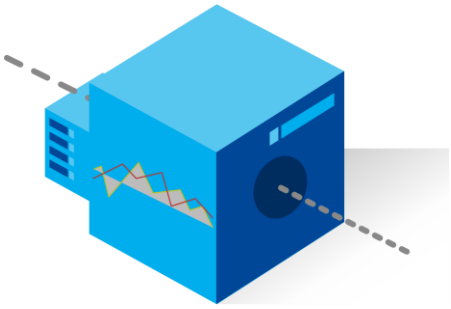
ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ



“Ανάκτηση και διαχείριση Tweets από τη βάση του Twitter (Tweets Retrieval and management of Twitter database) “

Πτυχιακή Εργασία των

Παντούλα Ιωάννα (3846)

Παπάς Μαυρουδής (3814)

Επιβλέπων καθηγητής: Δρ. Τσιμπίρης Αλκιβιάδης

ΣΕΡΡΕΣ, 2019

ΠΕΡΙΛΗΨΗ

Το Twittics χρησιμοποιεί το Twitter API (REST) για να αποκτήσει όλες τις δημόσια διαθέσιμες πληροφορίες από το προφίλ κάθε χρήστη. Το αποθηκεύει προσωρινά σε μια βάση δεδομένων, αναλύει το περιεχόμενο και το φιλτράρει για να δώσει τα επιθυμητά δεδομένα.

Ο κάθε χρήστης μπορεί να δει τους followers, τους following, τα συνολικά like και πολλές άλλες ενδιαφέρουσες πληροφορίες, σχετικές με το προφίλ του. Μπορεί να συγκρίνει το προφίλ του με τους followers και τους following του καθώς και να ανακαλύψει τις πιο δραστήριες στιγμές του μέσα στην ημέρα.

Ταυτόχρονα, όλα τα δεδομένα αναπαρίστανται με την μορφή γραφημάτων και διαγραμμάτων. Ο κάθε χρήστης έχει την δυνατότητα να δει εικονικά τα αποτελέσματα των συγκρίσεων του. Η αναπαράσταση αυτή βοηθάει τον χρήστη για την καλύτερη και πιο εύχρηστη κατανόηση των αποτελεσμάτων του.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΕΙΣΑΓΩΓΗ

Το Twittics, είναι ένα εργαλείο για την παρουσίαση στατιστικών και δεδομένων σχετικά με το Twitter προφίλ ενός χρήστη. Για την υλοποίηση αυτής της εφαρμογής, χρησιμοποιεί κάποιες από τις πιο πρόσφατες Web τεχνολογίες όπως, JQuery, AngularJS, PHP7, MySQL. Για την διευκόλυνση και ευκολότερη κατανόηση των δεδομένων, επιλέχθηκε η παρουσίαση μέσω γραφημάτων και διαγραμμάτων. Ο χρήστης έχει επίσης την δυνατότητα να επιλέξει ανάμεσα σε Ελληνική και Αγγλική γλώσσα για την προβολή των δεδομένων αυτών.

Το Twittics χρησιμοποιεί μια βάση δεδομένων MySQL για την αποθήκευση των δεδομένων κάθε χρήστη. Με αυτόν τον τρόπο ελαττώνονται τα request στην πλατφόρμα του Twitter και μειώνεται έτσι ο φόρτος.

ΤΕΧΝΟΛΟΓΙΕΣ ΚΑΙ ΕΡΓΑΛΕΙΑ

Για την υλοποίηση της πτυχιακής μας (Twittics), χρησιμοποιήθηκαν Web τεχνολογίες, δηλαδή HTML, CSS (Cascading Style Sheets), JavaScript, PHP 7 καθώς και MySQL για την υλοποίηση και την λειτουργία της βάσης δεδομένων. Η εφαρμογή γράφτηκε με την χρήση του Visual Studio Code, επίσης κατά την ανάπτυξη χρησιμοποιήθηκε το XAMPP για το local hosting της εφαρμογής και της βάσης δεδομένων μας. Παρακάτω θα δούμε αναλυτικά τις τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν, καθώς και γιατί επιλέχθηκαν ανάμεσα σε άλλα.

ΤΕΧΝΟΛΟΓΙΕΣ

HTML / CSS

Η CSS είναι μια γλώσσα που χρησιμοποιείται για την περιγραφή της παρουσίασης ενός εγγράφου γραμμένο σε μια γλώσσα σήμανσης όπως HTML. Η CSS είναι μια τεχνολογία ακρογωνιαίου λίθου του World Wide Web (WWW), μαζί με την HTML και την JavaScript.

Η HTML είναι η τυπική γλώσσα σήμανσης για τη δημιουργία ιστοσελίδων και εφαρμογών ιστού. Με την CSS και την JavaScript, αποτελεί μια τριάδα τεχνολογιών ακρογωνιαίου λίθου για τον του World Wide Web (WWW).

Η HTML και η CSS είναι ο βασικός κορμός για την υλοποίηση της εφαρμογής μας. Αποτελούν την βασική δομή της εφαρμογής και είναι στην ουσία υπεύθυνες για για το οπτικό μέρος (UI) της εφαρμογής μας.

JavaScript / JQuery

Η JavaScript, είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου που ερμηνεύεται σύμφωνα με την προδιαγραφή ECMAScript. Πρόκειται για μια γλώσσα προγραμματισμού που χαρακτηρίζεται ως δυναμική, ασθενώς δακτυλογραφημένη, βασισμένη στο πρωτότυπο και πολύ-παραδειγματική.

Η JavaScript σε συνδυασμό με την JQuery αποτελούν το λειτουργικό κομμάτι της εφαρμογής. Χωρίς αυτές δεν θα ήταν δυνατές οι περισσότερες από τις λειτουργίες της εφαρμογής.

AngularJs

Η AngularJS είναι ένα framework βασισμένο σε JavaScript και συντηρείται κυρίως από την Google. Είναι ένα framework που κατασκευάστηκε για να βοηθήσει σε θέματα και δυσκολίες που αντιμετωπίζονται κατά την δημιουργία ιστοσελίδων και εφαρμογών μιας σελίδας (single-page applications).

Η AngularJs επιλέχθηκε για να μας διευκολύνει στην δημιουργία της εφαρμογής μας. Με την χρήση της AngularJS η ανάπτυξη μια τέτοιας εφαρμογής γίνεται πολύ γρηγορότερα από αν χρησιμοποιούσαμε μόνο JavaScript και JQuery. Με την AngularJS μπορείς να διαχειρίζεσαι το DOM με μεγαλύτερη ευκολία.

Bootstrap

Το Bootstrap είναι ένα ελεύθερο και ανοιχτό front-end web framework. Περιέχει πρότυπα σχεδίασης HTML και CSS για τυπογραφία, φόρμες, κουμπιά, πλοήγηση και άλλα στοιχεία διεπαφής, καθώς και προαιρετικές επεκτάσεις JavaScript.

Με την χρήση του Bootstrap το UI της εφαρμογής γίνεται εύκολα και γρήγορα. Οι πολλές CSS κλάσεις που παρέχει επιταχύνουν την ανάπτυξη της εφαρμογής και μειώνουν τις γραμμές κώδικα. Το Bootstrap επιλέχθηκε διότι έχει ως γνώμονα το Responsive Design, πράγμα σημαντικό για την εφαρμογή μας.

PHP 7

Η PHP (Hypertext Preprocessor) είναι μια γλώσσα προγραμματισμού γενικής χρήσης που σχεδιάστηκε αρχικά για την ανάπτυξη ιστού.

Η PHP επιλέχθηκε για χρήση στον server μας (backend). Η PHP είναι μια γνώριμη σε εμάς γλώσσα και μία από τις πιο διαδεδομένες στις web εφαρμογές. Ένας από τους κύριους λόγους που την προτιμήσαμε είναι διότι η βιβλιοθήκη για το OAuth authentication του Twitter (Abraham Twitter oAuth), είναι βασισμένη σε PHP.

TwitterOAuth PHP Library

Η συγκεκριμένη βιβλιοθήκη είναι μια από τις σημαντικότερες στην εφαρμογή. Μας επιτρέπει να ξεκινήσουμε την πρώτη επικοινωνία με το Twitter ώστε να μπορέσουμε αργότερα να πάρουμε τα δεδομένα του κάθε χρήστη. Πιο συγκεκριμένα, χρησιμοποιείται για το oAuth authentication με τον server του Twitter, πράγμα πολύ σημαντικό για την

ασφάλεια του χρήστη, καθώς επίσης διασφαλίζει και την πληρότητα και ορθότητα των δεδομένων του.

Η συγκεκριμένη βιβλιοθήκη επιλέχθηκε διότι είναι η πιο γνωστή βιβλιοθήκη και είναι προτεινόμενη και από το ίδιο το Twitter. Επιπλέον είναι γραμμένη πάνω στην PHP, γλώσσα την οποία είμαστε εξοικειωμένοι.

Άλλες τεχνολογίες και Frameworks

Εκτός από τα παραπάνω, έγινε χρήση του MomentJS, για εύκολους υπολογισμούς και μετατροπές σχετικά με ημερομηνίες και ώρες, Animate.CSS, για τα animation της εφαρμογής, fontAwesome για τα διάφορα εικονίδια που χρειάστηκαν και το Chart.js για την κατασκευή των διαγραμμάτων την εφαρμογής.

ΕΡΓΑΛΕΙΑ

Visual Studio Code

Το Visual Studio Code είναι ένας από τους γνωστότερους editors / IDE για την ανάπτυξη web εφαρμογών. Αναπτύχθηκε και συντηρείται από την Microsoft. Επιλέχθηκε για το εύκολο UI του και την πληθώρα από προσθήκες (extensions) που διευκολύνουν την ανάπτυξη AngularJS/Bootstrap εφαρμογών.

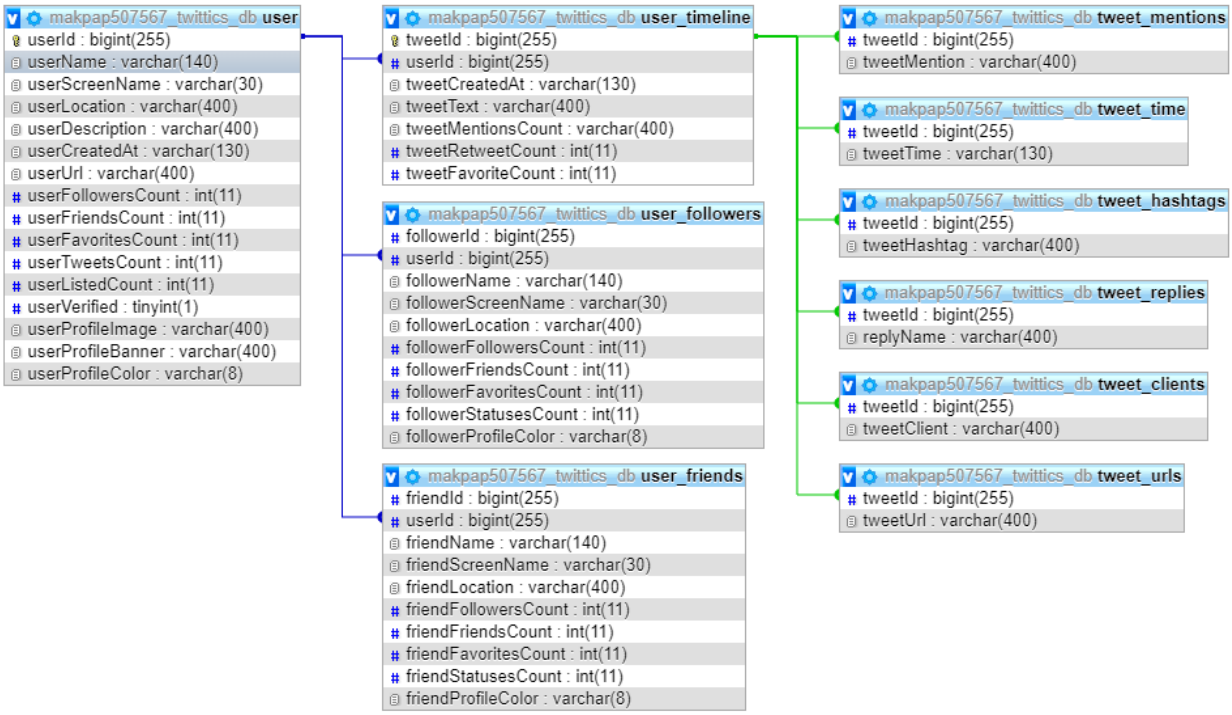
XAMPP

Το XAMPP είναι ένα ελεύθερο και ανοικτού κώδικα λογισμικό διαδικτυακών διακομιστών, αποτελούμενο κυρίως από τον διακομιστή Apache HTTP Server, τη βάση δεδομένων MariaDB και τους διερμηνείς για scripts γραμμένα στις γλώσσες προγραμματισμού PHP και Perl.

Το XAMPP επιλέχθηκε διότι είναι ο γνωστότερος HTTP Server που παράλληλα υποστηρίζει MySQL / MariaDB βάσεις δεδομένων, καθώς επίσης και PHP scripts. Ήταν ένα απαραίτητο εργαλείο για την ανάπτυξη και debugging της εφαρμογής.

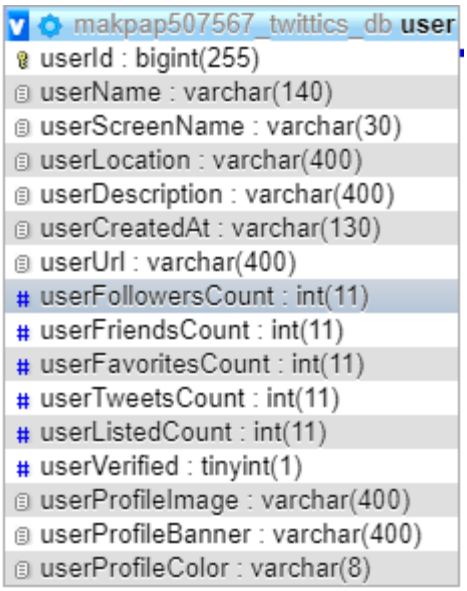
ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

Για την βάση δεδομένων της εφαρμογής μας, έγινε χρήση της MySQL. Η βάση μας είναι τύπου InnoDb με Collation utf8_general_ci και αποτελείται από 10 συσχετιζόμενους πίνακες. Οι συσχετίσεις των πινάκων χρησιμοποιούν συσχέτιση με λόγο πληθικότητας **1 - N (ένα-προς-πολλά)**. Για τον τύπο των πεδίων των πινάκων, ακολουθήθηκε η δομή της βάσης δεδομένων του Twitter (Twitter Database Schema). Η δομή της βάσης περιγράφεται στο διάγραμμα του σχήματος 1.



