ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Web 2.0 Social Platform and Viral Marketing Modular Development

Λουδάρος Νικόλαος

A.M.: 250

Επιβλέπων Καθηγητής
Παπαδουράκης Γεώργιος

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ – ABSTRACT
_____ 3

1. Τα Social Communities και ο σημερινός τους ρόλος
    1.1 Το Web 2.0 και οι τεχνολογίες του 4
    1.2 Χαρακτηριστικά του Web 2.0 5
    1.3 Κριτικές για το Web 2.0 5
    1.4 Ιδιωτικότητα στο Web 2.0 6
    1.5 Η ελληνική διάσταση του Web 2.0 και των Digital Media 7
    1.6 Virtual community 8
    1.7 Ο κύκλος ζωής 10
    1.8 Τι μας περιμένει στο μέλλον 11
2. Η κοινωνική πλατφόρμα Dolphin Community Builder 7.0
    2.1 Είδη κοινωνικής πλατφόρμας 14
    2.2 Η πλατφόρμα Dolphin 15
    2.3 Τεχνολογίες 15
    2.4 Πληροφορίες Ανάπτυξης Προγραμματιδίων 16

3. Ανάπτυξη Προγραμματιδίου Ιστολογίων
    3.1 Δομή Προγραμματιδίου (εγκαταστάσιμης/ λειτουργικής μορφής) 19
    3.2 Αλλαγή κειμένου 22
    3.3 Δυναμικό περιεχόμενο 23
    3.4 Προσθήκη σελίδας κέντρου Διαχείρισης 26
        3.4.1 Σελίδα διαχείρισης Προγραμματιδίου 26
        3.4.2 Μεταβλητές διαχείρισης 27
        3.4.3 Φόρμα επιλογών στο Κέντρο Διαχείρισης 29
        3.4.4 Σύνδεσμος προσελίδα Διαχείρισης του Προγραμματιδίου 31
    3.5 Λειτουργία Ιστολογίου 33
        3.5.1 Σελίδα Προσθήκης δημοσιεύματος 34
        3.5.2 Προβολή Δημοσιεύματος 37
    3.6 Αρχική Σελίδα 39
    3.7 Επεξεργασία δημοσιεύσεων Ιστολογίου 40
    3.8 Τελικά Αρχεία 43

Ευχαριστίες – Acknowledgments 58
Πηγές 59
Βιβλιογραφία & Χρήσιμες Ιστοσελίδες 60

ΠΕΡΙΛΗΨΗ

Σε αυτήν την πτυχιακή εργασία καλούμαστε να βελτιώσουμε τις εγγενείς ρουτίνες μιας κοινωνικής πλατφόρμας, με στόχο την βελτίωση του γραφικού περιβάλλοντος χρήστη και απώτερο στόχο την αξιοποίηση της έννοιας του viral marketing. Έτσι, μετά από την ανάπτυξη τμημάτων κώδικα που δρουν και αυτόνομα ως μικρο-εφαρμογές, η πλατφόρμα μας στο τελικό της στάδιο θα έχει σχεδόν την μορφή που έχει σήμερα μία γνωστή σε όλους μας Facebook application.

Στο πρώτο κεφάλαιο, λοιπόν, κάνουμε μια σύντομη ιστορική αναδρομή στον ρόλο που έπαιξαν οι Κοινωνικές Πλατφόρμες καθώς και η τεχνολογία που αξιοποιούν στην ιστορία του Διαδικτύου. Στο δεύτερο κεφάλαιο, συγκρίνουμε ποικίλες αντίστοιχες τεχνολογίες, και επιλέγουμε την αρμόζουσα, τελικώς αξιολογόντας το υπάρχον Περιβάλλον Ανάπτυξης (API) που μας προσφέρει. Στο τρίτο, και τελευταίο, κεφάλαιο αναπτύσσουμε μια μικρο-εφαρμογή με χρήση των προαναφερθέντων τεχνολογιών.

Κεφάλαιο 1ο
Τα Social Communities και ο σημερινός τους ρόλος
1.1 Το Web 2.0 και οι τεχνολογίες του

Ο όρος Web 2.0 (Παγκόσμιος Ιστός [έκδοση] 2,0), χρησιμοποιείται για να περιγράψει τη νέα γενιά του Παγκόσμιου Ιστού η οποία βασίζεται στην όλο και μεγαλύτερη δυνατότητα των χρηστών του Διαδικτύου να μοιράζονται πληροφορίες και να συνεργάζονται online. Μιλάμε για μία δυναμική διαδικτυακή πλατφόρμα στην οποία μπορούν να αλληλεπιδρούν χρήστες χωρίς εξειδικευμένες γνώσεις σε θέματα υπολογιστών.

Ιστορία του Web 2.0

Η φράση Web 2.0 ειπώθηκε για πρώτη φορά το 2004 κατά τη διάρκεια ενός συνεδρίου μεταξύ της O'Reilly Media και της MediaLive International όπου προτείνονταν ιδέες για την αναβάθμιση του παγκόσμιου ιστού. Ο Dale Dougherty, web pioneer και O'Reilly VP, παρατήρησε ότι το διαδίκτυο είχε αρχίσει να γίνεται πολύ δημοφιλές και σημαντικό μέρος της καθημερινότητας όλο και μεγαλύτερου ποσοστού ανθρώπων. Συνεχώς έβγαιναν νέες εφαρμογές και ιστοσελίδες οι οποίες αναγνωρίζονταν από το ευρύ κοινό σε σύντομο χρονικό διάστημα. Επιπλέον οι περισσότερες εταιρείες άρχισαν να στρέφουν την επιχειρηματική τους δράση στο διαδίκτυο και να προσπαθούν να φέρουν τους καταναλωτές τους προς αυτό το κανάλι.

Έχοντας όλα αυτά ως δεδομένα εξέτασαν το Web 2.0 ως μια δεύτερη γενιά υπηρεσιών βασισμένες στο Διαδίκτυο. Χρησιμοποίησαν αυτή την φράση σαν τίτλο για μια σειρά από συνέδρια. Τα συνέδρια αυτά συνεχίζονται και μέχρι σήμερα για να αποσαφινιστεί ο όρος και να προωθηθεί η ιδέα σε όλα τα ενδιαφερόμενα μέρη.

1.2 Χαρακτηριστικά του Web 2.0

Όπως έχει αναφερθεί το Web 2.0 ξεπερνά τα όρια της περιορισμένης πλατφόρμας ενός υπολογιστή. Ο χρήστης θα μπορεί να δρα στον Παγκόσμιο Ιστό όπως δρούσε μέχρι τώρα στον υπολογιστή του. Οι πιό ειδικοί μιλούν για έναν νέο τρόπο σχεδίασης των ιστοσελίδων ο οποίος θα βασίζεται στην διάδραση του χρήστη. Θα επιτρέπει στον χρήστη να αλλάξει τόσο το περιβάλλον της σελίδας όσο και να παρέμβει στο περιεχόμενό της. Χαρακτηριστικές εφαρμογές του Web 2.0 είναι τα κοινωνικά μέσα (social media), τα wiki και τα blog. Πολλές από τις εντολές διάδρασης που χαρακτηρίζουν την λειτουργία του Web 2.0 μας είναι ήδη γνωστές από διάφορες ιστοσελίδες social media όπως το facebook ή το youtube για παράδειγμα. Τέτοιες εκφράσεις είναι η **αναζήτηση** (search), το **tag**, η παράθεση **links** ή το **authoring** όπως λειτουργεί σε πολλά wiki όπου οι χρήστες μπορούν να δημιουργήσουν άρθρα αλλά και να ανανεώσουν ή να διαγράψουν ήδη υπάρχοντα.

