρμογής.

1.2.1 Google Scholar

Το **Google Scholar** είναι ένας ολοκληρωμένος ιστότοπος παρακολούθησης ακαδημαϊκών δημοσιεύσεων, που προσφέρει μία ενσωματωμένη μηχανή αναζήτησης. Κάθε ερευνητής μπορεί να δημιουργήσει ένα προφίλ, προσθέτοντας τα προσωπικά του στοιχεία, τα ερευνητικά του ενδιαφέροντα καθώς και το ίδρυμα με το οποίο συνεργάζεται. Έπειτα, μπορεί να παρακολουθεί τις δημοσιεύσεις για τις οποίες γίνονται ετεροαναφορές προς στο όνομα του, λαμβάνοντας ενημερώσεις ή βλέποντας τα αποτελέσματα στην εφαρμογή.

Το δεύτερο χαρακτηριστικό της εφαρμογής είναι ότι, ο χρήστης επιπλέον μπορεί να λαμβάνεις ενημερώσεις μέσω e-mail για ένα ερώτημα που θα κάνει στη μηχανή αναζήτησης. Το ερώτημα αυτό μπορεί να έχει γενική μορφή, να είναι δηλαδή μια λέξη κλειδί, ένας συγγραφέας, ένα τμήμα ή οτιδήποτε άλλο επιθυμεί ο χρήστης.

Το τρίτο χαρακτηριστικό που προσφέρει το Google Scholar είναι η προβολή

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

στατιστικών για συγγραφείς και δημοσιεύσεις. Ο χρήστης μπορεί να δει τους κορυφαίους συγγραφείς για ένα πανεπιστήμιο, ένα ερευνητικό τομέα, μια χώρα ή ένα τμήμα με βάση κάποιο αλγόριθμο κατάταξης, που συνδυάζει διάφορες παραμέτρους όπως είναι ο αριθμός των ετεροαναφορών και η σχετικότητα των αποτελεσμάτων. Επιπλέον, για κάθε ερευνητή και για κάθε δημοσίευση μπορεί να δει στατιστικά ανά έτος, όπως είναι ο δείκτης [h – index].

1.2.2 Microsoft Academic Search

Το **Microsoft Academic Search** είναι κι αυτό μια μηχανή αναζήτησης ερευνητών και δημοσιεύσεων από τον ακαδημαϊκό χώρο. Ο χρήστης μπορεί να κάνει αναζήτηση με βάση τον ερευνητή, το πανεπιστήμιο, τη δημοσίευση και τον ερευνητικό τομέα. Τα αποτελέσματα της αναζήτησης εμφανίζονται σε ένα γραφικό περιβάλλον, το οποίο παρέχει τη δυνατότητα εμφάνισης στατιστικών στο χρήστη όπως είναι ο αριθμός των ετεροαναφορών και ο h-index. Επιπλέον, ο χρήστης μπορεί να ζητήσει να ενημερώνεται μέσω e-mail για νέες δημοσιεύσεις που αφορούν κάποιο συγγραφέα, κάποιο πανεπιστήμιο ή ερευνητικό τομέα.

Η λειτουργικότητα του είναι παρόμοια με αυτή του Google Scholar, παρέχει όμως επιπρόσθετες δυνατότητες στο χρήστη, όπως είναι η παρακολούθηση τάσεων σε ένα συγκεκριμένο ερευνητικό τομέα, η γεωγραφική προβολή ιδρυμάτων στο χάρτη και η οπτική απεικόνιση των αλληλεπιδράσεων μεταξύ των συγγραφέων. Με την παρακολούθηση τάσεων σε έναν ερευνητικό τομέα, ο χρήστης μπορεί να δει ποιος τομέας συγκεντρώνει το μεγαλύτερο ερευνητικό ενδιαφέρον, ενώ στην οθόνη απεικόνισης αλληλεπιδράσεων μεταξύ των ερευνητών εμφανίζονται ποιοι ερευνητές έχουν συνεργαστεί στα πλαίσια συγγραφής ενός ερευνητικού άρθρου.

1.3 Σκοπός της διπλωματικής Εργασίας

Τα παραπάνω συστήματα παρέχουν αρκετές δυνατότητες και απευθύνονται σε ερευνητές, αλλά και σε απλούς χρήστες που θέλουν να παρακολουθήσουν τα τεκταινόμενα στον ακαδημαϊκό χώρο. Για τις ανάγκες ενός ερευνητικού ιδρύματος, τα παραπάνω συστήματα δεν αποτελούν ικανοποιητική λύση, λόγω της γενικής μη παραμετροποιήσιμης λειτουργικότητας που παρέχουν.

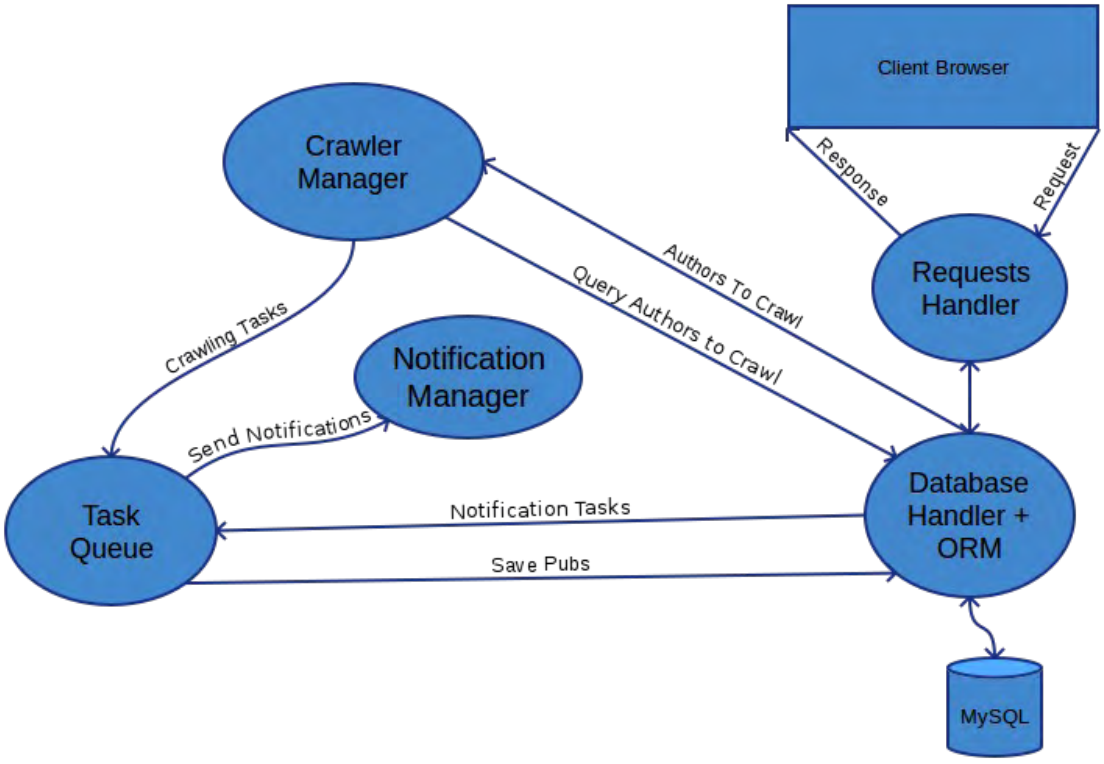
Εστιάζοντας στις ανάγκες ενός ερευνητικού ιδρύματος και των μελών του, επιλέχθηκε να υλοποιηθεί ένα σύστημα, το οποίο θα προβάλλει τα στατιστικά για το ερευνητικό έργο ενός πανεπιστημίου σε μία οθόνη, η οποία μπορεί να αναρτηθεί στον πίνακα ανακοινώσεων του εκάστοτε τμήματος. Επιπρόσθετα, θα ενημερώνει όσους ενδιαφέρονται μέσω e-mail και skype για νέες δημοσιεύσεις από μέλη του τμήματος. Τέλος, το σύστημα θα είναι εύκολα επεκτάσιμο, θα μπορεί δηλαδή οποιοσδήποτε να γράψει δικές του βιβλιοθήκες που θα επεκτείνουν την υπάρχουσα λειτουργικότητα.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 2 Αρχιτεκτονική Συστήματος

Η αρχιτεκτονική του συστήματος καθορίστηκε με βάση τις απαιτήσεις κλιμακωσιμότητας, εύκολης παραμετροποίησης και αλλαγών στο κώδικα. Επιπλέον, στοχεύουμε σε εύκολη επεκτασιμότητα όσον αφορά τη διαδικασία του crawling και τη διαδικασία των ειδοποιήσεων. Θέλουμε δηλαδή να μπορεί κάποιος τρίτος να γράφει εύκολα τους δικούς του crawlers που θα αναζητούν δημοσιεύσεις και σε άλλες βιβλιοθήκες εκτός από το DBLP και το IEEEXPLORE. Το ίδιο ισχύει και για το σύστημα ειδοποιήσεων. Κάθε χρήστης θα μπορεί να επεκτείνει το υπάρχον σύστημα ειδοποίησης προσθέτοντας δικούς του ειδοποιητές πέρα από τις ειδοποιήσεις μέσω skype και e-mail.

Για την καλύτερη οργάνωση της εφαρμογής μας και για την ευκολότερη συντήρησή της, επιλέξαμε να χρησιμοποιήσουμε την αρχιτεκτονική MTV(model-template-view). Με αυτόν τον τρόπο, διαχωρίζουμε τη λογική παρουσίασης των δεδομένων(template), από τη λογική επικοινωνίας με τη βάση δεδομένων(model) και τη λογική επικοινωνίας μεταξύ του client και του server(view). Επιπλέον, επειδή αρκετές από τις εργασίες μπορούν να εκτελεστούν στο παρασκήνιο, επιλέξαμε να επεκτείνουμε την υπάρχουσα αρχιτεκτονική χρησιμοποιώντας μια ουρά εργασιών που ονομάζεται celery, η οποία συντονίζει όλες τις εργασίες που γίνονται στο παρασκήνιο. Στην επόμενη εικόνα φαίνεται μία υψηλού επιπέδου απεικόνιση της αρχιτεκτονικής.



Εικ 1: Υψηλού επιπέδου απεικόνιση της αρχιτεκτονικής του συστήματος

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Όπως βλέπουμε και στην εικόνα, το σύστημα αποτελείται από πέντε υποσυστήματα. Ακολουθεί μία συνοπτική περιγραφή του καθενός.

Στο υποσύστημα Requests Handler βρίσκεται όλη η λογική που αφορά την επικοινωνία του client με τον server καθώς και η λογική που αφορά την επικοινωνία του Requests handler με τον Database Handler. Το υποσύστημα Requests Handler επεξεργάζεται τα requests που φτάνουν στο server και ανάλογα με το περιεχόμενο του request, μπορεί να επικοινωνήσει με το υποσύστημα DB Handler για να τροποποιήσει ή να διαβάσει πίνακες τη βάσης δεδομένων. Το υποσύστημα Requests Handler επιστρέφει ένα response στον browser φορτώνοντας ένα template, το οποίο περιέχει όλη τη λογική παρουσίασης των δεδομένων στο χρήστη.