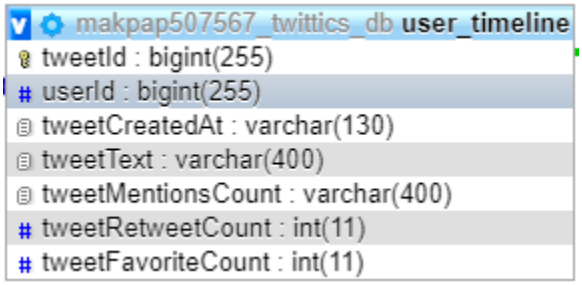
Σχήμα 1. Διάγραμμα δομής της βάσης δεδομένων

Από το παραπάνω σχήμα μπορούμε εύκολα να διακρίνουμε την δομή της βάσης και των πινάκων της. Ο πρωταρχικός μας πίνακας είναι ο “user”. Στον πίνακα “user” αποθηκεύονται οι γενικές και οι πιο βασικές πληροφορίες του χρήστη, δηλαδή μεταξύ άλλων, το userId, το user"name", το screen "name" κτλ. (βλέπε σχήμα 2.). Ο πίνακας “user” έχει ως πρωτεύον κλειδί (primary key) το πεδίο userId, ενώ παράλληλα χρησιμοποιείται ως ξένο κλειδί (foreign key) στους πίνακες “user_timeline”, “user_followers” και “user_friends”.



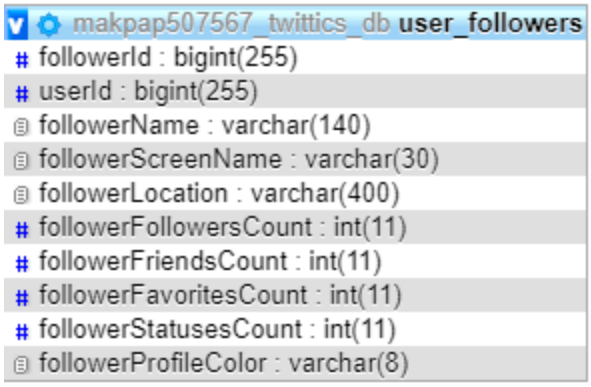
Σχήμα 2. Ο πίνακας "user"

Στην συνέχεια έχουμε τον πίνακα "user_timeline" όπου εκεί αποθηκεύονται οι βασικές πληροφορίες του χρήστη, σχετικά με την δραστηριότητά του, δηλαδή τα tweet και τα retweet (βλέπε σχήμα 3.). Ο πίνακας "user_timeline" έχει ως πρωτεύον κλειδί (primary key) το πεδίο "tweetId", Το πεδίο "tweetId" χρησιμοποιείται ως ξένο κλειδί (foreign key) στους πίνακες "tweet_mentions", "tweet_time", "tweet_hashtags", "tweet_replies", "tweet_clients", "tweet_urls".



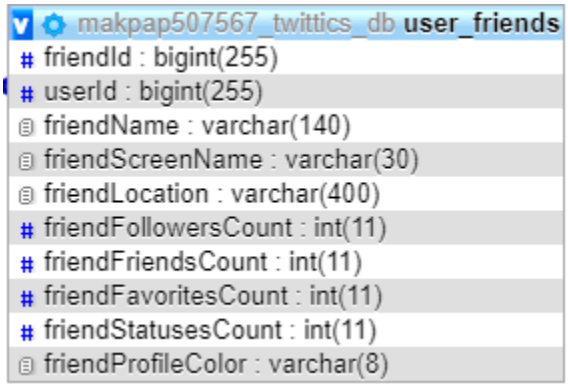
Σχήμα 3. Ο πίνακας "user_timeline"

Στον πίνακα "user_followers" αποθηκεύονται βασικές πληροφορίες που σχετίζονται με τους followers του χρήστη, όπως είναι το "name", "screenName", "location" κ.α (βλέπε σχήμα 4.). Ο πίνακας "user_followers" δεν έχει πρωτεύον κλειδί (primary key), χρησιμοποιεί ως ξένο κλειδί (foreign key) το userid του πίνακα "users".



Σχήμα 4. Ο πίνακας "user_followers"

Ο πίνακας "user_friends" περιλαμβάνει τις βασικές πληροφορίες των φίλων του χρήστη όπως είναι το "name", screenName, "location" κ.α. (βλέπε σχήμα 5.). Ο πίνακας "user_followers" δεν έχει πρωτεύον κλειδί (primary key), χρησιμοποιεί ως ξένο κλειδί (foreign key) το userid του πίνακα "users".



Σχήμα 5. Ο πίνακας "user_friends"

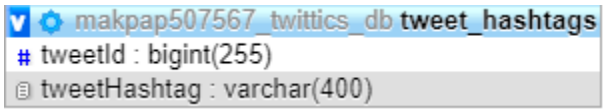
Επιπλέον έχουμε τον πίνακα "user_mentions", ο οποίος περιλαμβάνει τα mentions που έχει κάνει ο χρήστης (βλέπε σχήμα 6.). Ο πίνακας "user_mentions" δεν έχει πρωτεύον κλειδί (primary key), αλλά χρησιμοποιεί ως ξένο κλειδί (foreign key) το "tweetId" του πίνακα "user_timeline".

Σχήμα 6. Ο πίνακας "user_mentions"

Ο πίνακας “user_time”, περιλαμβάνει τις στιγμές τις οποίες ο χρήστης είναι δραστήριος (βλέπε σχήμα 7.). Ο πίνακας “user_time” δεν έχει πρωτεύον κλειδί (primary key), αλλά χρησιμοποιεί ως ξένο κλειδί (foreign key) το "tweetId" του πίνακα “user_timeline”.

Σχήμα 7. Ο πίνακας “user_time”

Ο πίνακας “user_hashtags”, ο οποίος περιλαμβάνει τα hashtags που έχει κάνει ο χρήστης (βλέπε σχήμα 8.). Ο πίνακας “user_hashtags” δεν έχει πρωτεύον κλειδί (primary key), αλλά χρησιμοποιεί ως ξένο κλειδί (foreign key) το "tweetId" του πίνακα “user_timeline”.



Σχήμα 8. Ο πίνακας “user_hashtags”

Στον πίνακα “user_replies”, αποθηκεύονται τα replies που έχει κάνει ο χρήστης (βλέπε σχήμα 9.). Ο πίνακας “user_replies” δεν έχει πρωτεύον κλειδί (primary key), αλλά χρησιμοποιεί ως ξένο κλειδί (foreign key) το "tweetId" του πίνακα “user_timeline”.

Σχήμα 9. Ο πίνακας “user_replies”

Στη συνέχεια έχουμε τον πίνακα “user_clients”, στον οποίο αποθηκεύονται τα clients που έχει κάνει ο χρήστης (βλέπε σχήμα 10.). Ο πίνακας “user_clients” δεν έχει πρωτεύον κλειδί (primary key), αλλά χρησιμοποιεί ως ξένο κλειδί (foreign key) το "tweetId" του πίνακα “user_timeline”.

Σχήμα 10. Ο πίνακας "user_clients"

Τέλος, στον πίνακα "user_urls", αποθηκεύονται τα urls που έχει χρησιμοποιήσει ο χρήστης (βλέπε σχήμα 11.). Ο πίνακας "user_urls" δεν έχει πρωτεύον κλειδί (primary key), αλλά χρησιμοποιεί ως ξένο κλειδί (foreign key) το "tweetId" του πίνακα "user_timeline".

Σχήμα 11. Ο πίνακας "user_urls"

ΑΡΧΙΚΗ (LANDING PAGE)

Το περιβάλλον της αρχικής σελίδας απαρτίζεται από το κουμπί της σύνδεσης (αναλύεται παρακάτω), από ένα κουμπί το οποίο ονομάζεται “μάθετε περισσότερα” όπου εκεί περιγράφεται το τι είναι το Twittics και πως δουλεύει, το ίδιο κουμπί υπάρχει και στο footer. Κεντρικά βλέπουμε μία εικόνα η οποία δημιουργήθηκε με την χρήση του εργαλείο σχεδιασμού Adobe XD. Επιπλέον στην αρχική σελίδα μπορεί ο χρήστης να δει, να καταλάβει και να κατανοήσει με τι ακριβώς ασχολείται η εφαρμογή, αφού γίνεται μία συνοπτική παρουσίαση στο τι πληροφορίες θα βρει κατά την περιήγηση του. Τέλος, έχουμε το footer, στο οποίο έχουμε πάλι το κουμπί “Learn more” και το κουμπί “Privacy”, καθώς και την επικοινωνία.

Τι πληροφορίες θα βρεις εδώ

Με μια ματιά

Βασικότερα, οι γενικές πληροφορίες του προφίλ σου. Ο αριθμός των tweets, ακολούθων, φίλων και likes. Όλα εύκολα και προσβάσιμα με μια γρήγορη ανάγνωση.

Στατιστικά

Όλα τα ενδιαφέροντα στατιστικά στοιχεία σχετικά με τα tweets σου. Αναφερόμαστε σχετικά με τους ανθρώπους που απαντάς περισσότερο, ποια είναι τα αγαπημένα σου hashtags και διάφορα άλλα θέματα ...

Σύγκριση

Σύγκρινε εύκολα τα στατιστικά στοιχεία σου με τα στατιστικά των ακολούθων σου και των ατόμων που ακολουθείς. Σύγκρινε σε κατηγορίες με τον αριθμό των tweets, των ακολούθων, των φίλων και των like, για να δεις ποιός έχει τον μεγαλύτερο αριθμό.

Δραστηριότητα

Έλεγξε τη δραστηριότητά σου στο πέρασμα των χρόνων. Μάθε αν έχεις γίνει πιο ενεργός ή όχι. Επίσης, έλεγξε τις πιο ενεργές ώρες σου μέσα στην ημέρα.

Συνδέσου με το Twitter προφίλ σου για να ξεκινήσεις!

Twittics

More

Learn more

Privacy

Contact

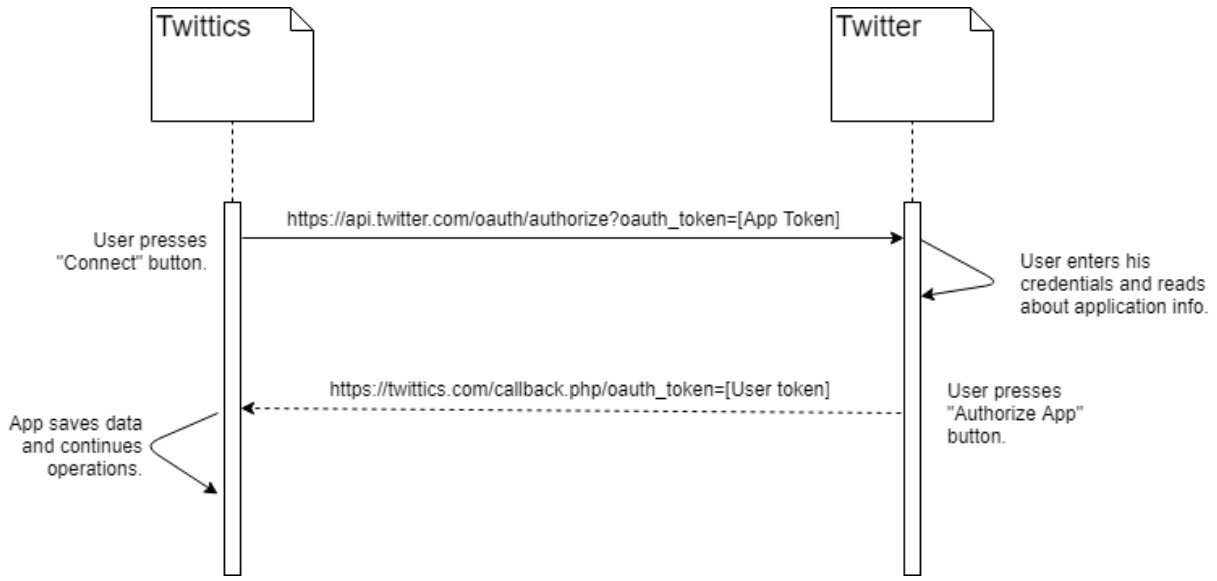
makisp@twittics.com

iwannap@twittics.com

ΣΥΝΔΕΣΗ

Για την σύνδεση του χρήστη με την εφαρμογή, γίνεται η παρακάτω διαδικασία. Ο χρήστης πατάει το κουμπί “connect”, το οποίο τον παραπέμπει στο site του twitter ώστε να μπορέσει να αποδεχθεί την εφαρμογή και να συνδέσει τον λογαριασμό του. Συνδέεται με τα στοιχεία του λογαριασμού του στο twitter, και αφού διαβάσει τις πληροφορίες της εφαρμογής, αποδέχεται τους όρους και στη συνέχεια πατάει το κουμπί “Authorize App. Στη συνέχεια το Twitter επιστρέφει τον χρήστη πίσω στην εφαρμογή μας (https://www.twittics.com). Το Twitter μαζί με την ανακατεύθυνση μας παρέχει ένα αλφαριθμητικό κλειδί (token), το οποίο είναι μοναδικό για κάθε χρήστη και θα επιτρέψει στην εφαρμογή - μαζί με το token της ίδιας της εφαρμογής - να κάνει τα απαραίτητα request για να πάρει τις απαιτούμενες πληροφορίες από το προφίλ του χρήστη. Με αυτό τον τρόπο η εφαρμογή θα μπορέσει αργότερα να παρουσιάσει στον χρήστη τα επεξεργασμένα δεδομένα του.

Στο παρακάτω διάγραμμα μπορούμε να δούμε ενδεικτικά την επικοινωνία μεταξύ της εφαρμογής και του Twitter.



Σχήμα 12. Διάγραμμα επικοινωνίας κατά την σύνδεση

ΑΠΟΘΗΚΕΥΣΗ ΚΑΙ ΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ

Παρακάτω θα δούμε την διαδικασία αποθήκευσης και επεξεργασία των δεδομένων του χρήστη, καθώς και τα request που χρειάζονται για να αποκτήσουμε πρόσβαση σε αυτά. Όλα τα request για την πρόσβαση των δεδομένων, γίνονται με την μέθοδο GET. Μετά το κάθε request, το Twitter απαντάει με ένα response σε μορφή JSON, το οποίο περιέχει όλες τα διαθέσιμα δεδομένα σχετικά με το request που κάναμε νωρίτερα. Στο σχήμα 13 μπορούμε να δούμε ένα παράδειγμα της δομής του JSON ενός response.