1.3 Κριτικές για το Web 2.0

Το ακριβές νόημά του όρου παραμένει ανοιχτό προς αντιπαράθεση, και μερικοί ειδικοί, συμπεριλαμβανομένου και του Tim Berners Lee, έχουν αμφισβητήσει εάν ο όρος έχει κάποιο πραγματικό νόημα. Ανάμεσα σ'άλλα έχει κατηγορηθεί ότι αποτελεί εφεύρεση του μάρκετινγκ. Αναφέρεται σαν επιχείρημα πως οι τεχνολογίες που είναι σε χρήση στο ίντερνετ αναβαθμίζονται συνεχώς και η προηγούμενη μέρα πριν το web 2.0 δεν απείχε πολύ. Επίσης με την ίδια λογική, μερικοί μήνες μετά την χρήση ενός υποτιθέμενου Web 2.0, το Web 2.1 θα εμφανιζόταν. Είναι όμως αδύνατο να εκφραστεί η τεχνολογική ιδιότητα του ίντερνετ ακριβώς με έναν αριθμό μιας και το αμάλγαμα τεχνολογιών που είναι αυτή τη στιγμή σε χρήση στο ίντερνετ είναι αχανώς πολυσύνθετο. Επιπροσθέτως, λειτουργίες όπως επικοινωνία με τον χρήστη υπήρχαν πριν πολλά χρόνια, όπως η χρήση σελίδων χρήστη (home pages), τα φόρα, τα chat (IRC) και πολλά άλλα. Αυτό που μπορεί να ειπωθεί ότι άλλαξε είναι η ευκολία χρήσης, κάτι αναμενόμενο.

1.4 Ιδιωτικότητα στο Web 2.0

Ίσως το σημαντικότερο πράγμα που πρέπει να γνωρίζουμε ως χρήστες του διαδικτύου είναι πως οτιδήποτε γράφεται στο web θα μείνει για πάντα εκεί. Εργοδότες, εταιρείες η ακόμα και η κυβέρνηση μπορεί να σε εντοπίσουν καθώς περιηγήσε στο διαδίκτυο και να συλλέξουν πληροφορίες για σένα. Το γεγονός αυτό γίνεται ιδιαίτερα έντονο όσον αφορά στη στοχευμένη διαφήμιση. Αξιοσημείωτο είναι το παράδειγμα των στοχευμένων διαφημίσεων Phorm από την British Telecom την περίοδο 2006-2007 που είχε ως αποτέλεσμα, η Ευρωπαϊκή Επιτροπή να δεχτεί χιλιάδες καταγγελίες από τους χρήστες της εταιρείας οι οποίοι δήλωναν ότι η καταγραφή των online δραστηριοτήτων τους για διαφημιστικούς σκοπούς έγινε εν αγνοία τους. Ως αποτέλεσμα, η Κομισιόν κάλεσε τη Μεγάλη Βρετανία να συμμορφωθεί με την κοινοτική οδηγία για την προστασία των προσωπικών δεδομένων στις ηλεκτρονικές επικοινωνίες και να αναθεωρήσει την εθνική νομοθεσία της σε αυτή την κατεύθυνση, απειλώντας τη χώρα ακόμη και με προσφυγή στο Ευρωπαϊκό Δικαστήριο. Ωστόσο παρόμοια υπηρεσία στοχευμένης διαφήμισης έθεσε σε λειτουργία και η Google. Βέβαια, η εταιρεία ανακοίνωσε ότι η online παρακολούθηση δεν θα αφορά ευαίσθητα προσωπικά δεδομένα που αφορούν θρησκευτικές πεποιθήσεις, σεξουαλικό προσανατολισμό, θέματα υγείας, κ.λπ. Επιπλέον, η πρόσφατη δημιουργία της μηχανής αναζήτησης pipl.com φαίνεται να περιορίζει και ίσως να θέτει σε κίνδυνο τα επίπεδα της ιδιωτικότητας. Αρκεί να αναφέρουμε ότι η pipl αποτελεί μηχανή αναζήτησης μόνο για ανθρώπους. Αξίζει να σημειωθεί ότι σύμφωνα με κοινοτική οδηγία της Ε.Ε όλοι οι πάροχοι ηλεκτρονικών υπηρεσιών είναι υποχρεωμένοι να κρατούν αρχεία με τα e-mail και τα site που επισκέπτονται οι συνδρομητές τους για ένα χρόνο για λόγους ασφαλείας. Στο σημείο αυτό, ας ορίσουμε ποια στοιχεία νοούνται ως προσωπικά δεδομένα: Ως προσωπικό δεδομένο χαρακτηρίζεται οποιαδήποτε αντικειμενική ή υποκειμενική πληροφορία αφορά ένα άτομο. Αναλυτικότερα, το όνομα, η διεύθυνση, προσωπικοί αριθμοί , το γένος, η ομάδα αίματος ,εισόδημα, τραπεζικοί αριθμοί, σεξουαλικές προτιμήσεις, πολιτικές πεποιθήσεις, θρησκευτικά πιστεύω είναι μερικά παραδείγματα ιδιωτικών στοιχείων.

Παρακάτω αναφέρουμε ,εν συντομία, τα δικαιώματα που έχουν οι χρήστες σύμφωνα με το μανιφέστο για την ιδιωτικότητα στο web 2.0:

1.Κάθε χρήστης έχει το δικαίωμα να γνωρίζει εάν οι ιδιωτικές πληροφορίες συλλέγονται και σε τι βαθμό.

2.Κάθε χρήστης έχει το δικαίωμα να γνωρίζει το σκοπό για τον οποίο συλλέγονται οι πληροφορίες. Ο σκοπός πρέπει να γίνεται ξεκάθαρος από την αρχή και να μην συμπεραίνεται από το περιεχόμενο του site.

3.Καθένας έχει την κυριότητα των προσωπικών του δεδομένων. Αυτό συνεπάγεται ότι οι οργανισμοί δεν μπορούν να πουλήσουν τα στοιχεία αυτά χωρίς τη συγκατάθεση του χρήστη. Επιπλέον ο χρήστης έχει το δικαίωμα να αλλάξει τα στοιχεία αυτά.

1.5 Η ελληνική διάσταση του Web 2.0 και των Digital Media

Όσοι ανήκουν στο χώρο του Interactive Marketing κάνουν συχνά λόγο για την **ψηφιακή επανάσταση** που έχει επιφέρει το **Web 2.0** - η δεύτερη γενιά του Internet που χαρακτηρίζεται από το **social networking sites** και τη δυνατότητα των χρηστών για **online collaboration**. Η επικοινωνιακή αυτή επανάσταση έχει καταστήσει τον καταναλωτή κυρίαρχο των media και συνεπώς των διαφημιστικών μηνυμάτων που «καταναλώνει».

Το ερώτημα που γεννιέται ειναι, αν αυτή η επανάσταση η οποία έχει αρχισει να εισχωρεί και στην Ελλάδα, έχει γίνει αντιληπτή από τον **Έλληνα marketer**, τις διαφημιστικές εταιρίες, και τη συνολική Ελληνική αγορά.

Για παράδειγμαι σε "broadcast yourself websites" όπως το Youtube.com, υπάρχουν **250.000 Έλληνες** επισκέπτες που παρακολουθούν, «ανεβάζουν» και σχολιάζουν εκατοντάδες videos καθημερινά. Πολλά από αυτά είναι καθημερινά γεγονότα, σημαντικά και ασήμαντα, και θα τα ζήλευαν ακόμα και οι πιο επιτυχημένοι ρεπόρτερ.