Στο υποσύστημα Database Handler βρίσκεται η λογική που αφορά τη βάση δεδομένων καθώς και την επικοινωνία με τη βάση δεδομένων. Το υποσύστημα Database Handler απαρτίζουν ο DB handler και ο Object Relational Mapper(ORM).

Στο υποσύστημα Task Queue βρίσκονται όλες οι εργασίες οι οποίες εκτελούνται στο παρασκήνιο, όπως είναι οι εργασίες crawling και οι εργασίες ειδοποίησης για νέες δημοσιεύσεις.

Στο υποσύστημα Crawler Manager βρίσκεται όλη η λογική του crawling για τις δημοσιεύσεις. Εδώ περιλαμβάνεται όλη η λογική για το πως και πότε θα γίνει το crawling από τους custom crawlers.

Στο υποσύστημα Notification Manager βρίσκεται όλη η λογική της ειδοποίησης για νέες δημοσιεύσεις. Ειδικότερα, σε αυτό το σύστημα αποφασίζεται πόσο συχνά και για ποιες δημοσιεύσεις θα ενημερώνονται οι χρήστες του συστήματος.

Τα παραπάνω υποσυστήματα περιγράφονται αναλυτικότερα στα κεφάλαια που ακολουθούν.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 3 Requests Handler

Το υποσύστημα Requests Handler είναι υπεύθυνο για την επικοινωνία μεταξύ του client και του server. Ο ρόλος του είναι να λαμβάνει HTTP requests από τον client, να επεξεργάζεται τα αιτήματα αυτά και να επιστρέφει responses στον client. Το υποσύστημα Requests Handler επικοινωνεί με το υποσύστημα model σε περιπτώσεις που χρειάζεται να τροποποιηθεί ή να διαβαστεί κάποιο μοντέλο της βάσης δεδομένων. Επιπλέον, κάθε φορά που πρέπει να σταλεί ένα response στον client, το υποσύστημα Requests Handler χρησιμοποιεί ένα template engine(jinja2) για να μετατρέψει τον κώδικα python σε html και να τον επιστρέψει στον client. Ο λόγος που χρησιμοποιείται ένα template engine είναι για να περιοριστεί ο διπλότυπος html κώδικας που γράφεται από τον προγραμματιστή, χρησιμοποιώντας έτοιμα templates. Ακολουθεί μια περιγραφή της τυπικής λειτουργίας ενός κύκλου request-response, από τη στιγμή που φτάνει ένα request στο υποσύστημα Requests Handler, μέχρι τη στιγμή που επιστρέφεται ένα response στον client. Για παράδειγμα, αν ο Requests handler λάβει το εξής request :

```
GET / HTTP 1.1
```

Χρησιμοποιώντας το flask έχουμε αντιστοιχήσει το root url(/) σε μια συνάρτηση, που αναλαμβάνει να φορτώσει ένα template(που περιέχει κώδικα python και html) από το φάκελο templates. έπειτα να το μετατρέψει σε html χρησιμοποιώντας μια template engine(jinja2) και να το επιστρέψει στο χρήστη.

Η αντιστοίχηση url σε συναρτήσεις και η φόρτωση ενός template γίνεται σε python με τον εξής τρόπο :

```
@app.route('/')
def index():
 return render_template(“index.html”)
```

Το template στην περίπτωση μας περιλαμβάνει μόνο εντολές html αλλά θα μπορούσε να περιλαμβάνει και μεταβλητές python αναμεμειγμένες με κώδικα html. Στην περίπτωση αυτή, θα έπρεπε να μετατραπούν οι μεταβλητές σε εντολές html χρησιμοποιώντας το jinja2. Τελικά, η html σελίδα επιστρέφεται σαν response στο χρήστη. Για την καλύτερη κατανόηση των templates ακολουθεί ένα παράδειγμα ενός template :

```
<!doctype html>
<html>
 <head>
  {% block head %}
   <title>{% block title %} Default title {% endblock %}</title>
  {% endblock %}
 </head>
 {%block body %}
            <div class="navbar">
              <ul class="nav nav-tabs">
```


Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

```
                {% if g.user.is_authenticated() %}
                  <li><a href="/subscriptions/new"> Subscriptions </a></li>
                {% else %}
                  <li><a href="/login"> Login </a></li>
                {% endif %}
              </ul>
            </div>
  {% endblock %}
</html>
```

3.1 RESTful Αρχιτεκτονική

Το σύστημα view έχει δομηθεί με βάση την αρχιτεκτονική REST. Αυτό σημαίνει ότι κάθε πόρος(resource) αντιπροσωπεύεται από ένα μοναδικό url και οι λειτουργίες πάνω σε κάθε πόρο γίνονται με τις HTTP μεθόδους GET, POST, PUT, DELETE. Στην περίπτωση μας, έχουμε επιλέξει να μοντελοποιήσουμε κάθε μοντέλο της βάσης δεδομένων σαν έναν πόρο. Δηλαδή, τα μοντέλα user, publication, author και affiliation αντιπροσωπεύουν το καθένα από αυτά ένα πόρο.

Ακολουθεί μία συνοπτική περιγραφή των μεθόδων που δρουν πάνω στο url ενός πόρου και προσφέρουν CRUD(Create, Read, Update, Delete) λειτουργικότητα για το μοντέλο User. Το ίδιο ισχύει και για τα υπόλοιπα μοντέλα Author, Publication, Affiliation και τις σχέσεις μεταξύ αυτών.

HTTP Method	**URL**	**Action**	**Description**
GET	/users	list	Get user list
GET	/users/{user_id}	show	Get a specific user's information
POST	/users	create	Create a new user
PUT	/users/{user_id}	update	Update user information
DELETE	/users/{user_id}	delete	Delete user

Πίνακας 1: Περιγραφή των μεθόδων της REST αρχιτεκτονικής

Για να δημιουργήσουμε το restful api που περιγράφεται πιο πάνω, χρησιμοποιήσαμε τις δυνατότητες που μας δίνει το flask για την αντιστοίχηση urls σε συναρτήσεις.

Χρησιμοποιώντας την αρχιτεκτονική RESTful παρέχεται ένα ευέλικτο API, πάνω από το οποίο μπορούν να δομηθούν άλλες web εφαρμογές. Αυτό έχει ως αποτέλεσμα, η εφαρμογή μας είναι επεκτάσιμη και να μπορεί οποιοσδήποτε τρίτος να γράψει το γραφικό περιβάλλον που επιθυμεί, αν δε μείνει ικανοποιημένος από το υπάρχον γραφικό περιβάλλον.

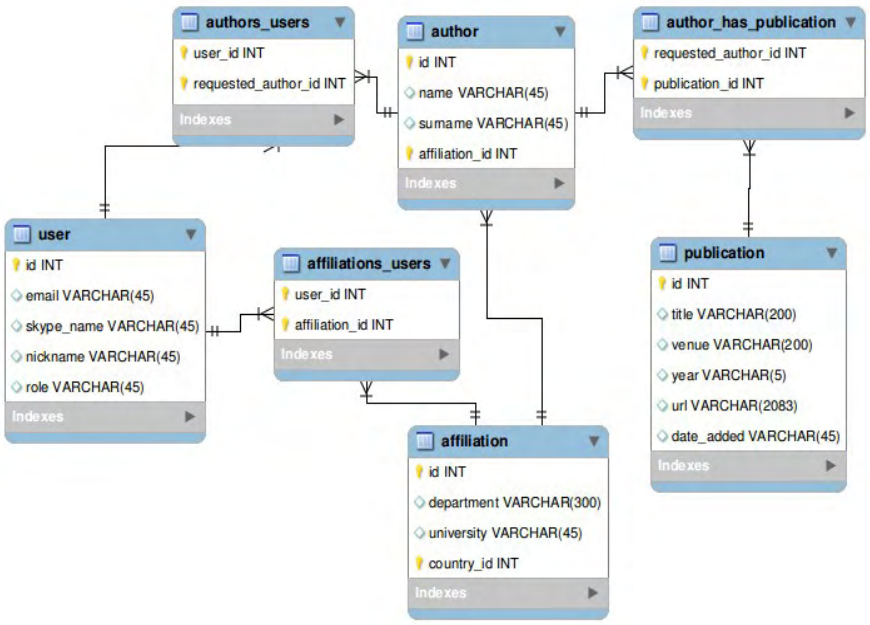
Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 4 Database handler

Το υποσύστημα Database Handler αφορά όλες τις λειτουργίες που έχουν να κάνουν με τη βάση δεδομένων. Αποτελείται από τα υποσυστήματα DB Handler και ORM, τα οποία είναι υπεύθυνα για την επικοινωνία με τη βάση δεδομένων. Σε αυτό το κεφάλαιο περιγράφεται το σχήμα της βάσης δεδομένων και τα υποσυστήματα που απαρτίζουν το σύστημα model.

4.1 Περιγραφή σχήματος ΒΔ

Με βάση τις απαιτήσεις του συστήματος, καθορίστηκε και το σχήμα της βάσης δεδομένων. Στόχος μας ήταν να διευκολύνουμε τα ερωτήματα απλοποιώντας τις σχέσεις μεταξύ των πινάκων και σχεδιάζοντας τη βάση δεδομένων έτσι ώστε να ελαχιστοποιείται ο αριθμός των διπλότυπων τιμών.



Εικ 2: Σχήμα της βάσης δεδομένων. Φαίνονται οι πίνακες και οι σχέσεις μεταξύ αυτών

User : Κάθε χρήστης αντιπροσωπεύεται από το e-mail του, το nickname του και το ρόλο του(πεδίο role). Το πεδίο role μπορεί να πάρει δύο διαφορετικές τιμές, είτε απλός user, είτε superuser. Στην περίπτωση που είναι απλός user, δε μπορεί να τροποποιήσει στοιχεία από άλλους χρήστες ενώ στην περίπτωση που είναι superuser έχει αυτή τη δυνατότητα. Προαιρετικά, για την περίπτωση που ένας χρήστης θέλει να ενημερώνεται μέσω skype, υπάρχει η δυνατότητα να αποθηκεύεται και το skype_name του.

Author : Κάθε συγγραφέας αντιπροσωπεύεται από το πλήρες όνομα του καθώς και από το ίδρυμα στο οποίο ανήκει. Η σχέση μεταξύ author – affiliation είναι πολλά προς ένα γιατί ένας συγγραφέας μπορεί σε κάθε δεδομένη χρονική στιγμή να ανήκει σε ένα ίδρυμα, ενώ ένα ίδρυμα μπορεί σε μια δεδομένη χρονική στιγμή να

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

συνεργάζεται με πολλούς συγγραφείς. Αυτή είναι μια απλοποίηση, η οποία έγινε για να διευκολύνει την υλοποίηση του συστήματος, καθώς σε περίπτωση που ένας συγγραφέας ανήκει σε περισσότερα από ένα ιδρύματα σε μια δεδομένη χρονική στιγμή, τότε δεν είναι εύκολο να αποφασιστεί σε ποιο ίδρυμα να προσμετρηθούν οι δημοσιεύσεις του συγγραφέα.