Πριν κάνουμε τα requests, αρχικοποιούμε την σύνδεση (oAuth) με το Twitter, γράφοντας τον παρακάτω κώδικα:

```
$connection = new TwitterOAuth(CONSUMER_KEY, CONSUMER_SECRET, $access_token['oauth_token'], $access_token['oauth_token_secret']);
```


```
{
    "created_at": "Sat May 09 17:58:22 +0000 2009",
    "status": {
        "created_at": "Tue Aug 28 05:44:24 +0000 2012"
        "geo": {
            "coordinates": [
                37.76484123,
                -122.45037293
            ],
            "type": "Point"
        },
        "id": 240323931419062272,
        "in_reply_to_screen_name": "messl",
        "in_reply_to_status_id": 240316959173009410,
        "in_reply_to_status_id_str": "240316959173009410",
        "in_reply_to_user_id": 18707866,
        "place": {
            "bounding_box": {
                "coordinates": [
                    [
                        [
                            -122.45778216,
                            37.75932999
                        ]
                    ]
                ],
                "type": "Polygon"
            },
            "country": "United States",
            "name": "Ashbury Heights"
        },
        "source": "Twitter for  iPhone",
        "text": "@messl congrats! So happy for all 3 of you."
    }
}
```


Σχήμα 13. Twitter response JSON structure

ΓΕΝΙΚΑ ΣΤΟΙΧΕΙΑ ΧΡΗΣΤΗ

Στα γενικά στοιχεία του χρήστη ανήκουν οι βασικές και πιο γενικές πληροφορίες του προφίλ του, όπως για παράδειγμα, το όνομα του, το ψευδώνυμο του, τοποθεσία, περιγραφή καθώς και τον συνολικό αριθμό από followers, following, tweets και αγαπημένων.

Για να μπορέσουμε να πάρουμε τα δεδομένα του χρήστη, πρέπει να κάνουμε το κατάλληλο request στο Twitter. Στην περίπτωση αυτή το request είναι το account/verify_credentials και πραγματοποιείται με τον παρακάτω κώδικα:

```
$user = $connection->get("account/verify_credentials");
```

Μετά την πρόσβαση μας στα δεδομένα, τα αποθηκεύουμε στην βάση δεδομένων μας και συγκεκριμένα στον πίνακα “user”. Τα συγκεκριμένα δεδομένα δεν χρειάζονται κάποια περαιτέρω επεξεργασία, οπότε μπορούμε να προχωρήσουμε με την αποθήκευση, με το SQL ερώτημα (SQL Query) του σχήματος 14.

```
$sqlUserCredentials = "REPLACE INTO `user`
(userId, userName, userScreenName, userLocation, userDescription, userCreatedAt, userUrl,
userFollowersCount, userFriendsCount, userFavoritesCount, userTweetsCount,
userListedCount, userVerified, userProfileImage, userProfileBanner, userProfileColor)
      VALUES
('$user->id', '".mysqli_real_escape_string($conn, $user->name)."',
'".mysqli_real_escape_string($conn, $user->screen_name)."',
'".mysqli_real_escape_string($conn, $user->location)."', '".mysqli_real_escape_string($conn,
$user->description)."', '$user->created_at', '$user->url', $user->followers_count, $user-
>friends_count, $user->favourites_count, $user->statuses_count, $user->listed_count, $verified,
'$profilePicture', '$userProfileBanner', '$user->profile_link_color')";
```

Σχήμα 14. SQL ερώτημα για αποθήκευση γενικών δεδομένων του χρήστη

Μετά την επιτυχημένη αποθήκευση των δεδομένων, θα έχουμε τα δεδομένα καταχωρημένα στην βάση με την μορφή του σχήματος 15.

userId	userName	userScreenName	userLocation	userDescription	userCreatedAt	userUrl
2176987173	Makis Papas	makisp_	Greece	Web Developer and a technology geek. Love watching...	Mon Nov 11 14:37:38 +0000 2013	https://t.co/shLKhB25RT

userFollowersCount	userFriendsCount	userFavoritesCount	userTweetsCount	userListedCount	userVerified
11	31	184	55	0	0

userProfileImage	userProfileBanner	userProfileColor
https://pbs.twimg.com/profile_images/9981380059854...	https://pbs.twimg.com/profile_banners/2176987173/1...	78E087

Σχήμα 15. Αποθηκευμένα δεδομένα του πίνακα "user" της βάσης

TWEETS

Η κατηγορία Tweets αναφέρεται στα posts (tweets) που ανάρτησε ο χρήστης. Το συγκεκριμένο request φέρνει τόσο τα tweet που ο χρήστης μόνος του έχει κάνει καθώς και τις απαντήσεις (retweets) σε άλλους χρήστες. Η απάντηση (response) σε αυτό το request, επιστρέφει όλα τα διαθέσιμα posts μαζί με όλες τις απαραίτητες πληροφορίες (text, hashtags, mentions, urls, date κτλ.). Το συγκεκριμένο request είναι το statuses/user_timeline. Ο κώδικας για την πραγματοποίηση του είναι ο ακόλουθος:

```
$timeline = $connection->get("statuses/user_timeline", ["id" => $user->id, "include_rts" => true, "count" => 200]);
```

Στη συνέχεια, μέσα από μια επανάληψη, αποθηκεύουμε στην βάση το κάθε post (tweet) ξεχωριστά μαζί με τις απαιτούμενες πληροφορίες σχετικά με αυτό. Στο σχήμα 16 βρίσκεται ο αναφερόμενος κώδικας.

```
$sqlUserTimeline = "REPLACE INTO `user_timeline`
(tweetId, userId, tweetCreatedAt, tweetText, tweetMentionsCount, tweetRetweetCount, tweetFavoriteCount)
        VALUES
('$tweetId', $user->id, '$tweetCreatedAt', '".mysqli_real_escape_string($conn, $tweetText)."', $tweetMentionsCount, $retweetCount, $favoriteCount)";
```

Σχήμα 16. Κώδικας αποθήκευσης των tweets στη βάση.

Μέσα στην επανάληψη, μαζί με την αποθήκευση του ίδιου του tweet, ξεχωρίζουμε με κώδικα, διάφορα στατιστικά και στοιχεία για αυτό. Πιο συγκεκριμένα βρίσκουμε εάν είναι απάντηση (retweet), εάν περιέχει URLs, εάν έχει mention σε χρήστη/ες, εάν έχει hashtags, την συσκευή (client) στην οποία γράφτηκε καθώς και την συγκεκριμένη ώρα που ανέβηκε online. Παρακάτω βρίσκονται οι αντίστοιχοι κώδικες για την επεξεργασία και αποθήκευση των παραπάνω δεδομένων.

REPLIES

Για την εύρεση των replies (retweets) μέσα στα tweets του χρήστη, αρκεί να αναζητήσουμε τα tweets που ξεκινάνε με την λέξη “RT”. Μόλις βρούμε ένα retweet, συγκρατούμε ένα συγκεκριμένο τμήμα του κειμένου του, το οποίο είναι το όνομα του χρήστη στον οποίο απαντήσαμε. Ύστερα, αποθηκεύουμε το συγκεκριμένο όνομα μέσα στον αντίστοιχο πίνακα της βάσης.

```
if(count($tweetText) != 0) {
        for ($j = 0; $j < count($tweetText); $j++) {
            if (startsWith($tweetText, 'RT')) {
                $name = substr($tweetText, 4, (stripos($tweetText, ":") - 3));

                $sqlTweetReply = "REPLACE INTO `tweet_replies`
                                  (tweetId,replyName)
                                  VALUES
                                  ('$tweetId', '$name')";

                if ($conn->query($sqlTweetReply) === TRUE) {
                //echo "New tweet info added.";
                } else {
                    //echo "Error: " . $sqlTweetReply . "<br>" . $conn->error;
                    errorLog("Error: " . $sqlTweetReply . " " . $conn->error);
                }
            }
        }
    }
```

Σχήμα 17. Κώδικας εύρεσης και αποθήκευσης των replies στην βάση

URLS

Τα URLs ενός tweet έχουν συγκεκριμένη θέση στο αντικείμενο του response. Πιο συγκεκριμένα βρίσκονται μέσα στο “entities->urls”. Για την εύρεση λοιπόν όλων των διαθέσιμων URL ενός tweet, αρκεί να περάσουμε με μια επανάληψη από όλα τα “urls” του αντικειμένου “entities”. Τέλος κρατάμε το κομμάτι του URL που χρειαζόμαστε (μόνο το domain name) και τα αποθηκεύουμε στον πίνακα της βάσης που του αντιστοιχεί.

```
if(count($timeline[$i]->entities->urls) != 0) {
   for ($j = 0; $j < count($timeline[$i]->entities->urls); $j++) {
      $initialTweetUrl = $timeline[$i]->entities->urls[$j]->expanded_url;
      $urlLastPos = strPosX($initialTweetUrl, '/', 3);
      $tweetUrl = substr($initialTweetUrl, 0, ($urlLastPos + 1));

      $sqlTweetUrl = "REPLACE INTO `tweet_urls`
                     (tweetId, tweetUrl)
                     VALUES
                     ('$tweetId', '$tweetUrl')";

      if ($conn->query($sqlTweetUrl) === TRUE) {
         //echo "New tweet info added.";
      } else {
         //echo "Error: " . $sqlTweetUrl . "<br>" . $conn->error;
         errorLog("Error: " . $sqlTweetUrl . " " . $conn->error);
      }
   }
}
```

Σχήμα 18. Κώδικας εύρεσης και αποθήκευσης των urls στην βάση

MENTIONS

Τα mentions ενός tweet είναι σε ένα συγκεκριμένο πίνακα. Βρίσκονται μέσα στο “entities->user_mentions”. Για να εντοπίσουμε όλα τα mentions που είναι διαθέσιμα πρέπει με μία επανάληψη να περάσουμε από όλα τα “mentions” του αντικειμένου “entities”.

```
if(count($timeline[$i]->entities->user_mentions) != 0) {
   for ($j = 0; $j < count($timeline[$i]->entities->user_mentions); $j++) {
      $tweetMention = $timeline[$i]->entities->user_mentions[$j]->screen_name;

      $sqlTweetMention = "REPLACE INTO `tweet_mentions`
                          (tweetId, tweetMention)
                          VALUES
                          ('$tweetId', '$tweetMention')";

      if ($conn->query($sqlTweetMention) === TRUE) {
         //echo "New tweet info added.";
      } else {
         //echo "Error: " . $sqlTweetMention . "<br>" . $conn->error;
         errorLog("Error: " . $sqlTweetMention . " " . $conn->error);
      }
   }
}
```

.Σχήμα 19. Κώδικας εύρεσης και αποθήκευσης των mentions στην βάση

HASHTAGS

Τα hashtags ενός tweet ομοίως με τα urls και τα mentions βρίσκονται σε συγκεκριμένη θέση. Συγκεκριμένα τα hashtags βρίσκονται μέσα μέσα στο “entities->hashtags”. Για να εντοπίσουμε όλα τα hashtags που είναι διαθέσιμα πρέπει με μία επανάληψη να περάσουμε από όλα τα “hashtags” του αντικειμένου “entities”.

```
if(count($timeline[$i]->entities->hashtags) != 0) {
   for ($j = 0; $j < count($timeline[$i]->entities->hashtags); $j++) {
      $tweetHashtag = $timeline[$i]->entities->hashtags[$j]->text;

      $sqlTweetHashtag = "REPLACE INTO `tweet_hashtags`
                          (tweetId, tweetHashtag)
                          VALUES
                          ('$tweetId', '$tweetHashtag')";

      if ($conn->query($sqlTweetHashtag) === TRUE) {
         //echo "New tweet info added.";
      } else {
         //echo "Error: " . $sqlTweetHashtag . "<br>" . $conn->error;
         errorLog("Error: " . $sqlTweetHashtag . " " . $conn->error);
      }
   }
}
```

Σχήμα 20. Κώδικας εύρεσης και αποθήκευσης των hashtags στην βάση

SOURCE

Ο client που χρησιμοποιήθηκε για το κάθε tweet βρίσκεται σε ένα συγκεκριμένο πεδίο του response, το “source”. Το μόνο που έχουμε να κάνουμε είναι να κόψουμε κάποιες περιττές πληροφορίες και να το αποθηκεύσουμε στην βάση δεδομένων στον αντίστοιχο πίνακα των tweet sources.

```
$sqlTweetSource = "REPLACE INTO `tweet_clients`
                   (tweetId, tweetClient)
                   VALUES
                   ('$tweetId', '$finalSource')";

if ($conn->query($sqlTweetSource) === TRUE) {
   //echo "New tweet info added.";
} else {
   //echo "Error: " . $sqlTweetSource . "<br>" . $conn->error;
   errorLog("Error: " . $sqlTweetSource . " " . $conn->error);
}
```

Σχήμα 21. Κώδικας εύρεσης και αποθήκευσης της πηγής στην βάση

TIME

Ο χρόνος που ανέβηκε το κάθε tweet, βρίσκεται και αυτός σε ένα συγκεκριμένο πεδίο του response, το “created_at”. Επειδή όμως το Twitter έχει μια δικιά του μορφή ώρας, πρέπει να την μετατρέψουμε σε μια global μορφή που θα την καταλάβει αργότερα η Javascript μέσα στην ιστοσελίδας μας. Μόλις γίνει η σωστή μετατροπή, αποθηκεύουμε την κάθε ώρα μέσα στην βάση.

```
$tweetParsedDate = date_parse($tweetCreatedAt);

$finalHour = $tweetParsedDate['hour'];
if ($finalHour == 0) {
   $finalHour = 24;
}
if($tweetParsedDate['minute'] > 29) {
   $finalHour++;
}
$finalTime = $finalHour;

$sqlTweetTime = "REPLACE INTO `tweet_time`
                 (tweetId, tweetTime)
                 VALUES
                 ('$tweetId', '$finalTime')";
```

Σχήμα 22. Κώδικας εύρεσης και αποθήκευσης του χρόνου που έγινε το tweet στην βάση