Αξιοσημειώτο είναι το γεγονός ότι στο Google υπάρχουν **περισσότεροι από 10.000 Έλληνες ανά εβδομάδα** (πιθανά ελληνίδες μητέρες) που πραγματοποιούν αναζήτηση με τη φράση «παιδικά παιχνίδια», σύμφωνα με τα στατιστικά μιας καμπάνιας search engine marketing για λογαριασμό της Fisher Price.

Με το λανσάρισμα του **Auris** στην Ελλάδα, του νέου μοντέλου αυτοκινήτου της Toyota, έχουν γίνει σ**ε διάστημα 1,5 μήνα περίπου 80.000 αναζητήσεις** με τις λέξεις «Toyota» και «Auris», ενώ η Ελλάδα είναι η πρώτη χώρα παγκοσμίως (!) σε αναζητήσεις στο Google με τις παραπάνω .

Στο pathfinder.gr υπάρχουν **9.000 κοινότητες/ "clubs" με 170.000 μέλη** που ανταλλάσσουν απόψεις για μια πλειάδα θεμάτων: από τον αθλητισμό και την μόδα μέχρι την πολιτική και τη μουσική. Στο zoo.gr υπάρχουν 190.000 άτομα που κάνουν networking, ενώ στο MSN Live Messenger **542.000 Έλληνες** κάνουν chat και άλλοι **320.000 στο Yahoo! Messenger**. Στο MSN Live Spaces **285.000 επισκέπτες** διαβάζουν τις προσωπικές σελίδες φίλων τους, δημιουργούν photo albums, γράφουν, ανεβάζουν videos.Επίσης, σε άλλα community και networking sites όπως στο hi5.com, στο myspace.com, στο joy.gr, και στο asmallworld.net χιλιάδες Έλληνες καθημερινά ασχολούνται με το δίκτυο φίλων που έχουν σχηματίσει, δημιουργούν περιεχόμενο, ανταλλάσσουν μηνύματα και έρχονται σε επαφή μεταξύ τους με τρόπο διαδραστικό.
Υπάρχουν **70.000.000 Blogs** παγκοσμίως (user generated websites σε στυλ ημερολογίου), **δημιουργούνται 120.000 νέα blogs** και γράφονται **1.500.000 άρθρα καθημερινά.** Παράλληλα, υπάρχουν blogs τα οποία δέχονται χιλιάδες επισκέψεις, ενώ εκατοντάδες είναι τα comments που ανταλλάσσουν οι χρήστες.
Το pestaola.gr έχει **40.000 επισκέπτες το μήνα**, το blog του γνωστού συγγραφέα Νίκου Δήμου δέχτηκε σε 15 μήνες 1.000.000 επισκέψεις και 90.000 σχόλια χρηστών σε 332 άρθρα του.

Καταλήγουμε λοιπόν στο συμπέρασμα ότι οι συνήθειες των Ελλήνων καταναλωτών, σε σχέση με την καθημερινή έκθεσή τους στα media, έχουν αλλάξει σημαντικά.Η τεχνολογία επηρεάζει όλο και περισσότερο την ρουτίνα κατανάλωσης media που έχει

καθένας από εμάς . Κι αυτό γιατί η τεχνολογία είναι που μας δίνει την ευκαιρία να μην είμαστε πλέον απλά παθητικοί αποδέκτες μηνυμάτων, αλλά να επιλέγουμε και να διαμορφώνουμε το περιεχόμενο και την πληροφορία που επιθυμούμε.

1.6 Virtual community

Μία εικονική κοινωνία (**virtual community**, **e-community** ή **online community)** είναι ένα σύνολο ανθρώπων που προτίστως αλληλεπιδρούν μεταξύ τους από διαύλους επικοινωνίας (κατά κύριο λόγο ψηφιακά), όπως είναι τα newsletters, το τηλέφωνο,το email, οι υπηρεσίες social network ή οι υπηρεσίες άμεσων μηνυμάτων (instant messages), με επαγγελματικούς, κοινωνικούς, εκπαιδευτικούς ή άλλους σκοπούς. Αν ο δίαυλος επικοινωνίας είναι το διαδίκτυο, τότε οι κοινωνίες αυτές ονομάζονται ψηφιακές κοινωνίες ή αλλιώς τα σε όλους μας γνωστά *online communities*. Έχουν υπάρξει και κατά το παρελθόν υποκατάστατα επικοινωνίας μεταξύ ανθρώπων που γνωρίζονται μεταξύ τους και στην καθημερινή τους ζωή. Οι ψηφιακές κοινωνίες εμφανίζονται με ποικίλους τρόπους και διαστάσεις όπως τα κοινωνικά fora, τα chat rooms, και τις υπηρεσίες δικτυακής επικοινωνίας κειμένου, βίντεο και ήχου.

Οι ψηφιακές ή διαδικτυακές κοινωνίες, χρησιμοποιούνται από ποικίλα σύνολα κοινωνικών και επαγγελματικών ομάδων, μέσα στο διαδίκτυο. Δεν σημαίνει απαραίτητα ότι υπάρχει ένας δυνατός δεσμός ανάμεσα στα μέλη, ωστόσο ο Howard Rheingold, συγγραφέας του ομώνυμου βιβλίου, αναφέρει ότι οι εικονικές κοινωνίες διαμορφώνονται «όταν οι άνθρωποι διατηρούν δημόσιες συζητήσεις για μεγάλο χρονικό διάστημα, με *επαρκές ανθρώπινο συναίσθημα*, για να διαμορφώσουν δίκτυα προσωπικών σχέσεων». Μία λίστα διανομής e-mail μπορεί να έχει εκατοντάδες μέλη και η επικοινωνία που λαμβάνει χώρα μπορεί να είναι απλώς ενημερωτική (ερωτήσεις και απαντήσεις δημοσιεύονται), όμως τα μέλη της μπορεί να παραμένουν σχετικά άγνωστα και το ποσοστό του κύκλου εργασιών μπορεί να είναι υψηλό. Αυτό συμφωνεί με τη φιλελεύθερη χρήση του όρου *κοινότητα*.

Οι εικονικές κοινότητες μπορεί να συνθέτουν της τεχνολογίες του web 2.0 με την κοινότητα, και γι’ αυτό έχουν περιγραφεί ως Κοινότητα 2.0, αν και ισχυροί δεσμοί κοινότητας έχουν σφυρηλατηθεί διαδικτυακά από τις πρώτες μέρες του USENET. Οι διαδικτυακές κοινότητες εξαρτώνται από την κοινωνική αλληλεπίδραση και την ανταλλαγή μεταξύ των διαδικτυακών χρηστών. Αυτό τονίζει το στοιχείο της αμοιβαιότητας του άγραφου κοινωνικού συμβολαίου μεταξύ των μελών της κοινότητας. Κάποιες από τις πρώτες μορφές των ιστοσελίδων των εικονικών κοινοτήτων του web 1.0, περιλαμβάνουν τις: Theglobe.com (1994), Geocities (1994) και Tripod (1995). Αυτές οι πρώτες κοινότητες εστίαζαν στο να φέρνουν ανθρώπους κοντά για να αλληλεπιδρούν μεταξύ τους μέσω των chat rooms, και να μοιράζονται προσωπικές πληροφορίες και ιδέες γύρω από οποιαδήποτε θέματα μέσω των εργαλείων δημοσίευσης της προσωπικής ιστοσελίδας, οι οποίες ήταν πρόδρομοι των

ιστολογίων και του φαινομένου της κοινωνικής δικτύωσης. Το κύμα της web 2.0 διαδικτυακής κοινότητας έφτασε στις αρχές του 2000 και χαρακτηρίζεται απαραίτητα από εικονικές κοινότητες όπως το Flickr, το Facebook και το Del.icio.us. Μια παρόμοια τάση αρχίζει να αναδύεται στις επιχειρήσεις, όπου οι διαδικτυακές ή εικονικές κοινότητες εδραιώνονται. Αυτές οι κοινότητες μπορεί να είναι οργανωτικές, περιφερειακές ή τοπικές ανάλογα με την επιχείρηση. Από τεχνικής άποψης, τα εργαλεία λογισμικών αφθονούν για να δημιουργήσουν και να γαλουχήσουν αυτές τις κοινότητες περιλαβάνοντας τα: BigTent, Yahoo Groups, Google Groups, LISTSERV, Microsoft Sharepoint και Lotus Connections.