Affiliation : Κάθε ίδρυμα αντιπροσωπεύεται από το τμήμα και από το πανεπιστήμιο στο οποίο ανήκει αυτό το τμήμα. Η σχέση μεταξύ affiliation – author είναι ένα προς πολλά, γιατί ένας συγγραφέας μπορεί σε κάθε δεδομένη χρονική στιγμή να ανήκει σε ένα ίδρυμα, ενώ ένα ίδρυμα μπορεί σε μια δεδομένη χρονική στιγμή να συνεργάζεται με πολλούς συγγραφείς.

Publication : Κάθε δημοσίευση αντιπροσωπεύεται από τον τίτλο της(title), το μέρος στο οποίο έγινε(venue), το έτος(year), καθώς και ένα σύνδεσμο προς τη δημοσίευση. Επιπλέον, το πεδίο date_added χρησιμοποιείται για να γνωρίζουμε πότε μια δημοσίευση προστέθηκε στη βάση δεδομένων.

Affiliations_Users : Η σχέση μεταξύ affiliations και users χρησιμοποιείται για να περιγράψει το γεγονός ότι ένας χρήστης μπορεί να ζητήσει να παρακολουθεί κάποιο ίδρυμα. Η σχέση αυτή είναι πολλά προς πολλά, γιατί ένας χρήστης μπορεί να παρακολουθεί πολλά ιδρύματα, ενώ ένα ίδρυμα μπορεί να ζητηθεί από πολλούς χρήστες.

Authors_Users : Η σχέση μεταξύ authors και users χρησιμοποιείται για να περιγράψει το γεγονός ότι ένας χρήστης μπορεί να ζητήσει να παρακολουθεί κάποιον συγγραφέα. Η σχέση αυτή είναι πολλά προς πολλά γιατί ένας χρήστης μπορεί να παρακολουθεί πολλούς συγγραφείς, ενώ ένας συγγραφέας μπορεί να ζητηθεί από πολλούς χρήστες.

Author_has_Publication : Η σχέση μεταξύ author και publication χρησιμοποιείται για να το γεγονός ότι ένας συγγραφέας μπορεί να έχει γράψει πολλές δημοσιεύσεις, ενώ μια δημοσίευση μπορεί να γραφτεί από πολλούς συγγραφείς. Επομένως, η σχέση είναι πολλά προς πολλά.

Το σχήμα της βάσης δεδομένων είναι σε CNF έτσι ώστε να μην υπάρχουν άσκοπα διπλότυπες τιμές στη ΒΔ.

4.2 Object Relational Mapper

Object Relational Mapping είναι μια τεχνική αντιστοίχησης πινάκων της βάσης δεδομένων σε αντικείμενα της γλώσσας προγραμματισμού. Κύριος σκοπός της είναι η αποσύνδεση(decoupling) της πρώτης από τη δεύτερη. Το κύριο πλεονέκτημα της χρήσης ενός ORM είναι η εύκολη αλλαγή του συστήματος διαχείρισης βάσεως δεδομένων, χωρίς να απαιτείται τροποποίηση του κώδικα. Επίσης το ORM παρέχει προστασία από διάφορες επιθέσεις ασφαλείας όπως είναι η SQL injection.

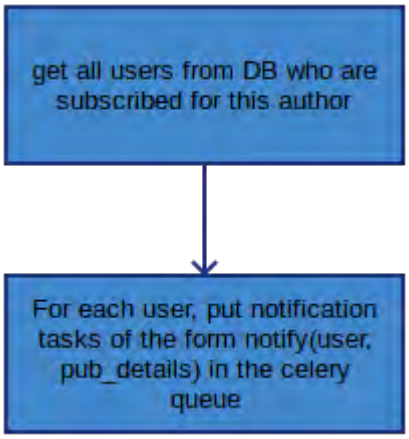
Για τους σκοπούς της παρούσας διπλωματικής επιλέχθηκε σαν ORM η [sqlalchemy]. Η αντιστοίχιση πινάκων της βάσης δεδομένων με τα αντικείμενα της γλώσσας προγραμματισμού γίνεται χρησιμοποιώντας μια Domain Specific Language στην οποία κάθε πίνακα της βάσης δεδομένων αντιπροσωπεύεται από ένα αντικείμενο και κάθε στήλη του πίνακα από μια μεταβλητή στιγμιότυπου στο αντίστοιχο αντικείμενο. Κάθε τέτοιο αντικείμενο αποκαλείται μοντέλο(model) και μπορεί να επεκταθεί με διάφορες μεθόδους.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Στην παρούσα εργασία, τα μοντέλα User, Author, Publication και Affiliation έχουν επεκταθεί με μεθόδους οι οποίες επιστρέφουν τα αντικείμενα σε μορφή JSON, για να είναι εύκολη η επεξεργασία τους αργότερα. Επιπλέον, για το μοντέλο User έχει προστεθεί μια μέθοδος που ελέγχει αν ένας χρήστης είναι authenticated, η οποία χρησιμοποιείται κατά τη διάρκεια ταυτοποίησης ενός χρήστη.

4.3 DB Handler

Ο DB Handler παρέχει όλη την λειτουργικότητα που αφορά την αποθήκευση των δημοσιεύσεων καθώς και την προεπεξεργασία τους. Χρησιμοποιεί το API του ORM για να αποθηκεύσει δημοσιεύσεις στη ΒΔ καθώς και για να εντοπίσει την ύπαρξη μια καινούριας δημοσίευσης. Πιο συγκεκριμένα, παρέχεται μία μέθοδος save(pubs) η οποία δέχεται σαν παραμέτρους μιας λίστα δημοσιεύσεων και έναν author και αποθηκεύει τις δημοσιεύσεις στη βάση δεδομένων, συσχετίζοντας τις με τον author. Στη μέθοδο save(pubs) γίνεται ο έλεγχος για νέες δημοσιεύσεις και η προ-επεξεργασία(μετατροπή σε πεζά και σε χαρακτήρες unicode) των δημοσιεύσεων πριν την αποθήκευση στη βάση δεδομένων. Ειδικότερα, κάθε φορά που καλείται η μέθοδος save με παράμετρο μια λίστα από δημοσιεύσεις και ένα συγγραφέα, γίνεται ένα επερώτημα στη ΒΔ για το αν υπάρχει κάποια από αυτές τις δημοσιεύσεις στη βάση δεδομένων. Αν δεν υπάρχει μια δημοσίευση, τότε ο DB Handler κάνει ένα επερώτημα στη ΒΔ για να βρει ποιοι χρήστες έχουν ζητήσει να λαμβάνουν ενημερώσεις από το συγκεκριμένο συγγραφέα. Έπειτα, για κάθε χρήστη στέλνει στην ουρά του celery(η λειτουργία του οποίου περιγράφεται στο επόμενο κεφάλαιο) notification tasks με παραμέτρους τα στοιχεία επικοινωνίας του χρήστη και τα στοιχεία της δημοσίευσης. Τα notification tasks περιγράφονται στο κεφάλαιο 7. Ακολουθεί μια σχηματική περιγραφή της διαδικασίας ειδοποίησης όλων των χρηστών, σε περίπτωση εντοπισμού νέας δημοσίευσης από τον DB handler.



Εικ 3: Η διαδικασία ειδοποίησης όλων των χρηστών σε περίπτωση εντοπισμού μίας νέας δημοσίευσης από τον DB Handler

Το πλεονέκτημα αυτής της παθητικής ενημέρωσης του notification manager από τον

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

DB handler είναι ότι ο notification manager ειδοποιείται αυτόματα από τον database handler μόλις προστεθεί μία νέα δημοσίευση και δε χρειάζεται να εκτελεί άσκοπα polling στη βάση δεδομένων, για να μάθει αν υπήρξε κάποια καινούρια δημοσίευση.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 5 Task Queue – Celery

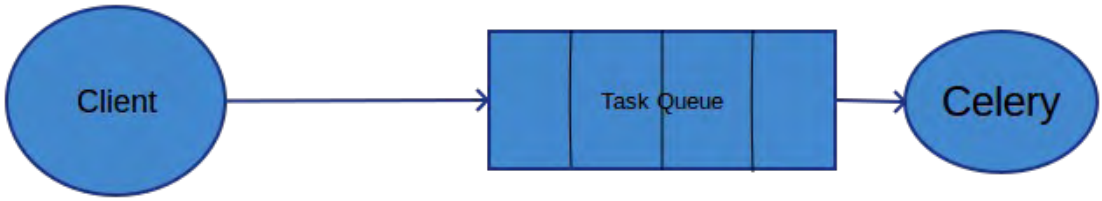
Σε αυτό το κεφάλαιο περιγράφουμε το celery, μια ουρά εργασιών, την οποία χρησιμοποιούμε για να τοποθετήσουμε και τρέξουμε όλες τις εργασίες που μπορούν να εκτελεστούν στο παρασκήνιο.

Κύριο ρόλο στην αρχιτεκτονική του συστήματος παίζει το Celery. Το Celery είναι μια ουρά εργασιών(task queue). Είναι δηλαδή μια ουρά, στην οποία μπορούν να εκτελεστούν ασύγχρονα διάφορες εργασίες(tasks). Ειδικότερα, τοποθετούμε εργασίες που επιθυμούμε να εκτελεστούν ασύγχρονα σε αυτήν την ουρά και έπειτα με κάποιο τρόπο αυτές οι εργασίες κατανέμονται αυτόματα σε επεξεργαστές, εκτελούνται και μας επιστρέφονται τα αποτελέσματα πίσω στην ουρά. Στην περίπτωση μας, ένα task μπορούμε να το φανταστούμε σαν μία συνάρτηση η οποία εκτελεί κάποιο υπολογισμό. Για παράδειγμα, θα μπορούσε αυτή η συνάρτηση να στείλει ένα e-mail, να στείλει ένα request σε ένα server ή να εκτελέσει κάποια άλλη εργασία που επιθυμούμε. Επειδή οι προηγούμενες εργασίες που αναφέρθηκαν(αποστολή e-mail, αίτημα σε server) μπορεί να είναι χρονοβόρες και δε θέλουμε το πρόγραμμα μας είναι μπλοκαρισμένο για όση ώρα διαρκεί μια τέτοια εργασία, επιλέγουμε να τις εκτελέσουμε ασύγχρονα. Δηλαδή, οι συναρτήσεις που θα καλούν αυτές τις εργασίες, θα επιστρέφουν αμέσως χωρίς να περιμένουν να ολοκληρωθεί η εργασία. Με αυτό τον τρόπο έχουμε καλύτερη αξιοποίηση των πόρων του συστήματος, γιατί για όση ώρα το σύστημα παραμένει μπλοκαρισμένο, θα μπορούσε να εκτελεί άλλους υπολογισμούς.

Στην python για να ορίσουμε μια συνάρτηση σαν ένα celery task χρησιμοποιούμε έναν decorator. Έναν decorator μπορούμε να τον φανταστούμε σα μια διαδικασία που μετασχηματίζει μια υπάρχουσα συνάρτηση, σε μια άλλη συνάρτηση ή αντικείμενο προσθέτοντας επιπλέον λειτουργικότητα, κατά τη διάρκεια της εκτέλεσης. Ο τρόπος με τον οποίο ορίζουμε μια συνάρτηση σαν ένα task στην python είναι ο εξής :

```
@task
def add(x, y):
  return x + y
```

Εφόσον έχουμε ορίσει ένα task, το celery αναλαμβάνει να δημιουργήσει ένα αντικείμενο τύπου task και να προσθέσει σε αυτό μια μέθοδο delay, η οποία κάθε φορά που καλείται, τοποθετεί ένα task στην ουρά. Από εκεί και πέρα, αναλαμβάνει το celery να αναθέσει αυτό το task σε workers, να το εκτελέσει και να επιστρέψει τα αποτελέσματα στην ουρά. Στην εικόνα που ακολουθεί φαίνεται το μοντέλο εκτέλεσης μιας εργασίας(task).