FRIENDS

Σε αυτή την κατηγορία κάνουμε το request για να πάρουμε όλους τους φίλους (Following/Friends) του συνδεδεμένου χρήστη. Το συγκεκριμένο request μας επιτρέπει να ζητάμε 200 φίλους κάθε φορά και σε κάθε response μας επιστρέφει και ένα cursor το οποίο χρησιμοποιούμε για να ζητήσουμε τους επόμενους 200 φίλους (αν και εφόσον υπάρχουν). Για τον λόγο αυτό αναπτύξαμε έναν αλγόριθμο που διαβάζει το εκάστοτε cursor και κάνει όσα requests χρειαστεί, έως ότου πάρουμε όλους τους φίλους του χρήστη. Το αναφερόμενο request είναι το friends/list. Ο κώδικας του είναι ο παρακάτω:

```
$friends = $connection->get("friends/list", ["id" => $user->id, "count" => 200]);
```

Ο αλγόριθμος που είναι υπεύθυνος για να κάνει όλα τα απαραίτητα requests των φίλων φαίνεται στο σχήμα 23 παρακάτω.

```
$friendsCursor = $friends->next_cursor;
do {
   errorLog('setData.php: Getting more friends with cursor: ' . (string) $friendsCursor);

   $tempFriends = $connection->get("friends/list", ["id" => $user->id, "cursor" => $friendsCursor,
"count" => 200]);
   $friendsCursor = $tempFriends->next_cursor;
   $friends = (object) array_merge_recursive((array) $friends, (array) $tempFriends);

   errorLog('setData.php: Next cursor for friends will be: ' . (string) $friendsCursor);
} while ($friendsCursor > 0);
```

Σχήμα 23. Request των φίλων

Για την αποθήκευση των φίλων στη βάση, αρκεί μια επανάληψη για όλο τον αριθμό των φίλων ώστε να μπορέσουμε να κρατήσουμε τις απαραίτητες πληροφορίες για τον κάθε φίλο. Ο κώδικας για την παραπάνω διαδικασία περιγράφεται στο σχήμα 24.

```
for ($i = 0; $i < count($friends->users); $i++) {
   $friendId = $friends->users[$i]->id;
   $friendName = $friends->users[$i]->name;
   $friendScreenName = $friends->users[$i]->screen_name;
   $friendLocation = $friends->users[$i]->location;
   $friendFollowersCount = $friends->users[$i]->followers_count;
   $friendFriendsCount = $friends->users[$i]->friends_count;
   $friendFavoritesCount = $friends->users[$i]->favourites_count;
   $friendStatusesCount = $friends->users[$i]->statuses_count;
   $friendProfileColor = $friends->users[$i]->profile_link_color;

   $sqlUserFriends = "REPLACE INTO `user_friends` (friendId ,userId, friendName,
friendScreenName, friendLocation, friendFollowersCount, friendFriendsCount,
friendFavoritesCount, friendStatusesCount, friendProfileColor)
                        VALUES
('$friendId', $user->id, '".mysqli_real_escape_string($conn, $friendName)."',
'".mysqli_real_escape_string($conn, $friendScreenName)."',
'".mysqli_real_escape_string($conn, $friendLocation)."', $friendFollowersCount,
$friendFriendsCount, $friendFavoritesCount, $friendStatusesCount, '$friendProfileColor')";
```

Σχήμα 24. Αποθήκευση των request των φίλων στην βάση

FOLLOWERS

Στην συνέχεια έχουμε την κατηγορία Followers, στην κατηγορία αυτή κάνουμε ένα request για να πάρουμε όλους τους ακολούθους του χρήστη. Όπως και στην κατηγορία friends το ίδιο και στην κατηγορία followers το request που κάνουμε μας επιτρέπει να ζητάμε 200 ακολούθους κάθε φορά και σε κάθε response μας επιστρέφει ένα cursor το οποίο χρησιμοποιούμε για να ζητήσουμε τους επόμενους 200 ακολούθους.(αν υπάρχουν). Έτσι δημιουργήσαμε έναν αλγόριθμο που διαβάζει το εκάστοτε cursor και κάνει όσα requests χρειαστεί, μέχρις ότου πάρουμε όλους τους ακολούθους του χρήστη. Το αναφερόμενο request είναι το followers/list. Παρακάτω επισυνάπτεται ο κώδικας:

```
$followers = $connection->get("followers/list", ["id" => $user->id, "count" => 200]);
```

Τα απαραίτητα request των followers γίνονται μέσω του παρακάτω αλγορίθμου και φαίνονται στο σχήμα 25.

```
$followersCursor = $followers->next_cursor;
    do {
        errorLog('setData.php: Getting more followers with cursor: ' . (string) $followersCursor);

        $tempFollowers = $connection->get("followers/list", ["id" => $user->id, "cursor" =>
$followersCursor, "count" => 200]);
        $followersCursor = $tempFollowers->next_cursor;
        $followers = (object) array_merge_recursive((array) $followers, (array) $tempFollowers);

        errorLog('setData.php: Next cursor for followers will be: ' . (string) $followersCursor);
    } while ($followersCursor > 0);

    if($user->verified == true) {
        $verified = 1;
    } else {
        $verified = 0;
    }
```

Σχήμα 25. Request των ακολούθων

Η αποθήκευση των ακολούθων στη βάση γίνεται με μία επανάληψη όπου κρατάμε όλες τις απαραίτητες πληροφορίες για τον κάθε φίλο. Ο κώδικας για την παραπάνω διαδικασία περιγράφεται στο σχήμα 26.

```
for ($i = 0; $i < count($followers->users); $i++) {
        $followerId = $followers->users[$i]->id;
        $followerName = $followers->users[$i]->name;
        $followerScreenName = $followers->users[$i]->screen_name;
        $followerLocation = $followers->users[$i]->location;
        $followerFollowersCount = $followers->users[$i]->followers_count;
        $followerFriendsCount = $followers->users[$i]->friends_count;
        $followerFavoritesCount = $followers->users[$i]->favourites_count;
        $followerStatusesCount = $followers->users[$i]->statuses_count;
        $followerProfileColor = $followers->users[$i]->profile_link_color;

        $sqlUserFollowers = "REPLACE INTO `user_followers`
(followerId ,userId, followerName, followerScreenName, followerLocation,
followerFollowersCount, followerFriendsCount, followerFavoritesCount, followerStatusesCount,
followerProfileColor)
                                VALUES
('$followerId', $user->id, '".mysqli_real_escape_string($conn, $followerName)."',
'".mysqli_real_escape_string($conn, $followerScreenName)."',
'".mysqli_real_escape_string($conn, $followerLocation)."', $followerFollowersCount,
$followerFriendsCount, $followerFavoritesCount, $followerStatusesCount,
'$followerProfileColor')";

        if ($conn->query($sqlUserFollowers) === TRUE) {
            //echo "New tweet info added.";
        } else {
           //echo "Error: " . $sqlUserFollowers . "<br>" . $conn->error;
           errorLog("Error: " . $sqlUserFollowers . " " . $conn->error);
        }
    }
```

Σχήμα 26. Αποθήκευση των request των ακολούθων στην βάση

ΑΝΑΠΑΡΑΣΤΑΣΗ ΔΕΔΟΜΕΝΩΝ

Η αναπαράσταση των δεδομένων στον χρήστη είναι ίσως το σημαντικότερο κομμάτι της εφαρμογής, διότι είναι αυτό που περιμένει ο χρήστης να δει, μετά την σύνδεση με τον λογαριασμό του. Τα δεδομένα πρέπει να παρουσιάζονται με εύκολο και κατανοητό τρόπο, δίχως να κουράζουν ή να μπερδεύουν τον χρήστη, ενώ παράλληλα πρέπει να είναι επαρκή ώστε να τον καλύπτουν σε οτιδήποτε περιμένει να βρει.

Γι' αυτό το λόγο, το Twittics χρησιμοποιεί γραφήματα, ενώ ταυτόχρονα χωρίζει τα δεδομένα σε ενότητες ώστε ο χρήστης να μπορεί εύκολα και γρήγορα να βρει ακριβώς αυτό που ψάχνει. Κάθε δεδομένο βρίσκεται μέσα σε μια ξεχωριστή κάρτα και επικρατεί κατακόρυφη στοίχιση και scrolling. Οι ενότητες αυτές είναι τέσσερις και ονομάζονται (με την σειρά που εμφανίζονται: **In a glance (Με μια ματιά), Statistics (Στατιστικά), Activity (Δραστηριότητα)** και **Compare (Σύγκριση)**.

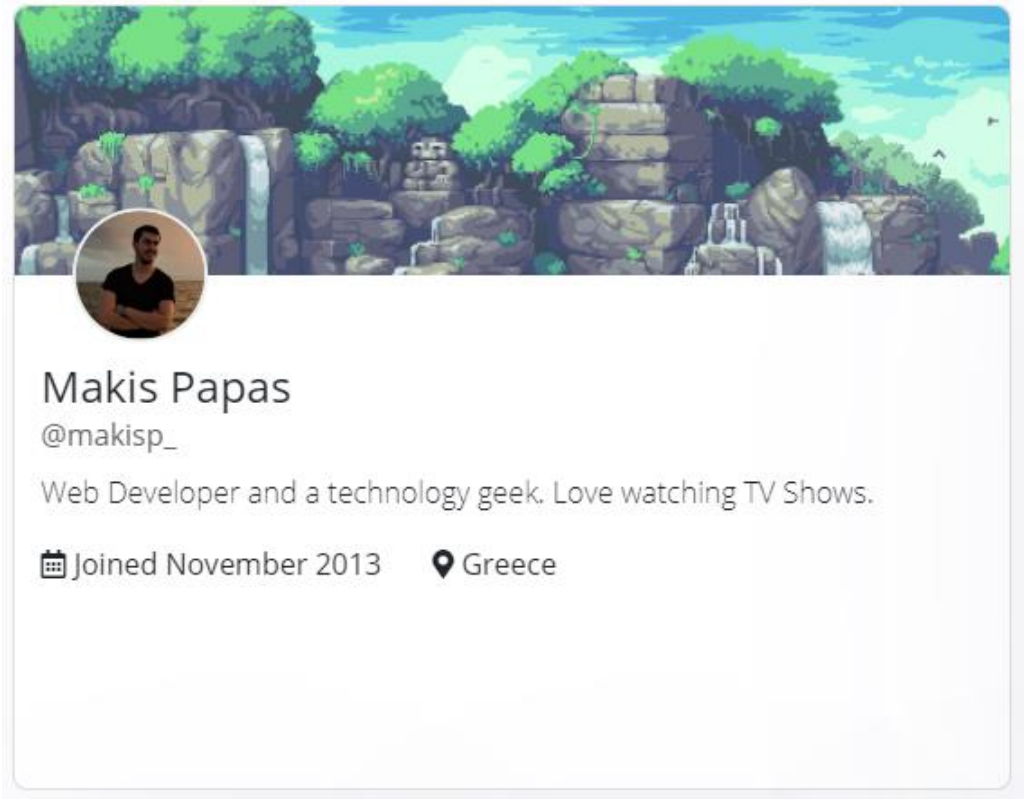
Παρακάτω θα δούμε αναλυτικά κάθε ενότητα, καθώς και τι δεδομένα παρουσιάζει.

IN A GLANCE (ΜΕ ΜΙΑ ΜΑΤΙΑ)

Στην κατηγορία "In a glance" παρουσιάζουμε στον χρήστη τα πιο γενικά στοιχεία του προφίλ του. Τα δεδομένα χωρίζονται συνολικά σε 7 διαφορετικές κάρτες.

ΒΑΣΙΚΑ ΣΤΟΙΧΕΙΑ ΤΟΥ ΧΡΗΣΤΗ

Στην πρώτη κάρτα εμφανίζουμε τα προσωπικά του στοιχεία. Αυτά είναι το **όνομα του** και το **ψευδώνυμο του**, η **περιγραφή του**, η **ημερομηνία εγγραφής του στο Twitter**, η **τοποθεσία του**, **το σήμα "verified"** (εφόσον ισχύει), η **εικόνα προφίλ του** καθώς και η **εικόνα (banner)** του προφίλ του. Από την συγκεκριμένη κάρτα ο χρήστης έχει μία συνοπτική εικόνα με τα βασικά στοιχεια του προφίλ του. Στο σχήμα 27 μπορείτε να δείτε ένα παράδειγμα της πρώτης κάρτας αυτής της κατηγορίας.



Σχήμα 27. Βασικά στοιχεία του χρήστη

Για το γέμισμα αυτής της κάρτας με δεδομένα χρειάζεται να κάνουμε ένα απλό request στον server μας για να πάρουμε τα δεδομένα του συνδεδεμένου χρήστη. Για αυτό το σκοπό, κάνουμε το παρακάτω http request, με την χρήση της JavaScript:

```
getOwnData: function () {
    var deferred = $q.defer();
    var self = this;
    this.mainRequestMethod('http://localhost/twanalytics/api/getData/getOwnData.php',
'ownData').then(function (resp) {
        if (resp.length > 0) {
            deferred.resolve(resp);
        } else {
            deferred.resolve('error');
        }

    });
    return deferred.promise;
}
```

Κάνοντας το παραπάνω request στον server μας, αυτός με την σειρά του θα στείλει ένα SQL ερώτημα (SQL Query) στην βάση ώστε να πάρει τα δεδομένα και ύστερα θα βάλει τα απαραίτητα δεδομένα σε ένα JSON object και θα τα επιστρέψει στην εφαρμογή.

Μπορείτε να δείτε το αναφερόμενο SQL ερώτημα στον παρακάτω κώδικα:

```
$sqlgetOwnData = "SELECT *
                  FROM `user`
                  WHERE userId = ".$_SESSION['user_id'];

$result = $conn->query($sqlgetOwnData);
```

Μόλις η εφαρμογή λάβει την απάντηση (response) με τα δεδομένα, το μόνο που έχει να κάνει είναι να κάνει parse την **ημερομηνία εγγραφής** σε JavaScript Date object ώστε να μπορέσει να τυπωθεί με σωστή σύνταξη πάνω στην κάρτα. Τα υπόλοιπα δεδομένα είναι έτοιμα, οπότε τα τυπώνουμε απευθείας χωρίς κάποια επεξεργασία.