Η εκρηκτική διάδοση του Διαδικτύου από τα μέσα της δεκαετίας του ’90 προώθησε τη διάδοση των εικονικών κοινοτήτων. Η φύση αυτών των κοινοτήτων είναι πολυποίκιλη, και οι παροχές που οραματίστηκε ο Rheingold δεν πραγματοποιούνται κατ’ανάγκη, ή επιδιώκονται, από πολλούς. Την ίδια στιγμή, είναι μάλλον κοινότυπο να δούμε ιστορίες κάποιου σε ανάγκη για ειδική βοήθεια ή σε αναζήτηση μιας κοινότητας, επωφελούμενος από τη χρήση του Διαδικτύου.

Διαφορετικές εικονικές κοινότητες έχουν διαφορετικά επίπεδα αλληλεπίδρασης και συμμετοχής μεταξύ των μελών τους. Αυτό κυμαίνεται από την προσθήκη σχολίων ή ετικετών σε ένα blog ή ένα πίνακα μηνυμάτων έως τον ανταγωνισμό με άλλους ανθρώπους σε διαδικτυακά βιντεοπαιχνίδια, όπως είναι το MMORGPGs. Πολλές κοινότητες που διαχειρίζονται από την Homeowner Associations χρησιμοποιούν τους συναιτερισμούς ιδιοκτητών ακινήτων για να διευκολύνουν όλες τις επικοινωνίες, τις επιχειρηματικές συνδιαλλαγές, και τις κοινωνικές προσπάθειες κάθε είδους. Χωρίς να διαφέρουν από τις παραδοσιακές κοινωνικές ομάδες ή συλλόγους, οι εικονικές κοινότητες συχνά χωρίζονται σε κλίκες ή χωρίζονται για να δημιουργήσουν νέες κοινότητες. Η συγγραφέας Amy Jo Kim επισημαίνει μία ενδεχόμενη διαφορά μεταξύ των παραδοσιακών δομημένων διαδικτυακών κοινοτήτων (πίνακες μηνυμάτων, chat rooms, κ.λπ.), και περισσότερο ανθρωποκεντρικά, πρωταρχικου/αυξάνοντος επιπέδου κοινωνικών εργαλείων (blogs, λίστες φίλων άμεσων μηνυμάτων), και προτείνει ότι τα τελευταία κερδίζουν σε δημοτικότητα.

Η ενσωμάτωση της εικονικής κοινότητας στις εμπειρίες της καθημερινής ζωής και τον προβληματισμό της για την επιρροή σχετικά με τις πρακτικές της επικοινωνίας και του ρυθμού σχηματισμού ταυτότητας κάνουν τη διαδικτυακή κοινότητα μια κολοσσιαία επιχείρηση έρευνας η οποία απαιτεί συνεχή έρευνα και θεωρία.

1.7 Ο κύκλος ζωής των μελών για τις εικονικές κοινότητες

Ένας κύκλος ζωής των μελών για τις διαδικτυακές κοινότητες προτάθηκε από την Amy Jo Kim (2000). Επισημαίνει ότι τα μέλη των εικονικών κοινοτήτων ξεκινούν τη ζωή τους σε μια κοινότητα ως επισκέπτες, είτε ως παρατηρητές. Έπειτα από τη υπέρβαση ένος ορίου, οι άνθρωποι γίνονται αρχάριοι και συμμετέχουν στη ζωή της κοινότητας. Μετά τη συμβολή τους για μια παρατεταμένη χρονική περίοδο γίνονται τακτικοί. Σε περίπτωση που σπάσουν και άλλο όριο γίνονται ηγέτες, και τη στιγμή που θα συμβάλουν στην κοινότητα για κάποιο χρονικό διάστημα γίνονται γηραιοί. Αυτός ο κύκλος της ζωής μπορεί να εφαρμοστεί σε πολλές εικονικές κοινότητες, οι σημαντικότερες των οποίων είναι τα bulletin boards, αλλά επίσης και τα blogs και οι κοινότητες βασισμένες στα wikis όπως η Wikipedia.

Ένα παρόμοιο μοντέλο μπορεί να βρεθεί στις εργασίες των Lave και Wenger, οι οποίοι επιδεικνύουν έναν κύκλο του πως οι χρήστες ενσωματώνονται μέσα σε εικονικές κοινότητες χρησμιοποιώντας τις αρχές της νόμιμης περιφερειακής συμμετοχής. Προτείνουν πέντε τύπους τροχιών μεταξύ των μαθησιακών κοινοτήτων:

1. Insider (π.χ. Τακτικός χρήστης/μέλος) – Ιδιαίτερα προσκολλημένος στην virtual community και ενεργός χρήστης σε όλες της τις εκφάνσεις.
2. Οδηγός/ Ηγέτης – Ηγέτης που υποστηρίζει και υποβοηθά την έμπρακτη συμμετοχή και τις διαδράσεις
3. Εξερχόμενος (π.χ. Βετεράνος) – Στο τέλος του Learning/ interacting curve, που λόγω νέων του κοινωνικών υποχρεώσεων πλέον έχει αρχίσει να διαφοροποιείται και να απογαλακτίζεται από την ψηφιακή μικρο-κοινωνία
4. Περιφεριακός χρήστης (γνωστός και ως Lurker) – Μία εξωτερική αδόμητη συμμετοχή
5. Εισερχόμενος χρήστης (π.χ. Αρχάριος) – Ο νεοφερμένος επενδύεται στην κοινότητα και κατευθύνεται προς την πλήρη συμμετοχή

1.8 Τι μας περιμένει στο μέλλον

Ο ντόρος που αναπτύσσεται σχετικά με το Web 3.0 είναι ήδη τεράστιος, αλλά, ως συνήθως, κάποιος θα πρέπει να φιλτράρει όλη τη διαφημιστική εκστρατεία, τους αυτοεξυπηρετούμενους PR, τις παλιομοδίτικες ανοησίες, τις μοντέρνες πολυλογίες του μάρκετινγκ και άλλους θορύβους. Θα βρει τότε κάποια γεγονότα με τα οποία μπορεί να συσχετιστεί το Web 3.0 και να καταλάβει τι περίπου συμβαίνει. Το πρώτο πράγμα που πρέπει κάποιος να θυμάται είναι πως, όπως το «Web 2.0», ο όρος Web 3.0 δεν είναι ένας επίσημος όρος οποιουδήποτε είδους, δεν αντιπροσωπεύει κανένα ιδιαίτερο πρωτόκολλο ή πρότυπο, δεν ανήκει σε κανέναν – και χρησιμοποιείται, καταχρηστικά χωρίς να βγάζει νόημα από τον καθένα. Είναι απλούστατα ένας αυθαίρετος «αριθμός έκδοσης», ο οποίος, στην καλύτερη περίπτωση, περιγράφει πως είναι δομημένο το Διαδίκτυο και πως παρέχει υπηρεσίες, τουλάχιστον από τη συγκεκριμένη, ορισμένη χρονικά στιγμή που αντιπροσωπεύει το τέλος του 2.0 και την αρχή του 3.0.