Εικ 4: Εικ 4: Το μοντέλο εκτέλεσης του celery. Ένας client εισάγει tasks στην ουρά και συνεχίζει την εκτέλεση του. Τα tasks κατανέμονται και εκτελούνται σε workers από το celery

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Στην περίπτωση μας, έχουμε ορίσει τρεις τύπους από tasks. Αυτά μπορεί να είναι :

- Crawler manager tasks (Αναλαμβάνουν να συντονίσουν τη διαδικασία του crawling και να φορτώσουν τους crawlers που απαιτούνται)
- Crawler tasks (Αναλαμβάνουν να φορτώσουν έναν custom crawler, να πάρουν τα αποτελέσματα που αυτός επιστρέφει και να τα αποθηκεύσουν στη ΒΔ)
- Notification tasks (Αναλαμβάνουν να στείλουν είτε ένα e-mail είτε ένα μήνυμα skype σε κάποιο χρήστη)

Τα συγκεκριμένα tasks περιγράφονται αναλυτικότερα στα κεφάλαια που ακολουθούν.

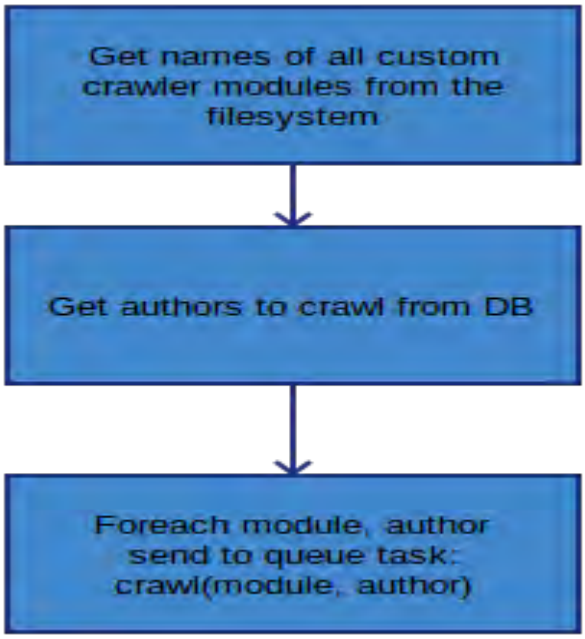
Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 6 Crawler Manager και Custom Crawlers

6.1 Crawler Manager tasks

Το celery πέρα από την ασύγχρονη εκτέλεση εργασιών, χρησιμοποιείται και για εργασίες που πρέπει να εκτελούνται περιοδικά στο παρασκήνιο. Μια από αυτές τις εργασίες είναι η διαδικασία του crawling. Κατά τη διαδικασία του crawling, θα πρέπει να εκκινούμε ανά τακτά χρονικά διαστήματα τον crawler manager ο οποίος θα αναλαμβάνει να φορτώσει τους custom crawlers που υπάρχουν στο σύστημα και να τους εκτελέσει. Για το σκοπό αυτό, έχουμε υλοποιήσει τον crawler manager σαν ένα celery task, το οποίο εκτελείται αυτόματα από το celery ανά τακτά χρονικά διαστήματα.

Ο ρόλος του crawler manager είναι να αποφασίζει ποια modules θα κάνουν crawl και για ποιους authors. Κάθε φορά που εκτελείται, διαβάζει τα ονόματα των modules από το filesystem και τα αποθηκεύει σε μια λίστα. Έπειτα, κάνει ένα επερώτημα στη βάση δεδομένων, για να μάθει για ποιους authors θα πρέπει να κάνει crawl. Μόλις λάβει τη λίστα με τους authors για τους οποίους θα πρέπει να γίνει crawl, στέλνει για κάθε author και για κάθε module ένα task στην ουρά του celery με το όνομα του module που θα πρέπει να εκτελεστεί, καθώς και το όνομα του author και τερματίζει την εκτελεσή του. Έπειτα, το celery αναλαμβάνει να διαβάσει τα tasks από την ουρά και να τα κατανείμει σε workers όπου θα εκτελεστούν. Κάθε τέτοιο task αποτελεί ένα crawler task, η λειτουργία το οποίου θα περιγραφτεί πιο κάτω. Στην επόμενη εικόνα περιγράφονται τα στάδια εκτέλεσης του crawler manager.



Εικ 5: Στάδια εκτέλεσης του crawler manager

Για να κατανοήσουμε καλύτερα τη λειτουργία του crawler manager παραθέτουμε ένα παράδειγμα:

Στην περίπτωση που υπάρχουν στο σύστημα οι DBLP και IEEE crawlers και ο crawler manager ανακαλύπτει, μετά από επερώτημα στη βάση δεδομένων, ότι πρέπει

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

να κάνει crawl για τους authors 'spyros lalis' και 'dimitrios katsaros', τότε θα στείλει στην ουρά του celery τα εξής tasks και θα τερματίσει τη λειτουργία του:

```
crawl('dblp', 'spyros lalis')
crawl('dblp', 'dimitrios katsaros')
crawl('ieee', 'spyros lalis')
crawl('ieee', 'dimitrios katsaros')
```

Έπειτα, το celery θα αναλάβει να κατανείμει αυτά τα tasks σε workers έτσι ώστε να εκτελεστούν.

6.2 Crawler tasks

Όπως αναφέραμε και προηγουμένως, ο crawler manager τοποθετεί ένα crawler task στην ουρά του celery, το οποίο αναλαμβάνει να εκτελεστεί κάποια στιγμή στο μέλλον.

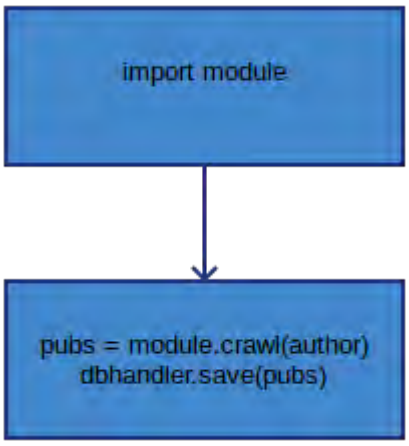
Κάθε crawler task λαμβάνει σαν παραμέτρους το όνομα του author για τον οποίο χρειάζεται να γίνει crawl καθώς και το custom crawler module το οποίο περιέχει τη λογική για την εκτέλεση του crawling. Έπειτα, φορτώνει αυτό το module στη μνήμη και αναλαμβάνει να το εκτελέσει καλώντας την crawl(author) μέθοδό του. Ο εκάστοτε custom crawler, μόλις ολοκληρώσει τη λειτουργία του, επιστρέφει στο crawler task τη λίστα με τις δημοσιεύσεις στην εξής μορφή :

```
pubs = [pub1, pub2, ...]
```

όπου ένα pub είναι ένα python dictionary(μια λίστα από key-value πλειάδες) με εξής μορφή :

```
pub = {'authors' : ['author1', 'author2', ...], 'title' : 'publication_title', venue : 'publication_venue', 'year' : 'publication_year', 'url' : 'publication_url'}.
```

Μόλις το crawler task λάβει αυτή τη λίστα, καλεί τη μέθοδο save(pub) του DB Handler έτσι ώστε να αποθηκεύσει τις δημοσιεύσεις στη βάση δεδομένων. Ακολουθεί μια σχηματική περιγραφή με τα στάδια εκτέλεσης ενός crawler task.



Εικ 6: Ένα crawler task δέχεται σαν παραμέτρους τον author και το module. Έπειτα εκτελεί το module και αποθηκεύει τις δημοσιεύσεις που θα επιστραφούν στη ΒΔ

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Για την καλύτερη κατανόηση της εκτέλεσης ενός crawler task, ακολουθεί ένα παράδειγμα :

Έστω ότι το crawler task λαμβάνει από τον crawler manager τις εξής παραμέτρους :

author : 'spyros lalis'

module : 'DBLP'

Το crawler task θα φορτώσει από το filesystem στη μνήμη το module dblp και έπειτα θα καλέσει τη μέθοδο του crawl(author) με παράμετρο τον author που έλαβε από τον crawler manager. Οι δημοσιεύσεις που θα επιστραφούν από το custom crawler module αποθηκεύονται στη βάση δεδομένων, χρησιμοποιώντας τη μέθοδο save του db handler.

6.3 Custom Crawler modules

Οι Custom Crawlers είναι βιβλιοθήκες της python(python modules) οι οποίες έχουν ως σκοπό να διαβάσουν δημοσιεύσεις από ένα website. Εκκινούνται πάντα από ένα crawler task και επιστρέφουν τις δημοσιεύσεις που διάβασαν στο crawler task από το οποίο εκκινήθηκαν. Με βάση τις απαιτήσεις του συστήματος, είναι δυνατή η δημιουργία custom crawler modules από τρίτούς.

Για να είναι εφικτή η δημιουργία custom crawler modules από τρίτους, θα πρέπει να ισχύουν οι εξής προϋποθέσεις :

1) Κάθε custom crawler θα πρέπει να είναι υλοποιημένος σαν ένα pyhon module.

2)Κάθε custom crawler θα πρέπει να έχει μια μέθοδο crawl(author) η οποία θα επιστρέφει μια λίστα με δημοσιεύσεις, όπου κάθε δημοσίευση είναι ένα dictionary με την εξής μορφή :

pub = {'authors' : ['author1', 'author2', ...], 'title' : 'publication_title', venue : 'publication_venue', 'year' : 'publication_year', 'url' : 'publication_url'}.

Εκμεταλλεύοντας την δυνατότητα της python να φορτώνει δυναμικά modules κατά τη διάρκεια εκτέλεσης του προγράμματος, μπορούμε να φορτώνουμε δυναμικά custom crawlers κατά την εκτέλεση του προγράμματος, χωρίς να απαιτείται η επανεκκίνηση του προγράμματος.

Για τις ανάγκες της παρούσας διπλωματικής έχουν υλοποιηθεί δύο custom crawlers, ο DBLPCRAWLER και ο IEEECRAWLER.

6.3.1 DBLPCRAWLER

ο **DBLPCRAWLER** χρησιμοποιεί το API του DBLP κάνοντας ένα GET request σε αυτό και έπειτα παίρνει τα αποτελέσματα σε μορφή JSON. Τα αποτελέσματα είναι μια λίστα από δημοσιεύσεις.

Για την καλύτερη κατανόηση παραθέτουμε ένα παράδειγμα ενός request στο DBLP και το αντίστοιχο response για μία δημοσίευση.