ΑΝΑΛΟΓΙΑ FOLLOWERS/FOLLOWING

Στην παρακάτω κάρτα παρατηρούμε την αναλογία follower/following. Εδώ ο χρήστης βλέπει σε γενικό βαθμό αν οι followers του είναι περισσότεροι, λιγότεροι ή ίδιοι με τους following. Αυτό που επιθυμούν οι χρήστες είναι να έχουν περισσότερους ακόλουθους, αυτή η κάρτα βοηθάει με μία γρήγορη ματιά να καταλάβει ο χρήστης εάν έχει το επιθυμητό αποτέλεσμα/αναλογία. Παρακάτω βλέπουμε το σχήμα 28 με περισσότερες πληροφορίες της κάρτας.

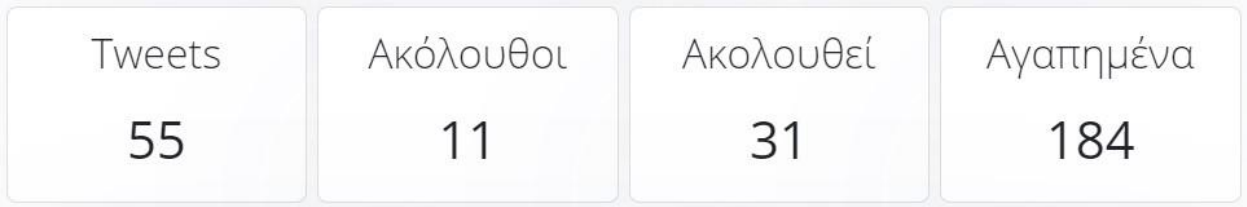
Σχήμα 28. Αναλογία followers/following

Για την υλοποίηση αυτής της κάρτας χρειαζόμαστε το αποτέλεσμα της διαίρεσης μεταξύ Follower και Following. Τις πληροφορίες αυτές τις έχουμε ήδη έτοιμες από το request της πρώτης κάρτας, οπότε ο κώδικας είναι μόνο ο εξής:

```
$scope.ratio = ($scope.ownData.userFollowersCount /
$scope.ownData.userFriendsCount).toFixed(2);
```

ΣΥΝΟΛΙΚΑ ΤΑ TWEETS - FOLLOWERS - FOLLOWING - LIKES

Στην συνέχεια βλέπουμε τις κάρτες **Tweets - Ακόλουθοι - Ακολουθείτε - Αγαπημένα.** Οι κάρτες αυτές με έναν αριθμό μας δίνουν ότι πληροφορία χρειαζόμαστε, εν συντομία, για τα συγκεκριμένα δεδομένα. Τα δεδομένα αυτά βοηθούν τον κάθε χρήστη να έχει μια γρήγορη αντίληψη στο τι συμβαίνει στο προφίλ του, δηλαδή μπορεί να δει συνολικά πόσα tweets, followers, following και likes έχει. Οι καρτέλες αυτές φαίνονται στο παρακάτω σχήμα 29.



Σχήμα 29. Συνολικά τα tweets - followers - following - likes

Οι παραπάνω κάρτες δεν χρειάζονται κάποια επιπλέον ενέργεια για την εμφάνισή τους. Οι πληροφορίες αυτές είναι και πάλι διαθέσιμες από το request της πρώτης κάρτας, καθώς όλες αυτές οι πληροφορίες αφορούν την δραστηριότητα του ίδιου του χρήστη.

TWEETS ΧΡΗΣΤΗ ΣΤΟ ΠΕΡΑΣΜΑ ΤΩΝ ΧΡΟΝΩΝ

Στην τελευταία καρτέλα βλέπουμε την αναπαράσταση ενός διαγράμματος. Το διάγραμμα αυτό δείχνει στο χρήστη την δραστηριότητα του στο Twitter, δηλαδή τον χρόνο που είναι περισσότερο ενεργός, πόσα tweets έχει κάνει τα παλαιότερα χρόνια, συγκριτικά μεταξύ τους, αλλά ταυτόχρονα και το πόσο δραστήριος είναι στο παρόν. Το να έχει ο χρήστης μία γενική στατιστική άποψη για την δραστηριότητα του στο Twitter είναι καλό για δύο λόγους, ο πρώτος είναι γιατί μπορεί να γνωρίζει και να ελέγχει την δραστηριότητα του στα social media, δεύτερον μπορεί να ερευνήσει ποιες είναι οι ώρες αιχμής στο Twitter και να αρχίσει να δραστηριοποιείται και ο ίδιος εκείνες τις ώρες. Ο δεύτερος λόγος είναι σημαντικός για την δημοσιότητα του χρήστη και την αύξηση των ακολούθων του. Στην συνέχεια βλέπουμε το διάγραμμα στο σχήμα 30.

Σχήμα 30. Tweets χρήστη στο πέρασμα των χρόνων

Η παραπάνω κάρτα χρειάζεται φυσικά μια περαιτέρω επεξεργασία για να υλοποιηθεί. Αρχικά μέσω μιας μεθόδου, βρίσκουμε όλες τις διαφορετικές χρονιές στις οποίες έχει γράψει ο χρήστης κάποιο tweet. Ύστερα βρίσκουμε τα χρόνια που πέρασαν από την δημιουργία του προφίλ του χρήστη και τα βάζουμε σε μια μεταβλητή (object). Με αυτό τον τρόπο θα έχουμε τον αριθμό των tweet που έκανε ο χρήστης σε κάθε χρονιά. Παρακάτω μπορούμε να δούμε την αναφερόμενη μέθοδο.

```
var findYearlyTweets = function (tweetsData, ownData) {

    // Create new date objects
    var userCreated = new Date(Date.parse(ownData.userCreatedAt)).getFullYear();
    var currentYear = new Date().getFullYear();
    var yearsPast = currentYear - userCreated;
    var tweetData = [];

    for (var i = yearsPast; i >= 0; i--) {
        tweetData.push({
            'year': currentYear - i,
            'color': '#' + ownData.userProfileColor
        });
    }

    var tweetYears = [];
    _.map(tweetsData, function (tweet) {
        tweetYears.push(new Date(Date.parse(tweet.tweetCreatedAt)).getFullYear());
    });

    for (var i = 0; i < tweetData.length; i++) {
        for (var y = 0; y < tweetYears.length; y++) {
            if (tweetYears[y] == tweetData[i].year) {
                tweetData[i].count = !tweetData[i].count ? 1 : tweetData[i].count + 1;
            }
        }
    }

    // Create yearly tweets chart.
    chartService.createLineChart('yearly-tweets-chart', tweetData);
}
```

Τέλος, για την δημιουργία του γραφήματος, δίνουμε το παραπάνω object στο “chartService”, που είναι υπεύθυνο για τα γραφήματα.

STATISTICS (ΣΤΑΤΙΣΤΙΚΑ)

Η κατηγορία **“Στατιστικά”** περιλαμβάνει πληροφορίες σχετικές με τις πιο συχνές αναφορές που κάνει ο κάθε χρήστης.

Πιο αναλυτικά η κατηγορία αυτή περιέχει τέσσερις κάρτες οι οποίες είναι οι **“Απαντήσεις”** του χρήστη τα **“mentions”**, τα **“hashtags”** και τα **“urls”**. Δείχνει στο χρήστη σε ποιους

άλλους χρήστες απαντάει περισσότερο, ποιά είναι τα πιο συχνά mentions, hashtags, και urls που χρησιμοποιεί. Γενικότερα η κατηγορία **“Στατιστικά”** περιλαμβάνει πληροφορίες οι οποίες απαρτίζονται κυρίως από την συχνοτητα δραστηριότητας του χρήστη. Είναι μία από τις βασικότερες κατηγορίες που θα μπορούσε ο χρήστης να πάρει πληροφορίες για το προφίλ του. Ο λόγος που την καθιστά σημαντική είναι γιατί παρέχει μία πλήρης εικόνα των δεδομένων του χρήστη και το τι μπορεί να κάνει για να επιτύχει όσο το δυνατόν, πιο δημοφιλέστερο προφίλ ο χρήστης

ΣΥΧΝΕΣ ΑΠΑΝΤΗΣΕΙΣ ΤΟΥ ΧΡΗΣΤΗ

Η κάρτα “Απαντήσεις” είναι χρήσιμη για τον χρήστη ώστε να δει με ποιους ανθρώπους είναι περισσότερο συνδεδεμένος. Έτσι αποκτά μία γενική άποψη με το ποιους ανθρώπους έχει περισσότερη επικοινωνία. Η κατηγορία αυτή συνδράμει ώστε να αποκτήσει περισσότερη επικοινωνία και με άλλους χρήστες που δεν είχε στο παρελθόν και να αυξήσει τους followers του και συνάμα την αναλογία followers/following. Παρακάτω βλέπουμε στο σχήμα 31 μία ενδεικτική κάρτα.

Σχήμα 31. Συχνές απαντήσεις του χρήστη

Για να παρουσιάσουμε την παραπάνω κάρτα στον χρήστη, θα πρέπει να πάρουμε με ένα νέο request όλες τις απαντήσεις του χρήστη μέσα από τα tweets του. Κάνοντας το request ο server ζητάει με ένα SQL ερώτημα (SQL Query) τις απαντήσεις (replies) του χρήστη από την βάση, και τα επιστρέφει στην εφαρμογή. Για το συγκεκριμένο ερώτημα

χρησιμοποιούμε INNER JOIN ώστε να πάρουμε τα σωστά δεδομένα για τον συγκεκριμένο χρήστη, εύκολα και γρήγορα. Το εν λόγω SQL ερώτημα είναι το παρακάτω:

```
$sqlgetOwnData = "SELECT t1.replyName
                  FROM `tweet_replies` as t1
                  INNER JOIN
                 `user_timeline` as t2 on t1.tweetId = t2.tweetId
                 WHERE t2.userId = '$userId'";

$result = $conn->query($sqlgetOwnData);
```

Έχοντας τα δεδομένα, η εφαρμογή προχωράει με την επεξεργασία τους ώστε να υπολογίσει τους τρεις πιο συχνούς χρήστες που απαντώνται από τον χρήστη. Ο υπολογισμός αυτός γίνεται από μια γενική μέθοδο στο “calculatorService” και είναι η εξής:

```
topThree: function (obj) {
    obj.sort(function (a, b) {
        if (a.value < b.value) {
            return 1;
        }
        else if (a.value == b.value) {
            return 0;
        }
        else {
            return -1;
        }
    });

    obj = obj.slice(0, 3);
    var temp = obj[0];
    obj[0] = obj[1];
    obj[1] = temp;

    for (var i = 0; i < obj.length; i++) {
        if (!obj[i] || typeof obj[i] == 'undefined' || obj[i].name == 'undefined' || obj[i].value ==
'undefined') {
            obj.splice(i, 1);
        }
    }

    return obj;
}
```

Τέλος, προχωράει με την δημιουργία του γραφήματος καλώντας το “chartService”, όμοια με τα προηγούμενα βήματα.

Ο χρήστης μπορεί εύκολα να δει όλες του τις απαντήσεις (replies) σε άλλους χρήστες πατώντας στο πάνω δεξιά κουμπί της αντίστοιχης κάρτας (βλέπε σχήμα 31).

ΣΥΧΝΑ MENTIONS ΤΟΥ ΧΡΗΣΤΗ

Στην συνέχεια συναντάμε την κάρτα “Mentions”, τα mentions είναι ένας πολύ καλός τρόπος να αναδείξουμε κάποιον στο Twitter. Στην κάρτα αυτή λοιπόν ο χρήστης βλέπει τα πιο συχνα mentions που έχει κάνει.
Με αυτό τον τρόπο επιτυγχάνει να δούνε το tweet πολλά άτομα και να αποκτήσει εξίσου και με αυτό τον τρόπο περισσότερο ενεργό ρόλο στο Twitter και ταυτόχρονα να αυξηθούν και πάλι οι followers του. Επίσης τα mentions είναι χρήσιμα, όταν ο χρήστης γνωρίζει ότι ένα συγκεκριμένο πρόσωπο θα ενδιαφερθεί και θα απολαύσει ένα άρθρο, ή μια πληροφορία που ο ίδιος θέλει να αναρτήσει, αλλά ότι και το υπόλοιπο των followers κάθε χρήστη θα επωφεληθεί επίσης από το άρθρο ή την πληροφορία. Στο σχήμα 32 βλέπουμε ένα παράδειγμα των mentions που χρησιμοποιεί ένας χρήστης.

Σχήμα 32. Συχνά mentions του χρήστη

Για τα mentions ακολουθούμε αντίστοιχη λογική με τις απαντήσεις (replies). Πρώτα, η εφαρμογή κάνει ένα request στον server για να πάρει τα δεδομένα και αυτός με την σειρά

του κάνει το κατάλληλο SQL ερώτημα (SQL Query) ώστε να τα πάρει από την βάση. Το ερώτημα είναι το παρακάτω:

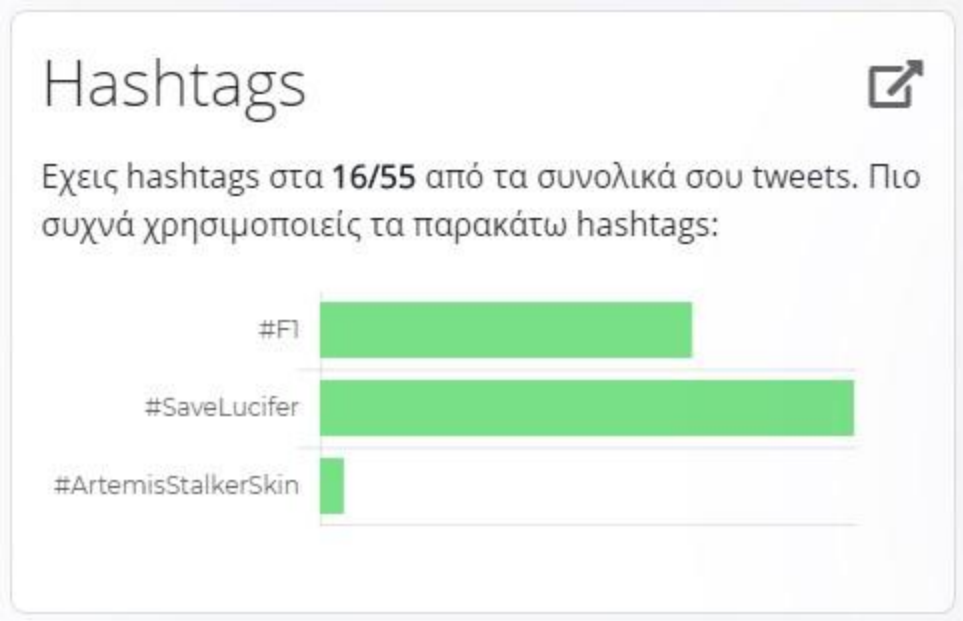
```
$sqlgetOwnData = "SELECT t1.tweetMention
                  FROM `tweet_mentions` as t1
                  INNER JOIN
                  `user_timeline` as t2 on t1.tweetId = t2.tweetId
                  WHERE t2.userId = '$userId'";

$result = $conn->query($sqlgetOwnData);
```

Όπως και προηγουμένως, καλείτε η κατάλληλη μέθοδος του “calculatorService” ώστε να βρεθούν οι τρεις συχνότερες αναφορές (mentions). Αμέσως μετά μπορεί να προχωρήσει με την παρουσίαση και δημιουργία του γραφήματος, καλώντας την κατάλληλη μέθοδο του “chartService”.