Ορισμένες φορές ονομάζεται «σημαντικός Ιστός», αλλά ίσως ο λιγότερο χρησιμοποιούμενος όρος «παντού» είναι πιο περιγραφικός. Το νέο λοιπόν σενάριο του Web 3.0, αφορά παγκοσμιοποιημένο computing, κατά μεγάλο μέρος cloud computing όπου ένας τελικός χρήστης με ένα "χαζό τερματικό" (με όχι κάτι παραπάνω από μία οθόνη και ένα πληκτρολόγιο), τρέχει πολύπλοκές εφαρμογές μέσω τεχνολογίας πληροφορικής-σύννεφου καθώς και υπηρεσίες με την ίδια μέθοδο. Το iPhone της Apple και το iPad είναι όλα παραδείγματα πρώην αυτόνομων συσκευών που ενσωματώθηκαν στον Ιστό, και συνδέουν ανθρώπους σε πραγματικό χρόνο και με πολύ απλό τρόπο με - λοιπόν, με τα πάντα, από βιβλιοθήκες και καταστήματα έως άλλους ανθρώπους, οπουδήποτε στον κόσμο.

Από το «Μόνο για ανάγνωση» στην Αλληλεπίδραση

Ένας από τους πραγματικούς «πατέρες» του Ιστού ήταν ο Tim Berners-Lee, ο οποίος είχε τη δική του άποψη του πως αναπτύχθηκαν η τεχνολογία και το Διαδίκτυο. Η πρώτη φάση του Ιστού είχε ικανότητες «μόνο για ανάγνωση». Ήταν ουσιαστικά μία εμπειρία βασισμένη στη θέαση έως ότου έφτασε η λειτουργία ανάγνωσης-εγγραφής (σίγουρα, ονομάστε το Web 2.0) η οποία περιελάμβανε λειτουργίες που καθιστούσαν δυνατή την προσφορά, τη συνεργασία, τη δημιουργία περιεχομένου και της διαδραστικότητας. Το επόμενο βήμα στην έκδοση του οράματος του Berners-Lee, το Web 3.0, αναγγέλεται ως «νέο έδαφος», όπου οι χρήστες μπορούν να συναρμολογήσουν και να εκτελέσουν τις δικές τους εφαρμογές, να δημιουργήσουν όλα τα είδη συνεταιριστικών και συνεργατικών επιχειρήσεων, και πραγματικά να βάλουν τις ιδέες τους σε δράση, αντί απλά να «ανεβάζουν» υλικό σε εκείνη ή την άλλη ιστοσελίδα.

Άνθρωποι με χρήματα που επένδυσαν σε άλλες , ακόμα χρήσιμες συσκευές – τηλέφωνα, PDA, μηχανήματα φαξ, κ.ά. - δε χρειάζεται να ανησυχούν ότι το Web 3.0 θα τους απαρχαιώσει. Εκτός του ότι επιτρέπει στους χρήστες να δημιουργούν τα δικά τους εργαλεία, το Web 3.0 είναι ακόμη ένα βήμα στην εξέλιξη της χρήσης και της

αλληλεπίδρασης στην οποία το Διαδίκτυο κρατά πολλαπλές βάσεις δεδομένων και περιεχόμενο το οποίο θα είναι προσβάσιμο από πολλές συσκευές που δεν υποστηρίζουν προγράμματα περιήγησης και εφαρμογές. Οι προφανείς χρήσεις θα είναι βίντεο που θα μεταφέρεται από έναν Η/Υ σε μια τηλεόραση, κορνίζες που θα λαμβάνουν ασύρματες ενημερώσεις από μία διαδικτυακή ή τοπική εφαρμογή φωτογραφιών, και τηλέφωνα που εκθέτουν αντικείμενα που προτείνονται από τις αξιόπιστες πηγές σας – τους φίλους σας, τις αξιολογήσεις των ιστοσελίδων, τους ειδικούς – ενώ εσείς κάνετε τις αγορές σας.

Από την Πληροφορία στη Γνώση

Εκτός από τα παραπάνω χαρακτηριστικά, το Web 3.0, πιστεύεται ότι συμπεριλαμβάνει και άλλα σημαντικά βήματα. Για ένα πράγμα, όλων των ειδών οι εισαγωγές είναι δυνατές, το οποίο σημαίνει ότι όλων των ειδών οι νέοι συνδυασμοί καθίστανται δυνατοί. Το περιεχόμενο μπορεί να γίνει ευρύτερα σχετικό όταν σχετίζεται με GPS, έτσι ώστε η κοινωνική δικτύωση, για παράδειγμα, μπορεί να ενισχυθεί με τη γνώση του ποιος είναι που και τι κάνει.

Το πιο σημαντικό, θα πάρετε περισσότερο και καλύτερο έλεγχο των δεδομένων σας και θα είστε ικανοί να ιδρύσετε έναν αριθμό από συστήματα προσωποποίησης για να «τυλίξετε» τις προσωπικές σας πληροφορίες με διαφορετικών επιπέδων και τυπών προστασίας – έτσι ώστε να το μοιράζεστε ευρέως, στενά ή και καθόλου. Με την πάροδο του χρόνου, η ακρίβεια των προτάσεων και της αξιοπιστίας των συστημάτων κατάταξης θα μας βοηθήσει να καθορίσουμε ποιες πηγές πληροφοριών να πάρουμε στα σοβαρά και ποιες να αποφύγουμε.

Από τα Εταιρικά Φαξ στα Διαδικτυακά Παιχνίδια

Με την άνοδο των «συνεργαζόμενων διαδικτυακών εφαρμογών» θα είμαστε σε θέση να χρησιμοποιήσουμε όλες τις διαφορετικές επιφάνειες εργασίας, τους εξυπηρετητές και τις συσκευές κινητών και τις εφαρμογές – τηλέφωνα, μηχανήματα φαξ και διαδικτυακές υπηρεσίες φαξ, άμεσα μηνύματα, συσκευές τηλεειδοποίησης – και να τα αξιοποιήσουμε από ένα μόνο παράθυρο του προγράμματος περιήγησης στην επιφάνεια εργασίας, το smartphone σας ή φορητή συσκευή μας. Όλα αυτά θα λάβουν χώρα σε ένα always-on, always-everywhere περιβάλλον, με με τη λειτουργικότητα ενσωματωμένα στο υλικό μερικές φορές, μερικές φορές σε λογισμικό, μερικές φορές και στα δύο - έτσι ώστε όταν χρειαστεί να τακτοποιήσετε την επιχείρησή σας χώρις να «κατεβάσετε» τη δυνατότητα, θα είστε σε θέση να το πράξετε.

Μαζί με περισσότερες από τις οπτικές και φωνητικές υπηρεσίες που ήδη έχουν αρχίσει να πολλαπλασιάζονται, θα υπάρχουν περισσότερες αλληλεπιδράσεις ειδώλων στον αναπτυσσόμενο εικονικό κοινωνικό δικτυακό κόσμο. Αυτό θα οδηγήσει σε κοινωνικά ταξίδια αγορών και σε εικονικά παιχνίδια ριάλιτι πολύ πέρα από όσα γίνονται αυτή τη στιγμή.