Request :

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

host : www.dblp.org

HTTP GET 1.1 /search?query=lalis

Response :


```
{
"@score":"100",
"@id":"1372182",
"info":{
"authors":{
"author":[
"Jaroslaw Domaszewicz",
"Spyros Lalis",
"Tomasz Paczesny",
"Aleksander Pruszkowski",
"Mikko Ala-Louko"]},
"title":{
"@ee":"http://dx.doi.org/10.1109/MCOM.2013.6525610",
"text":"Graspable and resource-flexible applications for pervasive
computing at home. "},
"venue":{
"@url":"db/journals/cm/cm51.html#DomaszewiczLPPA13",
"@journal":"IEEE Communications Magazine (CM)",
"@number":"6",
"@volume":"51",
"text":"IEEE Communications Magazine (CM) 51(6) (2013)"},
"year":"2013",
"type":"article"},
"url":"http://www.dblp.org/rec/bibtex/journals/cm/DomaszewiczLPPA13"
}
```


6.3.2 IEEECRAWLER

Όσον αφορά τον **IEEECRAWLER**, επειδή δεν υπήρχε κάποιο API για την ανάκτηση των δεδομένων, έπρεπε πρώτα να γίνει ένα request στο IEEE και εφόσον επιστραφεί η HTML σελίδα να παρθεί η κατάλληλη πληροφορία από εκεί. Επειδή τα αποτελέσματα των δημοσιεύσεων στο IEEEXPLORE βρίσκονται σε διαφορετικές σελίδες, αρχικά γίνεται πρώτα ένα GET request για τον author που ζητείται. Έπειτα, από την απάντηση αυτού του request, ο DBLPCRAWLER βρίσκει τον αριθμό των σελίδων και κάνει GET requests σε κάθε σελίδα ξεχωριστά για να συλλέξει όλες τις δημοσιεύσεις.

Για την ανάκτηση της κάθε δημοσίευσης, καθώς και για την εύρεση του συνολικού αριθμού σελίδων για έναν author, χρησιμοποιήθηκε η βιβλιοθήκη [beautifulsoup].

Για την καλύτερη κατανόηση παραθέτουμε ένα παράδειγμα ενός request στο DBLP και το αντίστοιχο response για μία δημοσίευση.

Request :

host : ieeexplore.ieee.org

HTTP GET 1.1 /search/searchresult.jsp?searchWithin=p_Last_Names:lalis&matchBoolean=true&queryText=(p_Authors:lalis)&rowsPerPage=100&pageNumber=1&resultAction=ROWS_PER_PAGE

Response :

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

```
<html>
 <head>
 ...
 </head>
 <body>
  ...
  <li class="noAbstract">
   <div class="header">
    <div class="select">
     <input name=" title='Select this article: A market-based protocol with leasing support for globally distributed computing' type='checkbox' value=" id='923243'/>
    </div>
   </div>
  </li>
 ...
 </body>
</html>
```


Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 7 Notification Manager και Custom Notifier tasks

7.1 Notification Manager

Ο notification manager είναι υπεύθυνος για την ειδοποίηση των χρηστών που θέλουν να λαμβάνουν ειδοποιήσεις για νέες δημοσιεύσεις. Κάθε notification manager είναι υλοποιημένος σαν ένα celery task, το οποίο αποστέλλεται στην ουρά του celery από τον DB Handler, κατά τον εντοπισμό μίας νέας δημοσίευσης, με παραμέτρους το όνομα τα στοιχεία του χρήστη και τις λεπτομέρειες της καινούριας δημοσίευσης. Ο notification manager έχοντας λάβει το όνομα χρήστη και τις λεπτομέρειες της καινούριας δημοσίευσης από τον DB handler, κάνει ένα ερώτημα στη βάση δεδομένων για να δει με ποιους τρόπους θέλει να ενημερώνεται ο συγκεκριμένος χρήστης(skype, e-mail). Έπειτα, ανάλογα με τους τρόπους που θέλει να ενημερώνεται ο χρήστης, αποφασίζει να εκτελέσει και τους αντίστοιχους custom notifiers, που αναλαμβάνουν να ειδοποιήσουν το χρήστη. Για παράδειγμα, αν ο χρήστης έχει επιλέξει να ενημερώνεται μόνο μέσω skype, τότε θα εκτελεστεί μόνο ο skype notifier και θα αποστείλει ένα μήνυμα skype στο χρήστη με τα στοιχεία της καινούριας δημοσίευσης.

Επειδή η αποστολή ενός μηνύματος σε ένα χρήστη είναι μια χρονοβόρα διαδικασία, χρησιμοποιήθηκε το celery για την ασύγχρονη εκτέλεση της. Εφόσον ο DB Handler δε μπλοκάρει πλέον κατά τη διάρκεια αποστολής ενός μηνύματος, μπορεί να εξυπηρετήσει με πιο αποδοτικό τρόπο τα αιτήματα που δέχεται.

Ο σχεδιασμός του συγκεκριμένου υποσυστήματος έχει γίνει με τέτοιο τρόπο έτσι ώστε να είναι εύκολη η προσθήκη περισσότερων ειδοποιητών, με ελάχιστες αλλαγές στον κώδικα.

7.2 Custom Notifiers

Όπως αναφέρθηκε και προηγουμένως, οι custom notifiers εκκινούνται από το notification manager σε περίπτωση που θέλει να ειδοποιηθεί κάποιος χρήστης για μια νέα δημοσίευση.

Για τις ανάγκες της παρούσας διπλωματικής, έχουν υλοποιηθεί δύο custom notifiers, ο e-mail notifier και ο skype notifier.

Ο **e-mail notifier** εφόσον λάβει το e-mail του χρήστη και τις λεπτομέρειες της κάθε δημοσίευσης, φορτώνει ένα template το οποίο αφορά το τι θα βλέπει στην οθόνη του ο χρήστης. Έπειτα, αναλαμβάνει να δημιουργήσει μία σύνδεση με τον mail server και να αποστείλει το e-mail, χρησιμοποιώντας το πρωτόκολλο SMTP. Για την αποστολή του χρησιμοποιήθηκε η βιβλιοθήκη flask-mail.

Ο **Skype notifier** είναι υπεύθυνος για την αποστολή νέων δημοσιεύσεων μέσω skype. Εφόσον, λάβει το skype_name ενός χρήστη και τις λεπτομέρειες της δημοσίευσης, αναλαμβάνει να στείλει ένα μήνυμα στο λογαριασμό skype που σχετίζεται με αυτό το skype_name. Αυτό υλοποιείται χρησιμοποιώντας τη βιβλιοθήκη Skype4Py, που είναι ένας wrapper σε python και βασίζεται στο API του skype client. Αρχικά, θα πρέπει ο notifier να συσχετιστεί με έναν client και αυτό προϋποθέτει, ότι είναι εγκατεστημένος

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

ένας client στο σύστημα και ο χρήστης έχει δώσει την έγκριση του για αυτή τη συσχέτιση.

Για τους σκοπούς της αποστολής μηνυμάτων μέσω skype, έχει δημιουργηθεί ένας λογαριασμός με το όνομα academic.publications. Ένας χρήστης θα πρέπει να προσθέσει το λογαριασμό αυτό στις επαφές του στο skype, για να μπορεί να λαμβάνει ενημερώσεις για νέες δημοσιεύσεις.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

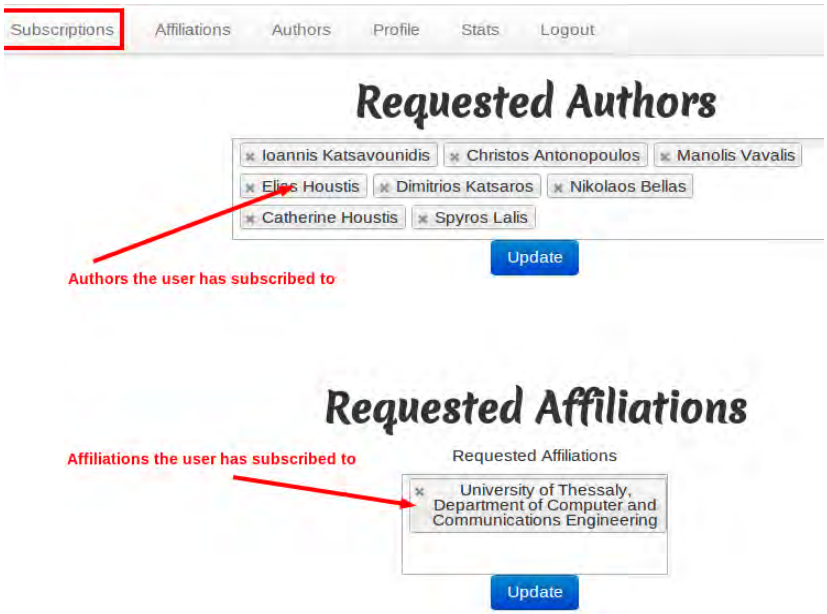
Κεφάλαιο 8 Admin και client UI

8.1 Admin UI

Ο ρόλος του admin UI είναι να παρέχει στο χρήστη έναν τρόπο διαχείρισης του περιεχομένου που θα προβάλλεται στο γραφικό περιβάλλον του client, καθώς και δυνατότητες παραμετροποίησης των ενημερώσεων που θα λαμβάνει ο χρήστης. Οι αλλαγές αυτές αντικατοπτρίζονται άμεσα στη γραφική διεπαφή του client. Το admin UI αποτελείται από τις καρτέλες subscriptions, affiliations, authors και profile.

8.1.1 Subscriptions

Ο χρήστης χρειάζεται έναν τρόπο να διαχειρίζεται τις εγγραφές του σε συγκεκριμένους ερευνητές και ιδρύματα. Για αυτό το λόγο δημιουργήθηκε η καρτέλα Subscriptions.

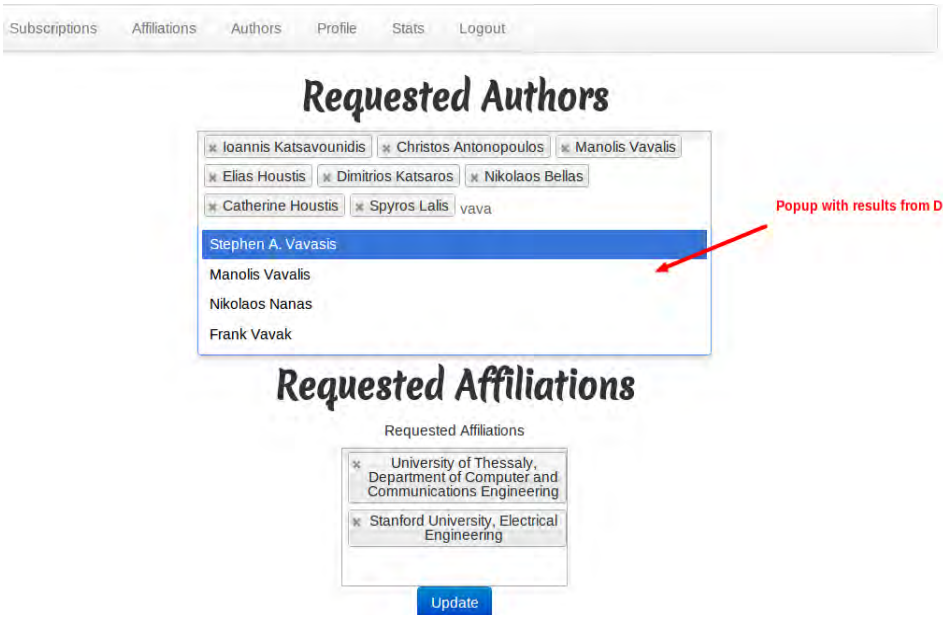


Εικ. 7: Καρτέλα Subscriptions. Φαίνονται οι ερευνητές και τα τμήματα για τα οποία έχει εγγραφεί ο χρήστης

Στην καρτέλα subscriptions, ο χρήστης μπορεί να δει για ποιους ερευνητές και για ποια τμήματα έχει ζητήσει να λαμβάνει ενημερώσεις και μπορεί να παρακολουθεί τα στατιστικά τους. Η καρτέλα αποτελείται από δύο μέρη, το Requested Authors και το Requested Affiliations.