Ο χρήστης και πάλι έχει την δυνατότητα να δει όλες του τις αναφορές (mentions), πατώντας πάνω δεξιά στην κάρτα των αναφορών.

ΣΥΧΝΑ HASHTAGS ΤΟΥ ΧΡΗΣΤΗ

Η επόμενη κάρτα αναφέρεται στα “Hashtags” όπου εδώ ο χρήστης μπορεί να δει ποια είναι τα πιο συχνά hashtags στα οποία αναφέρεται.
Τα hashtags είναι αρκετά δημοφιλή και οι χρήστες τα χρησιμοποιούν πάρα πολύ ώστε να αυξηθούν οι followers τους αλλά και για να αναδείξουν μια κατάσταση ή ένα θέμα. Τα hashtags βοηθούν τον χρήστη ώστε να γίνει περισσότερο δημοφιλής και να προσελκύσει περισσότερους χρήστες να ανατρέξουν στο προφίλ του. Επίσης τα hashtags μπορούν να “τραβήξουν “ χρήστες που έχουν ένα συγκεκριμένο ενδιαφέρον ή δραστηριότητα, με αυτό τον τρόπο οι χρήστες συνδέονται μεταξύ τους μέσω ενός κοινού ενδιαφέροντος. Στην συνέχεια παρουσιάζεται στο σχήμα 33 η κάρτα ενός χρήστη με τα πιο συχνά του hashtags.



Σχήμα 33. Συχνά hashtags του χρήστη

Για να φέρουμε τα δεδομένα των hashtags, πρέπει να κάνουμε ένα request στον server μας ώστε αυτός να κάνει το κατάλληλο SQL ερώτημα (SQL Query). Το ερώτημα είναι όμοιο με τα προηγούμενα, χρησιμοποιεί δηλαδή INNER JOIN ώστε να πάρουμε τα σωστά δεδομένα για τον κάθε χρήστη. Παρακάτω είναι το ερώτημα:

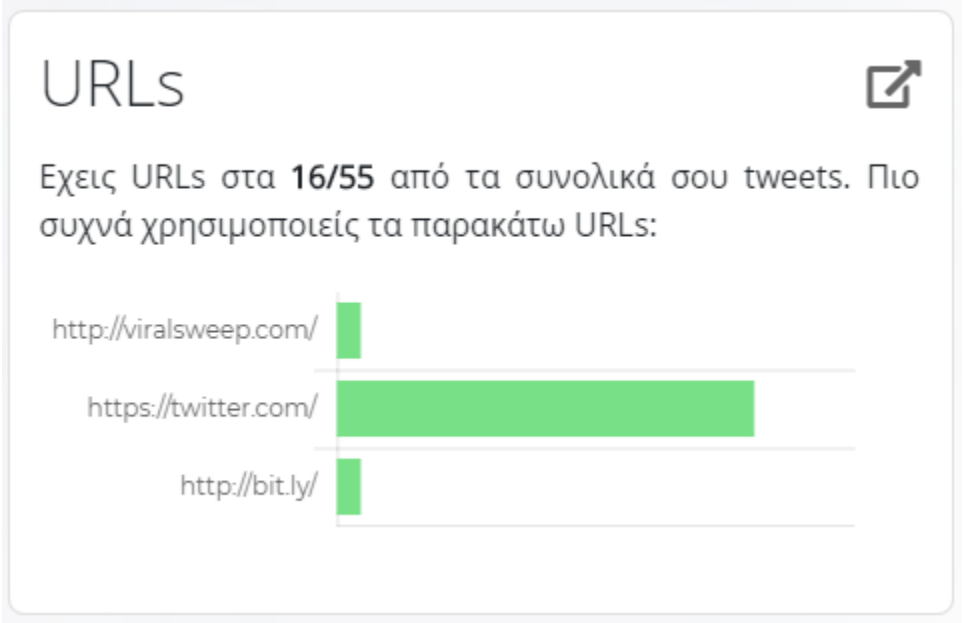
```
$sqlgetOwnData = "SELECT t1.tweetHashtag
                  FROM `tweet_hashtags` as t1
                  INNER JOIN
                  `user_timeline` as t2 on t1.tweetId = t2.tweetId
                  WHERE t2.userId = '$userId'";

$result = $conn->query($sqlgetOwnData);
```

Τα hashtags, έχουν και αυτά αντίστοιχη λογική με τις προηγούμενες κάρτες της κατηγορίας. Δηλαδή, βρίσκουμε τα τρία πιο συχνά hashtags με την βοήθεια του “calculatorService”. Τέλος προχωράμε με την δημιουργία του γραφήματος, δίνοντας τα κατάλληλα δεδομένα στην σωστή μέθοδο του “chartService”.

ΣΥΧΝΑ URLS ΤΟΥ ΧΡΗΣΤΗ

Τελευταία είναι η κάρτα με τα “URLs”, αφορά τα πιο συχνά URLs που έχει αναρτήσει ο χρήστης σε κάποιο tweet του. Τα urls φανερώνουν πιο συγκεκριμένα, τα ενδιαφέροντα που έχει ο χρήστης. Του δείχνουν ποιες είναι οι συνολικές του και πιο

συχνές του ασχολίες σε γενικό βαθμό, websites δηλαδή που ο ίδιος παρακολουθεί και αναρτά στο Twitter. Παρακάτω στο σχήμα 34 βλέπουμε την κάρτα των URLs.



Σχήμα 34. Συχνά urls του χρήστη

Για τα URLs, κάνουμε το αντίστοιχο request στον server ώστε να μας φέρει τα σωστά δεδομένα. Ο server με την σειρά του, κάνει το SQL ερώτημα (SQL Query) και φέρνει τα κατάλληλα δεδομένα από την βάση. Το ερώτημα είναι το εξης:

```
$sqlgetOwnData = "SELECT t1.tweetUrl
                  FROM `tweet_urls` as t1
                  INNER JOIN
                 `user_timeline` as t2 on t1.tweetId = t2.tweetId
                 WHERE t2.userId = '$userId'";

$result = $conn->query($sqlgetOwnData);
```

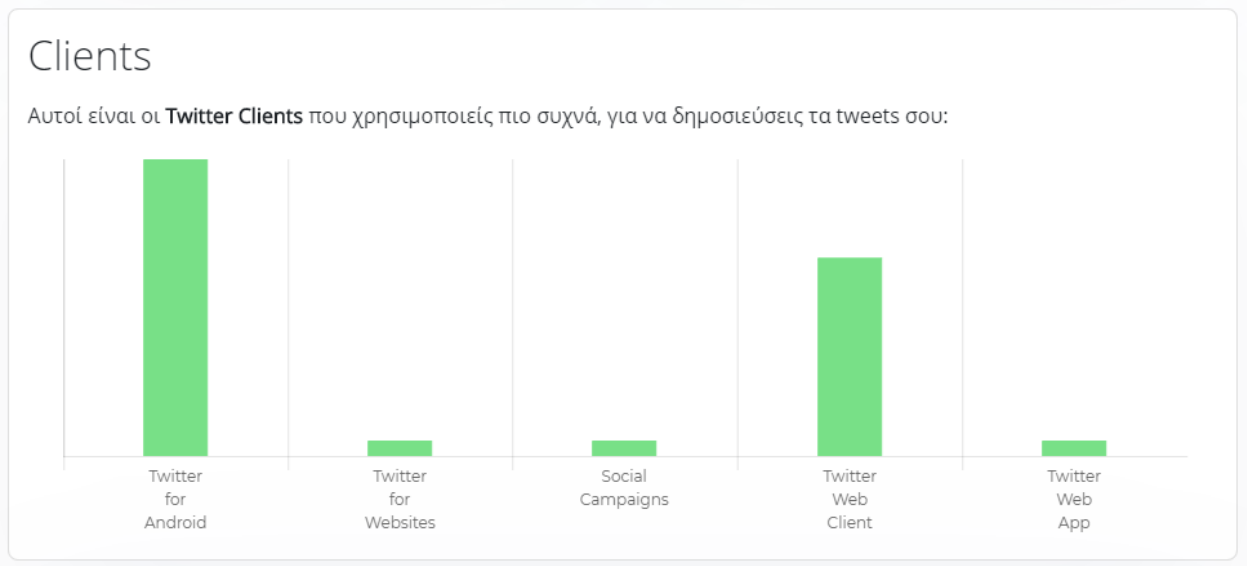
Μόλις έρθουν τα δεδομένα βρίσκουμε τα τρία πιο συχνά URLs και στη συνέχεια δημιουργούμε το κατάλληλο γράφημα, με τη χρήση του “chartService”.

ACTIVITY (ΔΡΑΣΤΗΡΙΟΤΗΤΑ)

Η κατηγορία “**Activity**” έχει να κάνει με την δραστηριότητα του χρήστη στο Twitter. Απαρτίζεται από δύο κάρτες τους “**clients**” και τον “**ενεργό χρόνο**”. Στόχος αυτής της κατηγορίας, είναι να μπορέσει ο χρήστης να δει πως και πόσο χρησιμοποιεί το προφίλ του. Πιο συγκεκριμένα, μπορεί εύκολα μέσα από το αντίστοιχο γράφημα, να δει ποιες είναι οι πιο συχνές ώρες που κάνει tweets στο εικοσιτετράωρο ή μέσα από ποιες εφαρμογές (και κατά συνέπεια συσκευές) κάνει τα περισσότερα από τα tweets του. Παρακάτω διατυπώνονται πιο αναλυτικά οι δύο αυτές κάρτες.

CLIENTS

Οι κάρτα των clients αφορά κυρίως την εφαρμογή που χρησιμοποιεί ο χρήστης για να κάνει την ανάρτησή του, τον σχολιασμό του η οτιδήποτε άλλο επιθυμεί. Δηλαδή μας δείχνει αν η οποιαδήποτε δημοσίευση του έγινε από android εφαρμογή ή από iOS εφαρμογή ή από web εφαρμογή κτλ. Παράλληλα ο χρήστης μπορεί να δει ποια εφαρμογή χρησιμοποιεί περισσότερο. Για παράδειγμα αν χρησιμοποιεί περισσότερο android εφαρμογή ή iOS εφαρμογή μπορεί να γνωρίζει ότι τα περισσότερα tweet που κάνει είναι σε εξωτερικό χώρο. Στο σχήμα 35 βλέπουμε την κάρτα με τους clients.



Σχήμα 35. Γράφημα των Twitter Clients

Για να πάρουμε τα δεδομένα των Twitter Client πρέπει και εδώ να κάνουμε ένα απλό request στον server ώστε να πάρουμε τα κατάλληλα δεδομένα. Ο server κάνει και εδώ

ένα SQL ερώτημα (SQL Query), όμοια με τα προηγούμενα, δηλαδή χρησιμοποιεί INNER JOIN. Το εν λόγω ερώτημα είναι το παρακάτω:

```
$sqlgetOwnData = "SELECT t1.tweetClient
                  FROM `tweet_clients` as t1
                  INNER JOIN
                 `user_timeline` as t2 on t1.tweetId = t2.tweetId
                 WHERE t2.userId = '$userId'";

$result = $conn->query($sqlgetOwnData);
```

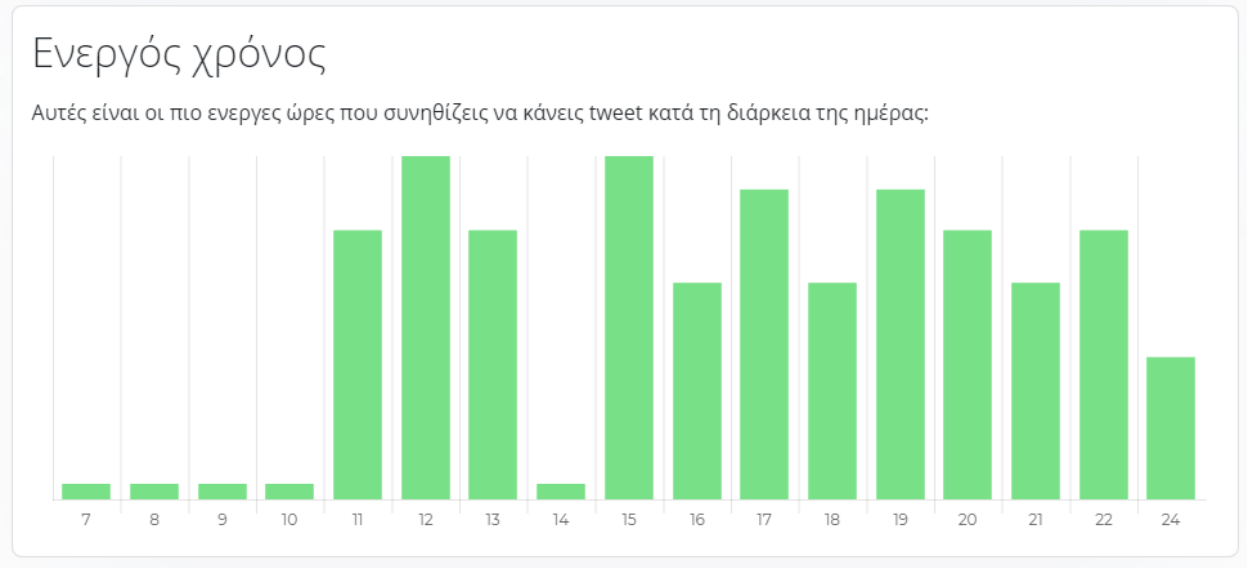
Όταν η εφαρμογή αποκτήσει τα δεδομένα, τα κάνει mapping, βρίσκει δηλαδή πόσες φορές εμφανίζεται κάθε client και στη συνέχεια τα βάζει σε μια μεταβλητή τύπου JSON object ώστε να δημιουργήσει το γράφημα μέσω του ‘chartService”. Η μέθοδος που κάνει το mapping είναι η mapDataObject() και χρησιμοποιείται σε πολλές περιπτώσεις (για τους clients, URLs, hashtags κ.α.). Η συνάρτηση περιγράφεται στην παρακάτω εικόνα:

```
var mapDataObject = function (data, categoryName) {
    var obj = [];
    var mappedObj = [];

    _.each(data, function (o) {
        obj.push(o.name);
    });

    _.each(obj, function (i) {
        mappedObj[i] = (mappedObj[i] || 0) + 1;
    });

    var objFinal = Object.entries(mappedObj).map(([name, value]) => ({
        name, value
    }));

    switch (categoryName) {
        case 'REPLIES':
            $scope.finalReplies = objFinal;
            break;
        case 'MENTIONS':
            $scope.finalMentions = objFinal;
            break;
        case 'HASHTAGS':
            $scope.finalHashtags = objFinal;
            break;
        case 'URLS':
            $scope.finalUrls = objFinal;
            break;
        default:
            break;
    }

    return objFinal;
}
```

ΕΝΕΡΓΟΣ ΧΡΟΝΟΣ

Έπειτα, έχουμε την κάρτα “ενεργός χρόνος” η οποία δείχνει στο χρήστη ποιες ώρες της ημέρας είναι περισσότερο ενεργός κάνοντας tweet. Από αυτή την κάρτα ο χρήστης μπορεί να παρατηρήσει ποιες ώρες της ημέρας κάνει τα περισσότερα tweet. Μπορεί να ρυθμίσει το χρόνο του και να κάνει τα tweet σε ώρες αιχμής ώστε να βοηθηθεί και να

αποκτήσει περισσότερους followers. Αυτό μπορεί ταυτόχρονα να τον κάνει πιο δημοφιλή. Παρακάτω βλέπουμε το σχήμα 36 που μας δείχνει ένα παράδειγμα ενός χρήστη.