Στα μέσα του 2009, η «Wall Street Journal» δημοσίευσε μια ιστορία για την ανάπτυξη των δυνατοτήτων του Web 3.0 και την υπόσχεσή της για όλο και

μεγαλύτερη διασύνδεση μεταξύ των τεχνολογιών, προϊόντων, υπηρεσιών και ανθρώπων. Η ιστορία μάλιστα μας έδωσε ένα μέτρο με το οποίο μπορεί να μετρηθεί η επιτυχία του Web 3.0, αν τελικά επιτύχει. Αν, όπως το θέτει το WSJ, «η χρήση των υπολογιστών μπορεί να ενσωματωθεί και να γίνει αόρατη όπως το ηλεκτρικό και εξίσου σημαντική» - και εμείς μπορούμε να το αποδώσουμε στο νέο και βελτιωμένο Ιστό – θα γνωρίζουμε πως η υπόσχεση ανταποκρίθηκε στις προσδοκίες!

Κεφάλαιο 2ο
Η κοινωνική πλατφόρμα Dolphin Community Builder 7.0

2.1 Είδη κοινωνικής πλατφόρμας

Μετά την εξάπλωση των ψηφιακών κοινωνιών, λογικό ήταν να επέλθει και η διαφοροποίηση τους (σε μεγάλο βαθμό παρομοιάζεται μάλιστα με την βιοποικιλότητα διαφόρων ανοιχτών συστημάτων). Έτσι λοιπόν, έχουμε πλέον μια πλειάδα ειδών ψηφιακών κοινωνιών, όπου τα μέλη τους είτε ενεργά είτε έμμεσα, συμβάλλουν στην πραγμάτωση του σκοπού τους.
Μερικά είδη τέτοιων ψηφιακών κοινωνιών είναι:

1)Chat rooms:
Ειδικά κανάλια γραπτής επικοινωνίας με ιδιαίτερη έμφαση στα άμεσα γραπτά μηνύματα.

2)IMs:
Η εξειδίκευση των παραπάνω σε one-on-one περιπτώσεις χρήσης

3)Bulletin Boards:
Τα γνωστά σε όλους μας fora, ξεκίνησαν με κοινωνική κυρίως εστίαση τόσο σε ζητήματα διαλόγου όσο και σε αυτοσκοπό τους.

4)Wikis:
Συναντόνται κυρίως ως εγκυκλοπαίδιες, πλέον αποκεντρωμένες από την βασική τους αρχή που τα δημηούργησε (βλ. Wikipedia)

5)VR Communities
Δυστυχώς δραστηριοποιούνται όπου το κέρδος είναι αυξημένο, δηλαδή στον τομέα της «ενήλικης» διασκέδασης.

6)Social Comunities
Με ιδιαίτερη έμφαση στα dating συστήματα, έως και τις αρχές της εμφάνισης του Facebook, όπου πλέον συναντόνται σε πιο απενοχοποιημένες μορφές (βλ. Twitter)

7)Social Mashups
Η πρώτες ενδείξεις του Web 3.0 στη ζωή μας, με εμφάνιση «συνοθυλευμάτων» χρηστικότητας όπως το NetBibes

2.2 Η πλατφόρμα Dolphin

Η πλατφόρμα αναπτύσσεται διαρκώς από το 2000 από την εταιρία Boonexκαι πρόκειται για μια καλή περίπτωση MashUp ψηφιακής κοινωνίας. Περιέχει μεταξύ άλλων τις εξείς δυνατότητες:

- Media Sharing, όλων των ειδών εικόνων, βίντεο, και μουσικών κομματιών αλλά και αρχείων.
- Έμφυτο σύστημα match making που μπορεί κάλλιστα να αξιοποιηθεί και για εναλλακτικές λειτουργίες
- Εξελιγμένο σύστημα ασφαλείας και ασφάλειας κατά την ανάπτυξη
- Εξελιγμένο σύστημα Modular development
- Σύστημα Groups για κοινωνικές ομάδες.
- Σύστημα Data Fetching για ευκολότερη ανεύρεση και ενσωμάτωση δεδομένων άλλων σελίδων.
- Templating Σύστημα για καλύτερο διαχωρισμό μεταξύ όψης/ δεδομένων
- Σύστημα Web Feeds για καλύτερη διανομή δραστηριοτήτων στο διαδίκτυο
- Cluster Τεχνολογία για ανάπτυξη σε επίπεδο εκατομμυρίων χρηστών.

Όπως βλέπουμε λοιπόν, συγκεντρώνει τα συμαντικότερα στοιχεία και όχι μόνο, όλων των ειδών των Social Communities.

2.3 Τεχνολογίες

Η πλατφόρμα Dolphin 7.0 υποστηρίζει πλήρως τα κάτωθι:

- PHP 5.2.0 ή νεότερες εκδόσεις
- MySQL 4.1.2 ή νεότερες εκδόσεις
- GD Library ή εναλλακτικά την βιβλιοθήκη επεξεργασίας γραφικών ImageShack
- SendMail και Postfix για την αποστολή mail προς τους χρήστες
- ***mbstring***, ***domxml σε περίπτωση που δεν έχουμε εγκατεστημένη την τελευταία έκδοση της PHP***
- ***GNU/GPL License***
- ***Open Source***

Η πλατφόρμα Dolphin 7.0 είναι μεν εμπορική πλατφόρμα (για την εμπορική χρήση της χρειάζεται η αγορά άδειας), αλλά όμως έχει εκδοθεί υπό το πρωτόκολλο GNU/GPL, δίνοντας έτσι την δυνατότητα να αξιοποιηθεί ως πλατφόρμα ανοιχτού κώδικα, αφού όλος πλέον δίδεται μη κρυπτογραφημένος από την εταιρία που την ανέπτυξε.

2.4 Πληροφορίες Ανάπτυξης Προγραμματιδίων

Τα βασικά στοιχεία της πλατφόρμας με τα οποία μπορούμε να δουλέψουμε για την παραγωγή προγραμματιδίων,αναφέρονται παρακάτω:

- **Privacy** – Το περιεχόμενο των προγραμματιδίων μπορεί να υπόκειται σε κανόνες συμπεριφοράς ιδιωτικότητας από τον χρήστη, όπως αυτά καθορίζονται από την κλάση **BxDolPrivacy**.

- **Alerts** – Η πλατφόρμα διεγείρει alerts για διαφορετικές διαδράσεις του χρήστη, που μπορούν να γίνουν αντιληπτά από κάθε προγραμματίδιο, σε αντιστοίχηση με τοπολογίες listeners.Επίσης,τα alerts,μπορούν να παραχθούν και από τα προγραμματίδια, όπως περιγράφονται στην κλάση **BxDolAlerts**.

- **Forms** – Ενοποιημένες φόρμες στην πλατφόρμα, με αυτόματο έλεγχο επικύρωσης/απόρριψης και αποθήκευσης στην βάση δεδομένων, μπορούν επίσης να χρησιμοποιηθούν, όπως αυτά περιγράφονται στην κλάση **BxDolForm**.

- **Page builders** – Μπορούμε να σχεδιάσουμε νέες σελίδες, χρησιμοποιώντας τον εγγενή page builder, που βρίσκεται στο κέντρο διαχείρισης της πλατφόρμας.

- **Cron** – Η πλατφόρμα υποστηρίζει επαναλαμβανόμενες λειτουργίες, μέσω του συστήματος Cron των Unix συστημάτων, χωρίς να χρειάζεται η χειροκίνητη εγκατάστασή τους μέσω SSH ή cPanel

- **Tags** –Αυτόματη διαχείριση όλων των ετικετών για οποιοδήποτε είδος περιεχομένου και media μπορούμε να επιτύχουμε με την αξιοποίηση της κλάσης **BxDolTags**.

- **Categories** – Αντίστοιχη αξιοποίηση του εγγενούς συστήματος των «ομάδων», για κατηγοριοποίηση όλων των ειδών περιεχομένου, μέσω της κλάσης **BxDolCategories**.