Στο πρώτο μέρος εμφανίζονται όλοι οι ερευνητές τους οποίους παρακολουθεί ο χρήστης. Ο χρήστης μπορεί να προσθέσει και να αφαιρέσει ερευνητές. Ειδικότερα, κάνοντας κλικ μέσα στη λίστα με τους ερευνητές, εμφανίζεται ένα popup μέσα από το οποίο γίνεται αναζήτηση για έναν ερευνητή στο DBLP. Έπειτα ο χρήστης κάνοντας κλικ, μπορεί να προσθέσει αυτόν τον ερευνητή στη λίστα όπως φαίνεται και στο σχήμα που ακολουθεί.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7



Εικ 8: Στην υποενότητα Requested Authors ο χρήστης επιλέγει ερευνητές από μια λίστα που ανανεώνεται δυναμικά από το DBLP

Ο χρήστης, για να μπορεί να παρακολουθήσει κάποιον ερευνητή μέσα από το γραφικό περιβάλλον του client, θα πρέπει να έχει δηλώσει πρώτα στην υποενότητα requested authors ότι ενδιαφέρεται για αυτόν, προσθέτοντας τον στη λίστα με τους requested authors. Το σύστημα προστατεύει το χρήστη από τυχόν τυπογραφικά λάθη, εμφανίζοντας μια λίστα αυτόματης συμπλήρωσης με τους διαθέσιμους ερευνητές από το DBLP, αντί να αφήνει το χρήστη να πληκτρολογήσει ελεύθερα το όνομα του ερευνητή. Επιπλέον, αυτή η προστασία εξασφαλίζει ότι ο χρήστης δε θα προσθέσει στη λίστα με τους requested authors κάποιον ερευνητή που δεν υπάρχει. Έτσι γλιτώνουμε από άσκοπα crawling για authors που δεν υπάρχουν και από άχρηστες εγγραφές που αποθηκεύονται στη βάση δεδομένων.

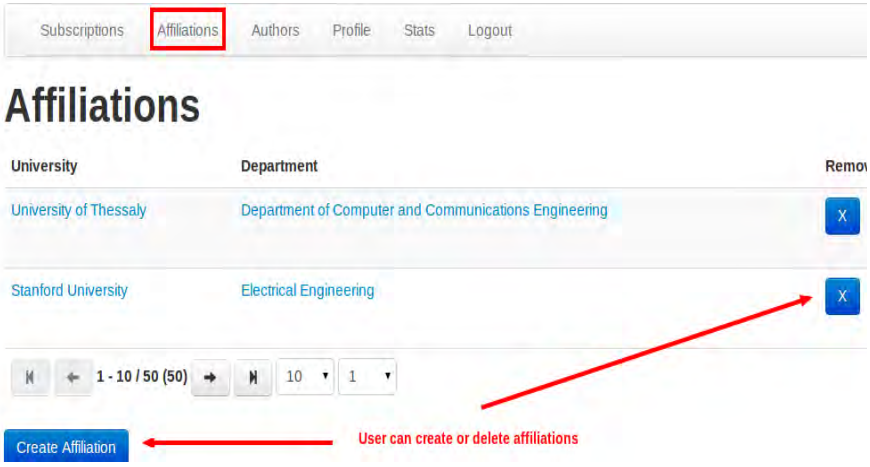
Αντίστοιχα, η ίδια λογική ισχύει και για το requested affiliations με τη διαφορά πως όταν ένας χρήστης κάνει κλικ στη λίστα, το popup που εμφανίζεται περιλαμβάνει τα ιδρύματα που βρίσκονται στη βάση δεδομένων, τα οποία έχουν προηγουμένως προστεθεί από το χρήστη μέσω της καρτέλας affiliations. Στην εικόνα που ακολουθεί, φαίνεται με ποιο τρόπο ο χρήστης προσθέτει στη λίστα ιδρύματα που τον ενδιαφέρουν.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Εικ 9: Requested affiliatons. Ο χρήστης επιλέγει ιδρύματα από μια λίστα με προκαθορισμένα ιδρύματα, τα οποία έχουν συμπληρωθεί προηγουμένως από την καρτέλα affiliations

8.1.2 Affiliations

Ο χρήστης χρειάζεται με κάποιο τρόπο να διαχειρίζεται τα ακαδημαϊκά ιδρύματα που υπάρχουν στο σύστημα. Επιπλέον, στην περίπτωση που δεν υπάρχει κάποιο ίδρυμα στο σύστημα, τότε θα χρειαστεί να το προσθέσει. Κάθε ίδρυμα αντιπροσωπεύεται από το πανεπιστήμιο στο οποίο ανήκει και το τμήμα του πανεπιστήμιου. Για παράδειγμα, στην επόμενη εικόνα η πρώτη εγγραφή αντιστοιχεί στο τμήμα μηχανικών Η/Υ του πανεπιστημίου θεσσαλίας.



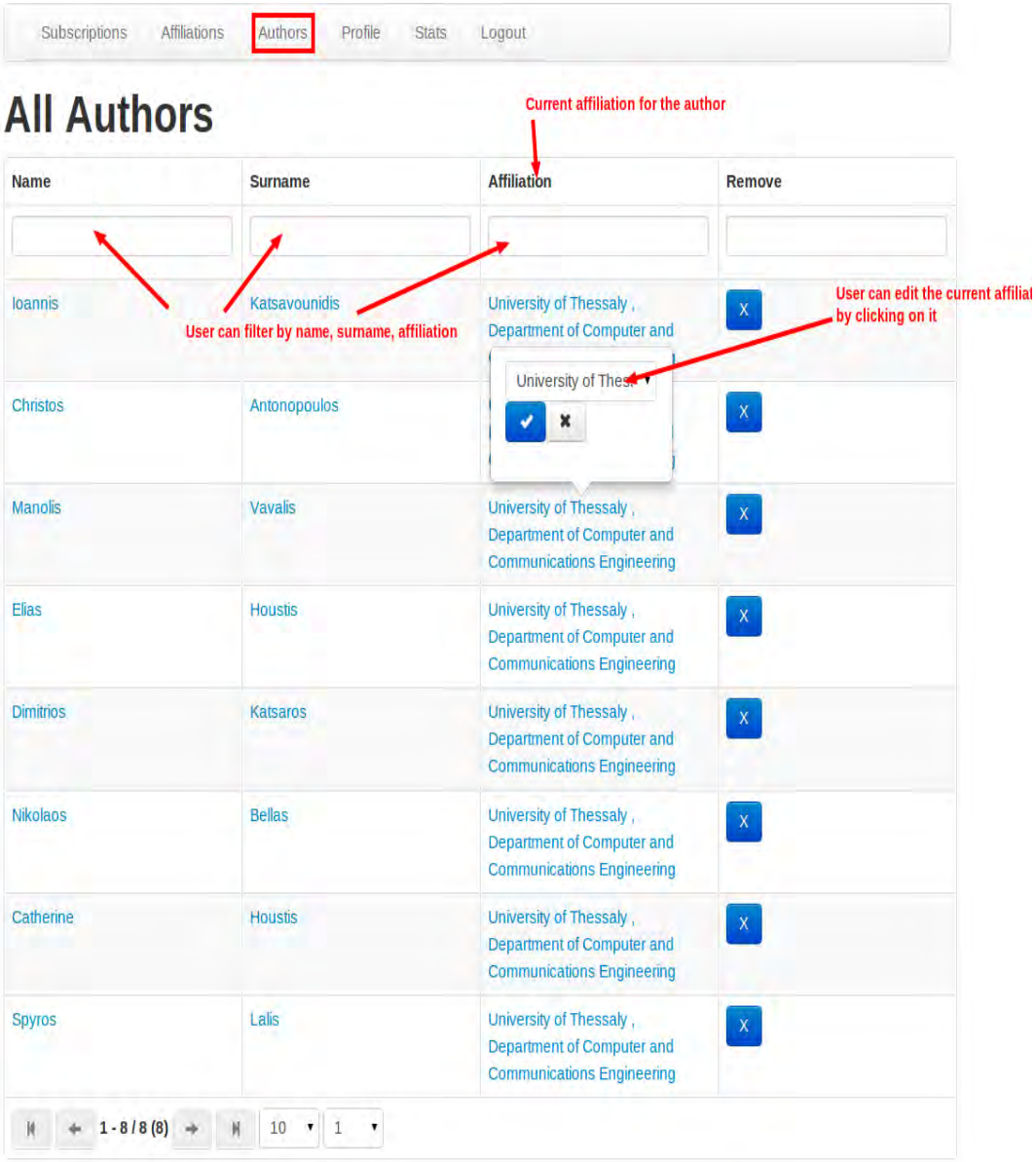
Εικ 10: Καρτέλα Affiliations. Εδώ φαίνονται όλα τα ακαδημαϊκά ιδρύματα τα οποία υπάρχουν στο σύστημα. Ο χρήστης μπορεί να προσθέσει ή να αφαιρέσει τμήματα και να τροποποιήσει τις λεπτομέρειες του καθενός

Ο χρήστης μέσα από αυτή την καρτέλα, μπορεί να δει όλα τα ιδρύματα που υπάρχουν στο σύστημα, να προσθέσει, να διαγράψει ή να τροποποιήσει τις λεπτομέρειες ενός ιδρύματος. Όλα τα ιδρύματα που βρίσκονται σε αυτή την καρτέλα εμφανίζονται στη λίστα αυτόματης συμπλήρωσης, όταν ο χρήστης πάει να γραφτεί για ένα ίδρυμα στην καρτέλα Subscriptions.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

8.1.3 Authors

Ο χρήστης χρειάζεται με κάποιο τρόπο να βλέπει ποιοι ερευνητές υπάρχουν στο σύστημα, ανεξαρτήτως του αν έχει εγγραφεί για αυτούς τους ερευνητές ή όχι. Επίσης, το σύστημα χρειάζεται με κάποιο τρόπο να γνωρίζει σε ποιο ίδρυμα ανήκει ο κάθε ερευνητής, για να μπορεί να εκτελέσει αποτελεσματικά το crawling. Η πληροφορία του ότι ένας ερευνητής ανήκει σε κάποιο ίδρυμα, θα πρέπει να εισαχθεί από το χρήστη . Αυτή η διαδικασία γίνεται στην καρτέλα Authors.