Σχήμα 36. Γράφημα του ενεργού χρόνου

Για την παρουσίαση του ενεργού χρόνου (active time), αρκεί να κάνουμε ένα request στον server, ώστε να παραλάβουμε τα σωστά δεδομένα. Ο server όπως και κάθε προηγούμενη φορά θα κάνει ένα SQL ερώτημα (SQL Query) με την χρήση του INNER JOIN ώστε τα δεδομένα που θα πάρει να είναι σωστά. Το SQL ερώτημα είναι το παρακάτω:

```
$sqlgetOwnData = "SELECT t1.tweetTime
                  FROM `tweet_time` as t1
                  INNER JOIN
                 `user_timeline` as t2 on t1.tweetId = t2.tweetId
                 WHERE t2.userId = '$userId'";

$result = $conn->query($sqlgetOwnData);
```

Μόλις πάρει τα δεδομένα η εφαρμογή, ακολουθεί την ίδια διαδικασία με την κάρτα των Twitter Client, δηλαδή βρίσκει πόσες φορές υπάρχει στην απάντηση (response) που μας έδωσε ο server η κάθε διαφορετική ώρα που έγινε κάποιο tweet. Αυτό γίνεται με την μέθοδο που αναφέρθηκε και προηγουμένως, δηλαδή της mapDataObject(). Στη συνέχεια

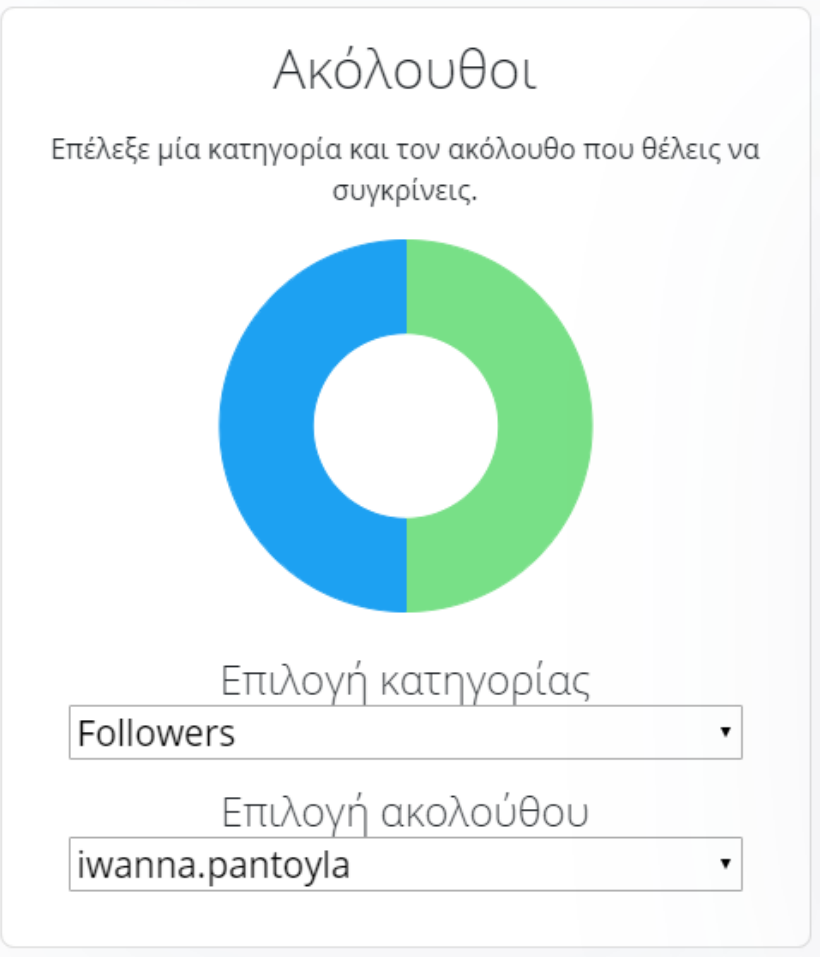
καλεί την κατάλληλη μέθοδο του “chartService” ώστε να δημιουργήσει το σωστό γράφημα για την περίσταση.

COMPARE (ΣΥΓΚΡΙΣΗ)

Η τελευταία μας κατηγορία είναι η “Compare” η οποία απαρτίζεται από δύο κάρτες οι οποίες είναι οι **“Ακόλουθοι”** και “**Ακολουθεί**”. Σε γενικές γραμμές, ο χρήστης μπορεί στην συγκεκριμένη κατηγορία να συγκρίνει τον εαυτό του με τους ακολούθους του αλλά και με εκείνους που ακολουθεί μόνο εκείνος. Μπορεί να συγκρίνει τον εαυτό του με έναν άλλο ακόλουθο να δει τις διαφορές μεταξύ τους ώστε να καταλήξει σε ένα δικό του συμπέρασμα για τον στόχο που μπορεί ο ίδιος να θέλει να επιδιώξει να φτάσει.

ΑΚΟΛΟΥΘΟΙ

Οι κάρτα Ακόλουθοι, αντιπροσωπεύει τους ανθρώπους οι οποίοι σε ακολουθούν. Ως χρήστης κάποιος μπορεί να συγκρίνει τους followers του, τους following του, τα tweets του και τα likes του με έναν οποιονδήποτε από τους ακόλουθους του. Μπορεί να συγκρίνει το πόσο δημοφιλής είναι σε σχέση με έναν ακόλουθο του, αυτό του δίνει κίνητρο να συνεχίσει να είναι ενεργός στο Twitter. Παρακάτω βλέπουμε τέσσερα σχήματα ανά κατηγορία.



Σχήμα 37. Ακόλουθοι - followers

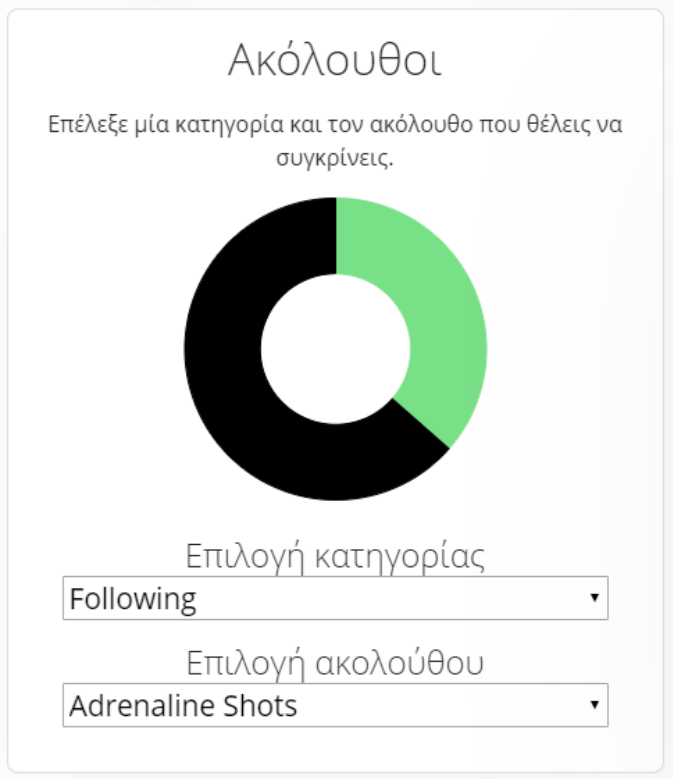
Για την κάρτα της σύγκρισης με τους followers (ακόλουθοι) χρειάστηκε ένα request στον server ώστε να πάρουμε τα δεδομένα των ακολούθων μας. Ο server ζητάει με ένα SQL ερώτημα (SQL Query) τα κατάλληλα δεδομένα από την βάση δεδομένων, με την χρήση του INNER JOIN και στη συνέχεια τα επιστρέφει στην εφαρμογή. Φυσικά, χρειαζόμαστε και τα δεδομένα του συνδεδεμένου χρήστη, όμως αυτά τα έχουμε ήδη από προηγούμενο request. Το ερώτημα είναι το εξής:

```
$sqlgetFollowersData = "SELECT *
                        FROM `user_followers`
                        WHERE userId = ".$userId;
$result = $conn->query($sqlgetFollowersData);
```

Η εφαρμογή, έχοντας τα δεδομένα των ακολούθων, τα ετοιμάζει μαζί με τα δεδομένα του συνδεδεμένου χρήστη μέσω της μεθόδου `$scope.prepareVerticalFollowerChart()`. Η συγκεκριμένη μέθοδος είναι η παρακάτω:

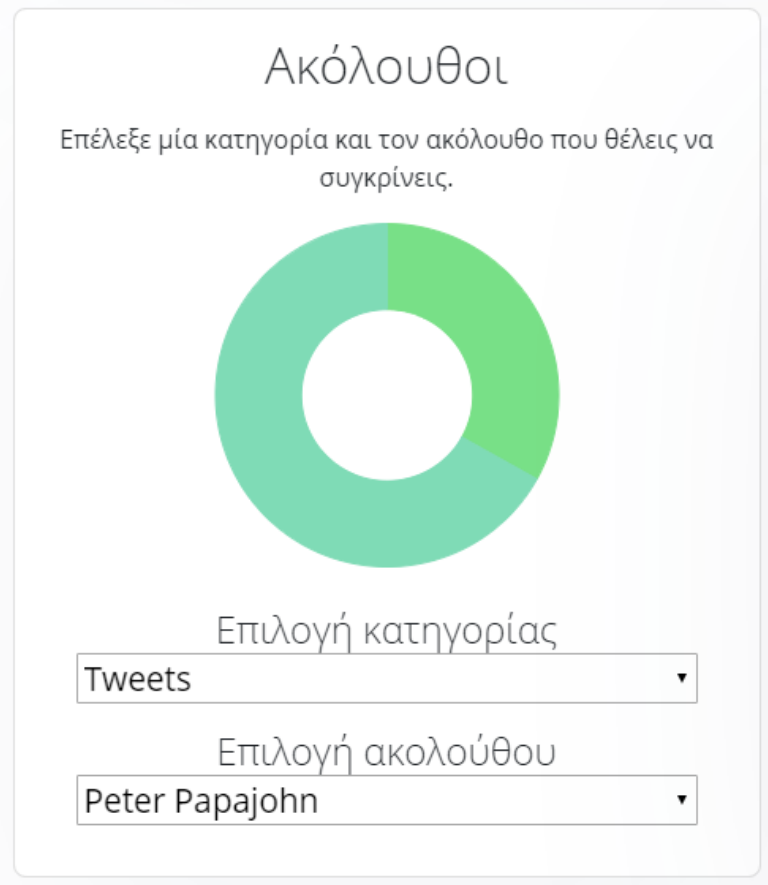
```
$scope.prepareVerticalFollowerChart = function (ownData, followerData) {

    chartService.createDoughnutChart('followers-chart',
        [{
            'label': ownData.userScreenName,
            'value': eval($scope.selectedCategory.ownAction),
            'color': ownData.userProfileColor
        }, {
            'label': followerData.followerScreenName,
            'value': eval($scope.selectedCategory.followerAction),
            'color': followerData.followerProfileColor
        }]);
}
```

Στη συνέχεια τα στέλνει στην κατάλληλη μέθοδο του “chartService” ώστε να εμφανιστούν μέσα από το διάγραμμα στον χρήστη.



Σχήμα 38. Ακόλουθοι - following

Για τους following δεν χρειάζεται να κάνουμε κάποια request από τον server, καθώς έχουμε ήδη τα δεδομένα από προηγουμένως. Το μόνο που θα αλλάξει είναι ότι αυτή τη φορά δεν θα απεικονίσουμε τους followers αλλά τους following.