- **Albums** – Λειτουργία για την χρήση Albums για το περιεχόμενο παντός τύπου και διαχείρισής τους από τον χρήστη, μέσω της κλάσης **BxDolAlbums**.

- **ACL (Επίπεδα Χρηστών και προσβάσεων)** – Ο διαχειριστής της πλατφόρμας είναι σε θέση να ορίζει επίπεδα χρηστών και τμήματα της πλατφόρμας στα οποία κάθε επίπεδο έχει πρόσβαση. Επίσης μπορεί να καθορίσει ρόλους για τους χρήστες και ο έλεγχος γίνεται μέσω της ρουτίνας **checkAction** function που συγχωνεύεται στο αρχείο **membership_levels.inc.php**.

- **Comments** – Για σχόλια στα προγραματίδιά μας, ανατρέχουμε στην κλάση **BxDolCmts**.

- **Voting** –για αξιοποίηση ψηφοφοριών για τα προγραμματίδιά μας χρησιμοποιούμε τις εγγενείς ρουτίνες της κλάσεως **BxDolVoting**.

- **Calendar** –Ενεργοποιούμε την δυνατότητα ημερολογίου για κάθε προγραμματίδιό μας με χ΄ρηση της κλάσης **BxDolCalendar**.

- **Member Menu** – Διαχειρίσιμο μενού επιλογών χρηστών, με δυνατότητα πρόσθεσης/διαχείρισης και αφαίρεσης επιλογών, με πλήρως διαχειριζόμενα υπομενού, μέσω της κλάσης **BxDolMemberMenu**.

- **Top Menu** – Το κεντρικό μενού επιλογών της πλατφόρμας, διαχειριζόμενο μέσω του πίνακα **sys_menu_top** της βάσης δεδομένων.

- **Admin menu** Το κεντρικό μενού επιλογών του κέντρου Διαχείρισης είναι επεκτάσιμο μέσω του πίνακα **sys_menu_admin** της βάσης δεδομένων.

- **Admin Top Menu** – Το άνωθεν ευρισκόμενο μενού του Διαχειριστή, φιλοξενείτε στον πίνακα **sys_menu_admin_top** και μπορεί να περιέχει εξωτερικούς συνδέσμους.

- **Admin Dashboard** – Κεντρικός πίνακας διαχείρισης στον πίνακα **sys_admin_dashboard** της βάσης δεδομένων, επεκτάσιμο μέσω της κλάσης **BxDolAdminDashboard**.

- **Bottom Menu** – Το κάτωθεν ευρισκόμενο μενού φιλοξενείται στον πίνακα **sys_menu_bottom**της βάσης δεδομένων.

- **Email templates** – Μάσκες εφαρμογής των ηλεκτρονικών γραμμάτων που αυτομάτων αποστέλλονται μέσω της πλατφόρμας, είτε δικές μας παραμετροποιημένες με χρήση της κλάσης **BxDolEmailTemplates**. Όλα πλήρως διαχειριζόμενα μέσω του πίνακα διαχείρισης της πλατφόρμας

- **Paginate** – Έτοιμη σελιδοποίηση παντός τύπου περιεχομένου με χρήση της κλάσης **BxDolPaginate**.

- **Permalinks** – Για φιλικές προς τον χρήστη διευθύνσεις και συνδέσμους, αξιοποιούμε την κλάση **BxDolPermalinks** της πλατφόρμας.

- **RSS** – Αυτόματη δημιουργία RSS XML για όλα τα είδη περιεχομένου, μέσω της κλάσης **BxDolRssFactory** της πλατφόρμας.

- **Site Search** – Ιδιότητα ευρετηρίου και αναζήτησης όλων των ειδών περιεχομένου, μέσω της κλάσης **BxDolSearch** της πλατφόρμας.

- **Services** – Υπηρεσίες κλήσης/ διάδρασης μεταξύ των προγραμματιδίων και μεταξύ προγραμματιδίου/πλατφόρμας, όπως αυτή περιγράφεται στην κλάση **BxDolService** της πλατφόρμας.

- **Subscriptions** – Κάθε χρήστης μπορεί να «εγγραφθεί» ούτως ώστε να λαμβάνει ειδοποιήσεις για ενημερώσεις χρηστών και περιεχομένου. Για χρήση της δυνατότητας αξιοποιούμε την κλάση **BxDolSubscription**.

- **Images resizing** – Για αυξομείωση και διαχείριση μεγέθους εικόνων και φωτογραφιών, κάνουμε χρήση της **BxDolImageResize** κλάσης του συστήματός μας.

- **Action buttons** Για να εμφανίσουμε κουμπιά ενεργειών που ο χρήστης μπορεί να κάνει, αξιοποιούμε τις ρουτίνες **BxBaseFunctions::genObjectsActions**, **BxBaseMenu::setCustomSubActions** και τον πίνακα **sys_objects_actions** της βάσης δεδομένων μας.

- **Authentication types** – Μοναδικοποίηση της εισόδου ενός χρήστη στο σύστημα, της πρωταρχικής μεθόδου αλλά και της μεθόδου που χρησιμοποιείται από το προγραμματίδιο Facebook Connect, μπορούμε να προσθέσουμε δικές μας μοναδικοποιήσεις, μέσω του πίνακα **sys_objects_auths** της βάσης δεδομένων.

- **Injections** – Εισαγωγή τμημάτων κώδικα σε διάφορα σημεία των σελίδων της πλατφόρμας, με χρήση της κλάσης **BxDolTemplate**.

- **Site stat** – Για εμφάνιση στατιστικών στοιχείων σε μέρος της πλατφόρμα,. χρησιμοποιούμε τον πίνακα **sys_stat_member** της βάσης δεδομένων.

- **User stat** – Για εμφάνιση στατιστικών στοιχείων που αφορούν κάθε χρήστη μεμονομένα, χρησιμοποιούμε τον πίνακα **sys_stat_site** της βάσης δεδομένων.

Κεφάλαιο 3ο

Ανάπτυξη Προγραμματιδίου στην πλατφόρμα Dolphin

3.1 Δομή Προγραμματιδίου (εγκαταστάσιμης/ λειτουργικής μορφής)

Η δομή κάθε προγραμματιδίου βρίσκεται μέσα στον κύριο φάκελο της πλατφόρμας (public_html) και στον ειδικό φάκελο Modules, υπό τον φάκελο του εκάστοτε δημιουργού (εν προκειμένω στον υποφάκελο **bloggie** ,όνομα και φάκελος υπό τον κεντρικό φάκελο προγραμματιδίων **modules/lamda/**. Το Bloggie είναι το όνομα του προγραμματιδίου,ενώ είναι καλή τεχνική το να ονομάζουμε τον φάκελο της εκάστοτε εφαρμογής με το ίδιο της το όνομα, όπου θα στεγάζονται τα αρχεία της.

Η εφαρμογή θα δίνει την δυνατότητα σε κάθε χρήστη της πλατφόρμας να ανοίγει όσα και όποια δικά του ιστολόγια επιθυμεί.