Εικ 11: Καρτέλα Authors. Στην καρτέλα φαίνονται όλοι οι ερευνητές που υπάρχουν στο σύστημα. Ο χρήστης μπορεί να προσθαφαιρέσει ερευνητές καθώς και να δηλώσει σε ποιο τμήμα ανήκει ο κάθε ερευνητής

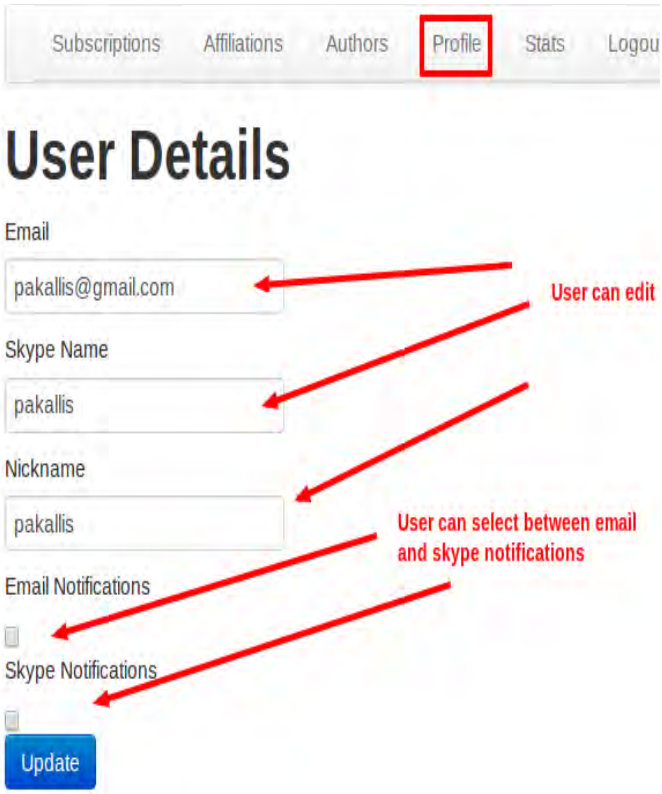
Όπως βλέπουμε και στην εικόνα 6, ο χρήστης μπορεί να διαχειριστεί όλους τους ερευνητές που υπάρχουν στο σύστημα. Ειδικότερα, υπάρχει η δυνατότητα για αναζήτηση, προβολή, προσθήκη και διαγραφή ερευνητών. Επιπλέον, μέσα από αυτή

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

την καρτέλα επιλέγεται σε ποιο ίδρυμα ανήκει ο κάθε ερευνητής. Αυτό επιτυγχάνεται, κάνοντας κλικ μέσα στο κελί που αναφέρει το ίδρυμα για κάθε ερευνητή και επιλέγοντας ένα από τα διαθέσιμα ιδρύματα που υπάρχουν στο σύστημα.

8.1.4 Profile

Ο χρήστης στις περιπτώσεις που δε θέλει να λαμβάνει e-mail ή μηνύματα skype, θα πρέπει με κάποιο τρόπο να ενημερώνει το σύστημα για αυτές του τις προτιμήσεις. Επιπλέον, αν ο χρήστης θέλει να αλλάξει το e-mail στο οποίο θα ενημερώνεται ή το skype name του, θα πρέπει να του δίνεται αυτή η δυνατότητα. Για τους σκοπούς αυτούς, έχει δημιουργηθεί η καρτέλα profile. Στην εικόνα που ακολουθεί φαίνεται η συγκεκριμένη καρτέλα καθώς και τα πεδία που μπορεί να τροποποιήσει ο χρήστης.

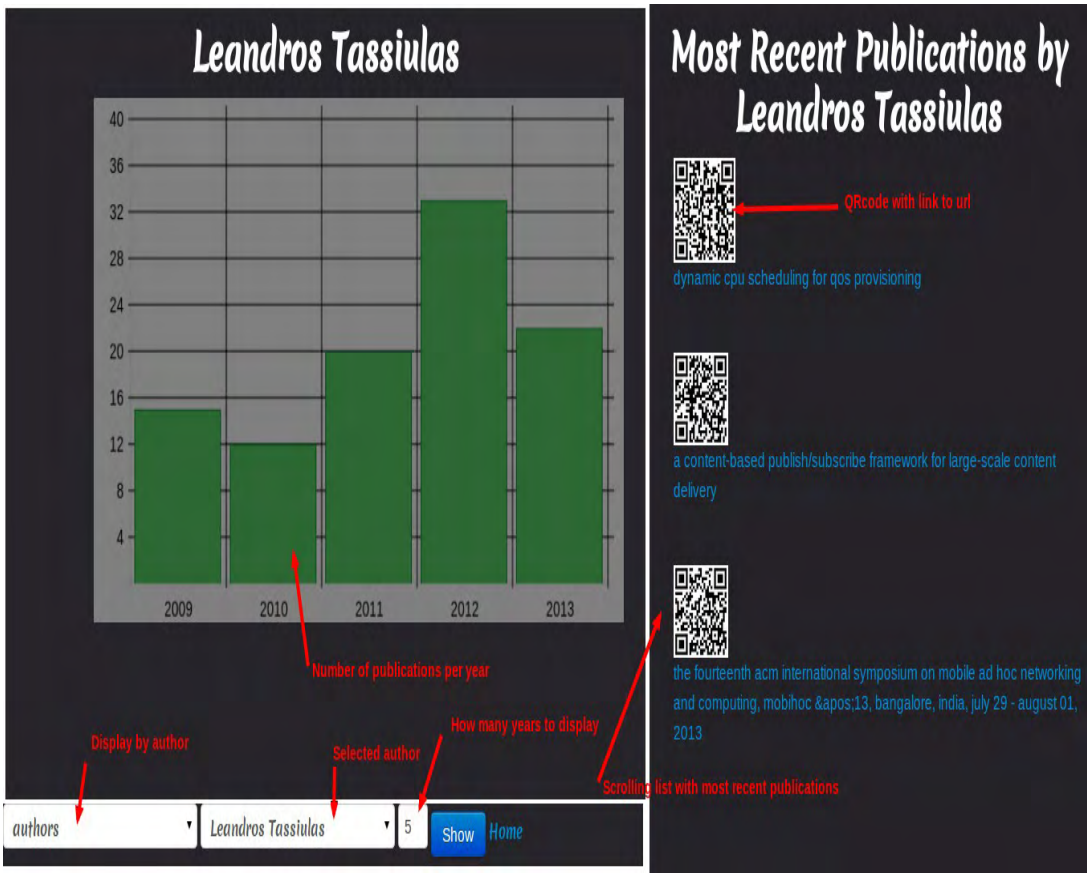


Εικ 12: Καρτέλα Profile. Ο χρήστης μπορεί να τροποποιήσει τα προσωπικά του στοιχεία και να επιλέξει τι είδους ενημερώσεις θα λαμβάνει

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

8.2 Client UI

Το client UI αναπτύχθηκε με σκοπό να αναρτηθεί σε έναν πίνακα ανακοινώσεων, έτσι ώστε να είναι ορατό από όλα τα μέλη της ακαδημαϊκής κοινότητας. Έτσι, κάθε μέλος της κοινότητας θα μπορεί να παρακολουθεί τις επιδόσεις του τμήματος συνολικά, αλλά και του κάθε διδάσκοντος. Επιπλέον, θα είναι σε θέση να ενημερωθεί για καινούριες δημοσιεύσεις και να τις διαβάσει άμεσα από φορητές συσκευές,, σκανάροντας το QR Code που βρίσκεται πάνω από τον τίτλο κάθε δημοσίευσης.



Εικ 13: Κεντρική οθόνη του client UI. Στην αριστερή στήλη φαίνεται ανά έτος ο αριθμός των δημοσιεύσεων. Στη δεξιά στήλη φαίνονται οι πιο πρόσφατες δημοσιεύσεις

Το client UI αποτελείται από τρία επιμέρους τμήματα.

Το πρώτο είναι το τμήμα που ο χρήστης μπορεί να επιλέξει τι θα εμφανίζεται στην οθόνη του. Αρχικά, υπάρχει η επιλογή μεταξύ author και affiliation. Μεσω αυτής μπορεί να επιλέξει αν τα δεδομένα θα εμφανίζονται ανά affiliation ή ανά author. Ανάλογα με την επιλογή αυτή, εμφανίζεται ένα πρόσθετο μενού με όλους τους authors για τους οποίους ο χρήστης έχει εγγραφεί, ή με όλα τα affilliations για τα οποία έχει εγγραφεί. Επιπλέον, ο χρήστης μπορεί να επιλέξει τον αριθμό των ετών για τα οποία θα εμφανίζεται ο αριθμός των δημοσιεύσεων. Τέλος, υπάρχει ένα κουμπί για την προβολή των αποτελεσμάτων και ένα ακόμη για την επιστροφή στην αρχική σελίδα.

Το δεύτερο τμήμα είναι η οθόνη στην οποία εμφανίζεται ένα γράφημα με τον αριθμό των δημοσιεύσεων ανά έτος, για κάθε author ή για κάθε affiliation. Το γράφημα

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

προσαρμόζεται δυναμικά, ανάλογα με τον αριθμό των δημοσιεύσεων καθώς και τον αριθμό των ετών. Επιπλέον, ανανεώνεται αυτομάτως, χωρίς να χρειάζεται να επέμβει ο χρήστης.

Το τρίτο είναι η οθόνη που προβάλλει τις πιο πρόσφατες δημοσιεύσεις. Σε αυτήν υπάρχει μια λίστα, η οποία εμφανίζει τον τίτλο της κάθε δημοσίευσης και ένα QR Code, το οποίο αντιπροσωπεύει το url της δημοσίευσης, έτσι ώστε να είναι εύκολη η ανάγνωση του από φορητές συσκευές. Ο ενδιαφερόμενος χρήστης για παράδειγμα, θα μπορεί με το κινητό του τηλέφωνο να διαβάσει το QR Code από μία συγκεκριμένη δημοσίευση, το οποίο αντιστοιχίζεται σε ένα url και έπειτα να επισκεφτεί αυτό το url και να διαβάσει περισσότερες λεπτομέρειες που αφορούν τη δημοσίευση.

Το γραφικό περιβάλλον του client ανανεώνεται δυναμικά κάθε φορά που βγαίνει μία καινούρια δημοσίευση ή προστίθεται στο σύστημα κάποιος νέος ερευνητής ή ίδρυμα. Αυτό σημαίνει ότι ο χρήστης δε χρειάζεται να επεμβαίνει κάθε φορά στο γραφικό περιβάλλον, για να κάνει ανανέωση της σελίδας, έτσι ώστε να φαίνονται οι καινούριες δημοσιεύσεις ή τα καινούρια στατιστικά.

Η επικοινωνία μεταξύ client-server γίνεται χρησιμοποιώντας polling. Ειδικότερα, ανά τακτά χρονικά διαστήματα ο client κάνει Ajax αιτήματα προς τον request handler, για να μάθει αν υπήρξαν νέες δημοσιεύσεις για το συγκεκριμένο author ή affiliation.

Χρησιμοποιώντας την τεχνολογία Ajax, ο client λαμβάνει μόνο την πληροφορία που χρειάζεται, χωρίς να απαιτείται η ανανέωση ολόκληρης της σελίδας.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 9 Ασφάλεια εφαρμογής

Στην παρούσα εργασία λάβαμε υπόψιν μας τις παραμέτρους ασφαλείας που απαιτούνται, έτσι ώστε να έχουμε μια πλήρως λειτουργική εφαρμογή που θα είναι προστατευμένη από τις πιο γνωστές επιθέσεις ασφαλείας.