Σχήμα 39. Ακόλουθοι - Tweets

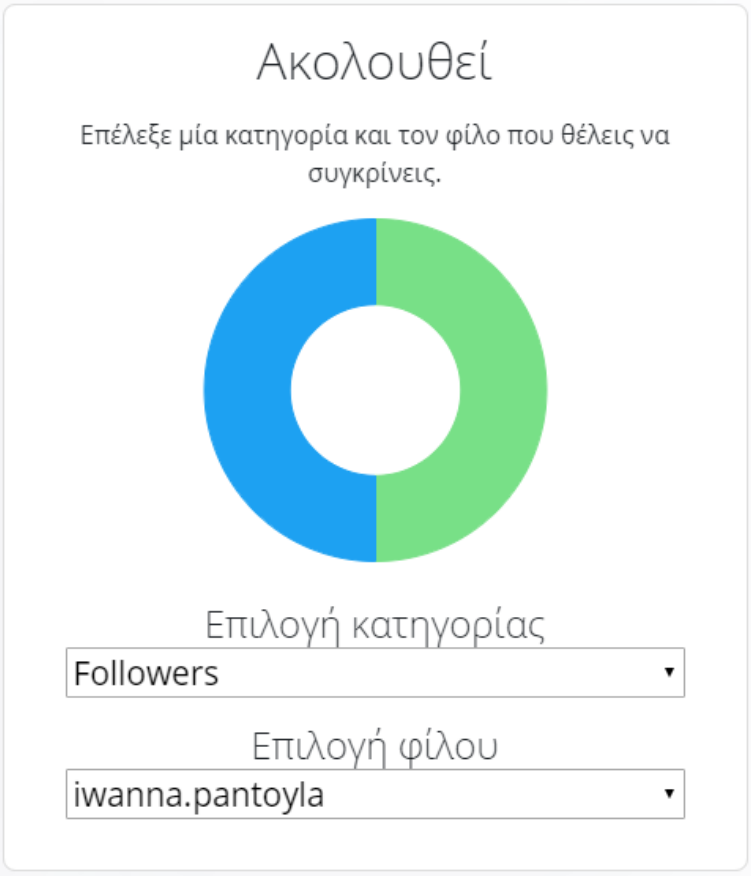
Τα δεδομένα για την σύγκριση των tweets είναι και αυτά έτοιμα από προηγουμένως, όπως και παραπάνω, η διαδικασία είναι ίδια, μόνο που τώρα απεικονίζουμε στον κυκλικό διάγραμμα τα tweets των δύο χρηστών.

Σχήμα 40. Ακόλουθοι - Likes

Τελευταία στην κάρτα με τους followers είναι τα likes. Τα likes είναι ήδη έτοιμα από το προηγούμενο request οπότε προχωράμε με την παρουσίαση τους στο κυκλικό διάγραμμα, όπως ακριβώς και προηγουμένως.

ΑΚΟΛΟΥΘΕΙ

Σε αυτή την κάρτα ο χρήστης μπορεί να δει τους ανθρώπους που ο ίδιος ακολουθεί. Όπως και στη παραπάνω κάρτα ο χρήστης μπορεί να συγκρίνει τους followers του, τους following του, τα tweets του και τα likes του με έναν οποιονδήποτε χρήστη που ακολουθεί. Μπορεί να συγκρίνει το πόσο δημοφιλής είναι σε σχέση με ένα άτομο που ακολουθεί, ταυτόχρονα, αν κάποια από τα άτομα που ακολουθεί είναι περισσότερο δημοφιλή μπορεί να δει σε ποια από τις παραπάνω κατηγορίες υστερεί και να αφιερώσει περισσότερο χρόνο σε εκείνη για να πετύχει το επιθυμητό αποτέλεσμα που θέλει να έχει. Στην συνέχεια βλέπουμε τα παρακάτω σχήματα ανά κατηγορία.



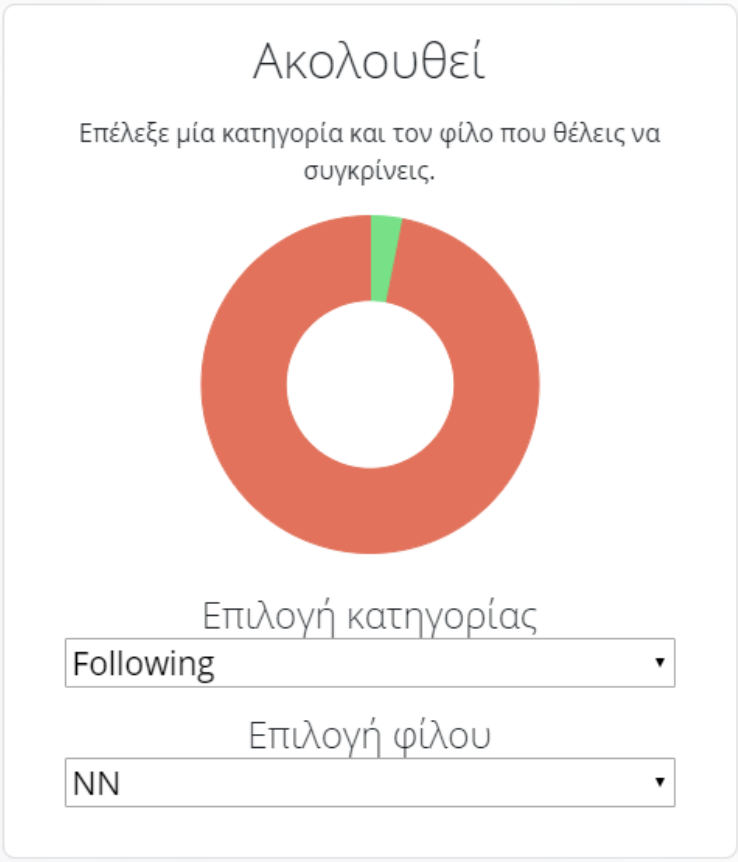
Σχήμα 41. Ακολουθεί - followers

Όμοια και με τις κάρτες σύγκρισης προηγουμένως, για να δημιουργήσουμε αυτή την κάρτα πρέπει να κάνουμε το κατάλληλο request στον server ώστε να μας φέρει τα δεδομένα. Αυτή τη φορά θα χρειαστούμε τα δεδομένα των following (αυτών που ο συνδεδεμένος χρήστης ακολουθεί). Φυσικά, ο server θα κάνει με τη σειρά του ένα SQL ερώτημα (SQL Query) στην βάση και αφού πάρει τα δεδομένα, θα μας τα επιστρέψει με μορφή JSON. Το αναφερόμενο SQL ερώτημα είναι το παρακάτω:

```
$sqlgetFriendsData = "SELECT *
                      FROM `user_friends`
                      WHERE userId = ".$userId;

$result = $conn->query($sqlgetFriendsData);
```

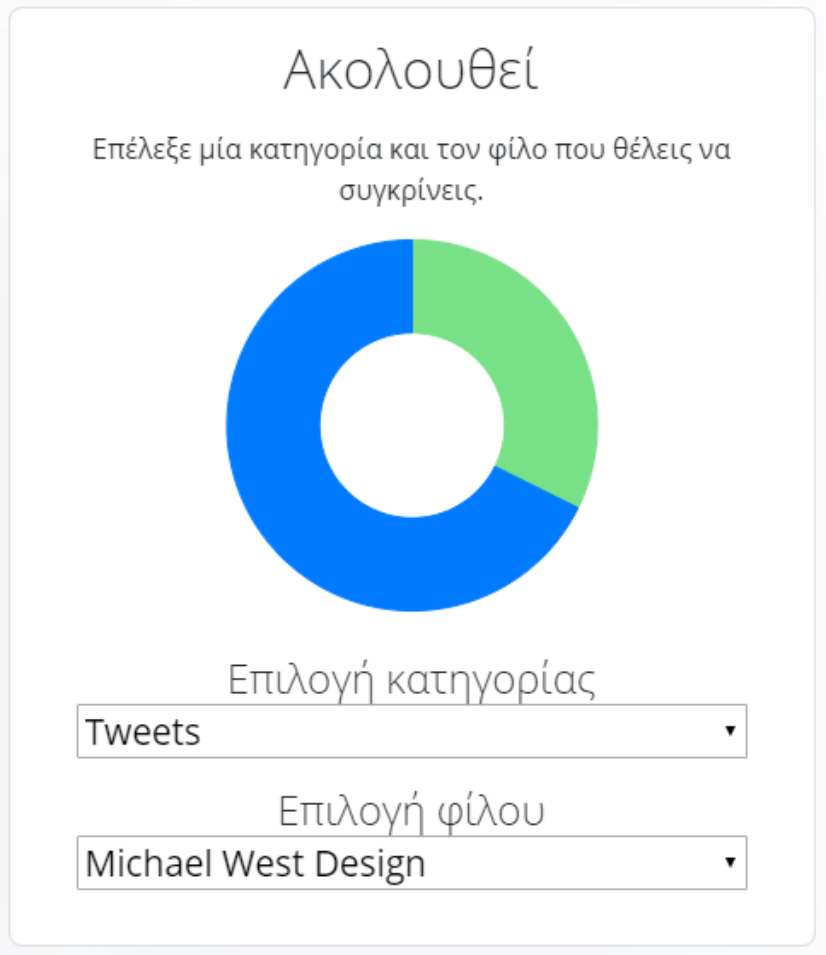
Εφόσον η εφαρμογή αποκτήσει τα δεδομένα, τα στέλνει μαζί με τα δεδομένα του συνδεδεμένου χρήστη στην μέθοδο `$scope.prepareVerticalFollowerChart()` όπου τα ετοιμάζει για να δημιουργήσει το κατάλληλο γράφημα, με την χρήση της κατάλληλης μεθόδου του “chartService”. Η αναφερόμενη μέθοδος είναι όμοια με αυτή των follower και είναι η παρακάτω:

```
$scope.prepareVerticalFriendChart = function (ownData, friendData) {
    chartService.createDoughnutChart('friends-chart',
        [{
            'label': ownData.userScreenName,
            'value': eval($scope.selectedCategory.ownAction),
            'color': ownData.userProfileColor
        }, {
            'label': friendData.friendScreenName,
            'value': eval($scope.selectedCategory.friendAction),
            'color': friendData.friendProfileColor
        }]);
}
```



Σχήμα 42. Ακολουθεί - following

Για τους following δεν χρειάζεται να ξανά κάνουμε κάποια request καθώς τα δεδομένα είναι έτοιμα από το request της προηγούμενης κάρτας. Οπότε προχωράμε με την δημιουργία του κατάλληλου γραφήματος μέσα από την αντίστοιχη μέθοδο του “chartService”.



Σχήμα 43. Ακολουθεί - Tweets

Όπως και με τους following, τα δεδομένα για την δημιουργία αυτού του γραφήματος είναι ήδη διαθέσιμα, η διαφορά είναι ότι τώρα χρησιμοποιούμε τα tweets του χρήστη για την σύγκριση. Για την δημιουργία του γραφήματος χρησιμοποιούμε την κατάλληλη μέθοδο του “chartService”.

Σχήμα 44. Ακολουθεί - Likes

Τέλος, τα Likes έχουν και αυτά την ίδια λογική. Τα δεδομένα για το γράφημα είναι έτοιμα, οπότε χρησιμοποιώντας τα δεδομένα των Like και με την χρήση του “chartService” δημιουργούμε το κυκλικό γράφημα.

ΑΠΟΣΥΝΔΕΣΗ

Για το κλείσιμο του website, ο χρήστης το μόνο που έχει να κάνει είναι να κλείσει την καρτέλα ή τον browser του. Παρόλα αυτά, ο χρήστης έχει την δυνατότητα να κάνει αποσύνδεση του λογαριασμού του από το site χειροκίνητα. Ο χρήστης μπορεί να το επιλέξει αυτό είτε για να είναι σίγουρος ότι αποσυνδέθηκε, είτε γιατί θέλει να συνδεθεί ξανά με κάποιον άλλο Twitter λογαριασμό.

Πατώντας το κουμπί της αποσύνδεσης (πάνω δεξιά), η εφαρμογή κάνει στον server ένα ειδικό request για να τον ειδοποιήσει ότι ο συνδεδεμένος χρήστης θέλει να αποσυνδεθεί. Η παραπάνω διαδικασία ξεκινάει με τον παρακάτω κώδικα:

```
$scope.disconnectUser = function () {
    requestService.disconnectUser().then(function (resp) {
        cacheService.clearAllSession();
        $rootScope.isConnected = false;
        location.hash = '';
    });
}
```

Πρώτο βήμα στον παραπάνω κώδικα είναι να κάνει το αναφερόμενο request στον server. Μόλις ο server λάβει το request θα αδειάσει τις αποθηκευμένες πληροφορίες από την σύνδεση του χρήστη (token, secretToken) και θα στείλει πίσω στην εφαρμογή το response. Στην PHP, η εντολή για τον καθαρισμό του Session είναι ο `session_destroy();`.

Μόλις η εφαρμογή λάβει το response, προχωράει με το επόμενο βήμα, το οποίο είναι να καθαρίσει το Session Storage του browser. Το Session Storage είναι ένα προσωρινό μέρος στον browser που μπορεί ένα site να κρατάει δεδομένα για γρηγορότερη πρόσβαση. Στην περίπτωση μας, κρατάμε τα δεδομένα του χρήστη τα οποία ζητάμε από τον server καθόλη την διάρκεια της περιήγησης. Με αυτό τον τρόπο εξασφαλίζουμε γρηγορότερη φόρτωση των δεδομένων και αποφεύγουμε τον φόρτο, τόσο στον server μας, όσο και στο Twitter το ίδιο.

Ο κώδικας για τον καθαρισμό του Session Storage είναι ο εξής: `sessionStorage.clear();`

Τέλος, καθαρίζουμε μια τοπική μεταβλητή που υποδεικνύει την κατάσταση σύνδεση του χρήστη και τον επιστρέφουμε στην αρχική σελίδα (landing page).

ΣΤΟΧΟΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Η εφαρμογή μας είναι ευανάγνωστη ταυτόχρονα εύχρηστη και εξυπηρετική για τον κάθε χρήστη. Στόχος της χρήσης της εφαρμογής είναι ο χρήστης να εξοικειωθεί περισσότερο με το λογαριασμό του στο Twitter, παράλληλα να είναι περισσότερο ενεργός, να συγκρίνει τον εαυτό του με άλλους χρήστες και να παρακολουθεί τα διάφορα στατιστικά δεδομένα του. Αυτό έχει ως αποτέλεσμα ο χρήστης να έχει μια γενική άποψη για το προφίλ του, με βασικό του στόχο να γίνει όσο περισσότερο δημοφιλής γίνεται.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Twitter database schema
https://gist.github.com/abraham/352528

Abrahams Twitter oAuth PHP Library
https://twitteroauth.com/

Database Entity Relationship Model
https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model

Twitter API Reference Index
https://developer.twitter.com/en/docs/api-reference-index

AngularJS
https://angularjs.org/

Bootstrap 4
https://getbootstrap.com/

Animate.CSS
https://daneden.github.io/animate.css/

Chart.JS
https://www.chartjs.org/

HTML
https://www.w3schools.com/html/default.asp

CSS
https://www.w3schools.com/css/css_intro.asp

JavaScript / JQuery
https://www.w3schools.com/js/default.asp

PHP
https://www.w3schools.com/php/php_intro.asp