Για την προστασία από μη εξουσιοδοτημένη χρήση των λειτουργιών του website, έχει υλοποιηθεί ένα βασικό σχήμα authentication -authorization.

Για την υλοποίηση του **authentication** έχει χρησιμοποιηθεί η τεχνολογία [openID]. Επιλέξαμε να χρησιμοποιήσουμε την τεχνολογία openID για να εκμεταλλευτούμε τις δυνατότητες ασφαλείας που μας δίνει και επιπλέον για να κάνουμε πιο εύκολη τη διαδικασία ταυτοποίησης. Με βάση το πρωτόκολλο που χρησιμοποιεί το openid, ο κωδικός χρήστη βρίσκεται αποθηκευμένος σε ένα τρίτο ιστότοπο και δεν αποθηκεύεται στη βάση δεδομένων του δικού μας ιστοτόπου. Με αυτό τον τρόπο, δε χρειάζεται να λάβουμε πρόσθετα μέτρα ασφαλείας, όπως είναι η κρυπτογράφηση του κωδικού στη βάση δεδομένων για να μην είναι δυνατή η ανάγνωση του από τρίτους. Επιπλέον, χρησιμοποιώντας το openID ο χρήστης μπορεί να ταυτοποιηθεί χωρίς να απαιτείται η εγγραφή του στον ιστότοπο μας. Έτσι, η διαδικασία ταυτοποίησης γίνεται πιο απλή για το χρήστη.

Authorization είναι η διαδικασία καθορισμού πρόσβασης χρηστών σε πόρους. Για τις ανάγκες του συγκεκριμένου συστήματος έχουν καθοριστεί δύο τύποι χρηστών: οι απλοί και οι διαχειριστές. Οι τελευταίοι έχουν πρόσβαση σε όλους τους πόρους με πλήρη δικαιώματα πρόσβασης(δημιουργία, διαγραφή, τροποποίηση). Οι απλοί χρήστες έχουν πρόσβαση μόνο σε πόρους που αφορούν τους ίδιους. Για την υλοποίηση του authorization έχουν χρησιμοποιηθεί βοηθητικές βιβλιοθήκες του flask(flask-auth).

Το Flask προσφέρει προστασία από επιθέσεις [**SQL Injection]**, παρέχοντας μεθόδους που ελέγχουν το input για κακόβουλο κώδικα. Χρησιμοποιώντας αυτές τις μεθόδους και προσέχοντας να ελέγχεται ο κώδικας SQL για κακόβουλες εισαγωγές δεδομένων, εξασφαλίζεται η προστασία του ιστοτόπου από αυτές τις επιθέσεις.
Επιπλέον, το flask εξασφαλίζει την προστασία από επιθέσεις [**CSRF]** και [**XSS]**.
Οι επιθέσεις που περιγράφτηκαν προηγουμένως είναι οι πιο δημοφιλείς για εφαρμογές web και γι αυτό δώσαμε μεγάλη προτεραιότητα στην προστασία από αυτές τις επιθέσεις.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 10 Επίλογος

10.1 Συμπεράσματα

Η παρούσα εργασία είχε ως στόχο την υλοποίηση ενός συστήματος παρακολούθησης ακαδημαϊκών δημοσιεύσεων, προσαρμοσμένου στις ανάγκες ενός ερευνητικού ιδρύματος. Κύριος σκοπός μας ήταν να προσφέρουμε στα μέλη του κάθε ιδρύματος μια πλήρως λειτουργική εφαρμογή, η οποία θα μπορούσε να ενσωματωθεί σε μια οθόνη στον πίνακα ανακοινώσεων του κάθε ακαδημαϊκού τμήματος μέσα από την οποία θα μπορούν οι χρήστες να παρακολουθούν το ερευνητικό έργο του τμήματος, σε μια φιλική προς το χρήστη γραφική διεπαφή. Επιπλέον, οι χρήστες θα μπορούν να λαμβάνουν ενημερώσεις είτε μέσω e-mail είτε μέσω ειδοποιήσεων skype για νέες δημοσιεύσεις σχετικές με το τμήμα.

10.2 Μελλοντική εργασία

Το σύστημα θα μπορούσε να επεκταθεί μελλοντικά για να παρέχει καλύτερη λειτουργικότητα στο χρήστη. Ακολουθούν μερικές ιδέες για μελλοντικές επεκτάσεις.

Είναι γνωστό ότι ο αριθμός των δημοσιεύσεων δεν είναι αξιόπιστη μετρική αξιολόγησης ενός ερευνητή ή ενός ιδρύματος. Τη δεδομένη χρονική στιγμή, ο πιο αξιόπιστος δείκτης θεωρείται ο h-index. Επομένως, στην οθόνη εμφάνισης των στατιστικών θα μπορούσε αντί για τον αριθμό των δημοσιεύσεων, να εμφανίζεται ο h-index, έτσι ώστε να παρουσιάζεται μια πιο αξιόπιστη εικόνα για κάθε ερευνητή και ίδρυμα.

Επιπλέον, θα μπορούσαν να υλοποιηθούν επεκτάσεις και για άλλες ψηφιακές βιβλιοθήκες, έτσι ώστε να αυξηθεί ο αριθμός των δημοσιεύσεων. Για παράδειγμα, θα μπορούσε να υλοποιηθεί μία επέκταση, που θα ψάχνει για δημοσιεύσεις στο Google Scholar, έτσι ώστε να αυξηθεί ο αριθμός των προς αναζήτηση δημοσιεύσεων και κατά συνέπεια να βελτιωθεί η ποιότητα των αποτελεσμάτων.

Τέλος, άλλη μία βελτίωση θα ήταν η προσθήκη ενός co-author graph, δηλαδή ενός γραφήματος μέσα στο οποίο θα εμφανίζονται οι αλληλεπιδράσεις μεταξύ των ερευνητών του τμήματος. Έτσι ο χρήστης θα μπορεί να βλέπει ποιοι ερευνητές συνεργάζονται με ποιους.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Κεφάλαιο 11 Βιβλιογραφία

[1] [FlaskMegaTutorial]

http://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world

[2] [Introduction to databases Stanford course]

https://class2go.stanford.edu/db/

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Παράρτημα Α - Τεχνολογίες που χρησιμοποιήθηκαν

Η **Javascript** είναι η πιο διαδεδομένη client-side γλώσσα προγραμματισμού. Χρησιμοποιείται για τη δυναμική τροποποίηση του περιεχόμενου μιας ιστοσελίδας, έτσι ώστε ο χρήστης να μπορεί να αλληλεπιδράσει με αυτή.

Η **Ajax (Asynchronous JavaScript And XML)** είναι μία τεχνολογία που χρησιμοποιείται για την ασύγχρονη φόρτωση τμημάτων μιας σελίδας χωρίς να απατείται η ανανέωση της από το χρήστη.

Η **JQuery** είναι μια javascript βιβλιοθήκη, η οποία επιτρέπει την εύκολη τροποποίηση του DOM της html σελίδας, απαιτώντας λιγότερη δουλειά από την πλευρά του προγραμματιστή. Επιπλέον, παρέχει έτοιμες λύσεις δημιουργίας γραφικών διεπαφών με ελάχιστο κόπο.

Η **jinja2** είναι μία βιβλιοθήκη μετατροπής κώδικα python σε html. Χρησιμοποιείται κυρίως για να εξαφανίσει την ανάγκη εγγραφής διπλότυπου κώδικα html, που επαναλαμβάνεται είτε στο ίδιο είτε σε διαφορετικά αρχεία.

Η **Python** είναι μια διερμηνευόμενη γλώσσα προγραμματισμού
υψηλού επιπέδου, με έμφαση στην ευκολία σύνταξης και διαχείρισης του κώδικα, χωρίς να υπολείπεται σε δυνατότητες. Παρέχει στον προγραμματιστή μια πληθώρα βιβλιοθηκών και πολλαπλές μεθοδολογίες δόμησης προγραμμάτων, όπως διαδικαστικό, αντικειμενοστραφή και scripting προγραμματισμό. Το οικοσύστημα το οποίο έχει δημιουργηθεί γύρω της την καθιστά ευρέως αποδεκτή και σε αυτό έχει βοηθήσει η υιοθέτησή της από την Google και διάφορα
πανεπιστήμια του εξωτερικού (MIT κ.α.).

Το **Celery** είναι ένα εργαλείο γραμμένο σε python, το οποίο
είναι υπεύθυνο για την ασύγχρονη εκτέλεση διάφορων διεργασιών (tasks) στο παρασκήνιο. Πρόκειται για μια ουρά διεργασιών (task (job) queue) που βασίζεται σε μετάδοση μηνυμάτων. Ο ρόλος του είναι να μεταβιβάζει μηνύματα σε
workers που τα επεξεργάζονται και έπειτα να λαμβάνει τα αποτελέσματα. Είναι ιδανικό για την αποφόρτιση του server από βαριές και χρονοβόρες διεργασίες, οι οποίες μπορούν να εκτελεστούν κάποια στιγμή στο μέλλον.

Η **SQLAlchemy** είναι ένας μηχανισμός αντιστοίχησης αντικειμένων της εκάστοτε γλώσσας προγραμματισμού(python), σε οντότητες της βάσης δεδομένων(**Object relational mapper**). Παρέχει ένα επίπεδο αφαίρεσης πάνω από αυτή, έτσι ώστε να είναι εφικτή η χρήση οποιασδήποτε βάσης δεδομένων χωρίς να χρειάζεται να γραφτεί κώδικας SQL. Επιπλέον, παρέχει προστασία από επιθέσεις του τύπου SQL injection, δυνατότητες session management και transactions.

Το **Flask** είναι σύνολο από βιβλιοθήκες και εργαλεία γραμμένες σε python που βοηθάνε στην ταχύτερη ανάπτυξη εφαρμογών web. Παρέχει δυνατότητες authentication, form validation και session management καθώς και προστασία από διάφορες επιθέσεις όπως cross-site request forgery(CSRF) και cross-site scripting(XSS).

Η **MySQL** είναι η πιο γνωστή Βάση Δεδομένων ανοιχτού κώδικα για τη διαχείριση

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7

Σχεσιακών Βάσεων (Relational Database Management System-RDBMS). Χρησιμοποιείται ευρέως από τους από τους web developers και παρέχει όλα τα χαρακτηριστικά που θα περίμενε κανείς από σύγχρονο σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων όπως. Οι κύριοι λόγοι για τους οποίους επιλέχθηκε είναι οι ACID(atomicity, consistency, isolation, durability) ιδιότητες και η υποστήριξη χαρακτήρων unicode.

Το **OpenID** είναι ένα πρότυπο, το οποίο επιτρέπει στους χρήστες να ταυτοποιηθούν σε κάποιο ιστότοπο χρησιμοποιώντας στοιχεία τους που είναι διαθέσιμα από έναν άλλο ιστότοπο. Ο χρήστης μπορεί να εγγραφεί για ένα λογαριασμό openID και μετά να χρησιμοποιεί τα στοιχεία του από αυτόν το λογαριασμό για να ταυτοποιηθεί σε άλλα websites.

Institutional Repository - Library & Information Centre - University of Thessaly
09/12/2017 03:21:38 EET - 137.108.70.7