Η δομή των υποφακέλων/αρχείων του προγραμματιδίου θα είναι η κάτωθι:

bloggie/classes
Φάκελος με όλες τις κλάσεις του προγραμματιδίου

bloggie/classes/MeBlggConfig.php
Η κλάση δειχειριστικού περιεχομένου του προγραμματιδίου.Κληρονομεί την κλάση συστήματος και οποιεσδήποτε διαχειριστικές μεταβλητές χρειαστούν

bloggie/classes/MeBlggDb.php
Κλάση της βάσης δεδομένων του προγραμματιδίου. Όλες οι ρουτίνες SQL βρίσκονται σε αυτή την κλάση, ούτως ώστε να είμαστε σε θέση να καλέσουμε ήδη έτοιμες μεθόδους της πλατφόρμας

bloggie/classes/MeBlggModule.php
Κλάση τύπου Controller. Όλες οι βασικές λειτουργίες γίνονται σε αυτήν.

bloggie/classes/MeBlggTemplate.php
Κλάση template. Κληρονομεί την κεντρική κλάση templating system της πλατφόρμας και μας δίνει την δυνατότητα να χρησιμοποιούμε templates για να διαχωρίσουμε την όψη του προγραμματιδίου από τα δυναμικά δεδομένα τα οποία αυτό πιθανόν να καλεί.

bloggie/install
Φάκελος που χρειάζεται για την εγκατάσταση του προγραμματιδίου στην μητρική πλατφόρμα. Περιέχει αρχεία και ρουτίνες SQL που χρειάζονται ώστε το προγραμματίδιο να γίνεται αντιληπτό από την υπόλοιπη πλατφόρμα

bloggie/install/config.php
Διαχειριστικό αρχείο εγκατάστασης. Δίνει πληροφορίες στον εγκαταστάτη της πλατφόρμας όπως την ανακατασκευή γλωσσών και μενού επιλογών.

bloggie/install/info
Πληροφορίες εγκατάστασης προγραμματιδίου.παρέχει πληροφορίες κατά τη ν εγκατάσταση/απεγκατάσταση του προγραμματιδίου, καθώς και ειδικές οδηγίες που χρειάζονται, όπως π.χ. προαπαιτούμενα προγραμματίδια καθώς και επιτυχία/αποτυχία εγκατάστασης.

bloggie/install/installer.php
Εγκαταστάτης προγραμματιδίου. Εδώ προσθέτουμε οποιαδήποτε παρακαμπτήρια τμήματα κώδικα χρειάζονται για την εκγατάσταση, αλλιώς, περαμένει ως έχει, κληρονομόντας την αντίστοιχη κλάση της πλατφόρμας.

bloggie/install/langs
Φάκελος γλωσσών προγραμματιδίου. Το php αρχείο με όλα τα απαιτούμενα αρτικόλεξα που χρησιμοποιεί το προγραμματίδιο, και δεν υπάρχουν στην κεντρική γλώσσα της πλατφόρμας.

bloggie/install/langs/en.php
Βασικό αρχείο αγγλικής γλώσσας.

bloggie/install/sql
SQL αρχεία για εγκατάσταση/απεγκατάσταση του προγραμματιδίου.

bloggie/install/sql/install.sql
Αρχείο εγκατάστασης SQL.

bloggie/install/sql/uninstall.sql
Αρχείο απεγκατάστασης SQ.

bloggie/request.php
Αρχείο που δρομολογεί όλες τις κλήσεις προς την κλάση

bloggie/templates
Φάκελος templates του προγραμματιδίου. Εδώ βρίσκονται τα αρχεία html, css και αρχεία εικόνων.

bloggie/templates/base
Βασικός φάκελος Template. Δεν είναι ανάγκη να δημιουργήσουμε ξεχωριστό φάκελο για κάθε template, αλλά σε περίπτωση που θελήσουμε να αναπτύξουμε ένα τεμπλατε, μας υποβοηθά στην γενικευμένη προσαρμογή της όψης.

bloggie/templates/base/css
Τα CSS αρχεία μας.

bloggie/templates/base/images
Τα αρχεία εικόνας μας.

bloggie/templates/base/images/icons
Τα αρχεία εικονιδίων μας.

bloggie/templates/base/main.html
Το βασικό μας αρχείο Template.

Η θέση κάθε αρχείου/φακέλου είναι πολύ σημαντική, ειδικά αν πρόκειται για προγραμματίδιο που θα διανεμηθεί σε πολλούς χρήστες. Επισημαίνουμε ότι, διατηρώντας την δομή των φακέλων ακι τις ονομασίες των σημαντικών αρχείων, βελτιστοποιούμε την πιθανότητα το προγραμματίδιο να εφαρμόζεται και σε επόμενες εκδόσεις της πλατφόρμας..

Για αρχή, θα εγκαταστήσουμε την δοκιμαστική εφαρμογή της πλατφόρμας, πηγαίνοντας στο κέντρο διαχείρισης του Dolphin και επιλέγοντας τα μη εγκατεστημένα προγραμματίδια.Εδώ θα βρούμε το προγραμματίδιο **Bloggie version 1.0.0 by Lamda**. Αφού επιλέξουμε το προγραμματίδιο και πατήσουμε το κουμπί εγκατάστασης, θα είμαστε σε θέση να δούμε την λειτουργία του προγραμματιδίου μέσω της διευθύνσεως:

```
http://www.your-site.com/lamda/m/bloggie/home
```

ή αν το mod rewrite του Εξυπηρετητή Ιστού μας δεν είναι ενεργοποιημένο:

http://www.your-site.com/lamda/modules/?r=bloggie/home

3.2 Αλλαγή κειμένου

Η επόμενη φράση βρίσκεται μέσα στο αρχείο γλώσσας (en.php) του προγραμματιδίου μας. Το συγκεκριμένο αρτικόλεξο μπορεί να αλλάξει από το αρχείο, είτε από το Κέντρο Διαχείρισης από τον σύνδεσμο Διαχείρισης Γλωσσών.Εμείς θα εφαρμόσουμε την πρώτη μέθοδο, μέσω επεξεργασίας του αρχείου **bloggie/install/langs/en.php**.

Έτσι τα περιεχόμενα του αρχείου είναι:

```
$sLangCategory = 'My Bloggie';

$aLangContent = array(
    '_me_blgg' => 'My Bloggie',
    '_me_blgg_text' => 'For a community to be whole and healthy, it
must be based on people\'s love and concern for each other.',
);
```

Το αρχείο γλώσσας είναι ένα κλασσικό PHP αρχείο, που περιέχει τα κλειδιά γλώσσας του συστήματος, και τα αντίστοιχα αρτικόλεξά τους, την μετάφρασή τους δηλαδή στην αναμενόμενη γλώσσα. Όποτε επεξεργαζόμεστε τα αρχεία γλωσσών, θα πρέπει να προσέχουμε ώστε οι χαρακτήρες να είναι σε κωδικοποίηση UTF-8.Τέλος, το τελικό αρχείο μας θα είναι ως εξής:

```
$sLangCategory = 'My Bloggie';

$aLangContent = array(
    '_me_blgg' => 'My Bloggie',
    '_me_blgg_text' => 'Hello World!',
);
```

Οι αλλαγές μας θα είναι εμφανείς μόνο μετά από Recompiling των γλωσσών στην πλατφόρμα μας. Αυτό συμβαίνει επειδή, τα αρτικόλεξα και τα αντίστοιχα κλειδιά συστήματος, δεν ανακαλούνται από τα αρχεία γλωσσών αλλά από ένα cached αρχείο που δημιουργείται κατά το compilation με βάση τα δεδομένα της βάσης δεδομένων και των κατά τόπους γλωσσών των προγραμματιδίων.

3.3 Δυναμικό περιεχόμενο

Για μερικούς χρήστες, μπορεί να θεωρηθεί χρήσιμο το να βλέπουν τον χρόνο χρήσης του προγραμματιδίου μας, καθώς και τον αντίστοιχο χρόνο του εξυπηρετητή μας.

Έτσι, δημιουργούμε ένα νέο κλειδί γλώσσας στο αρχείο μας.:

```
$sLangCategory = 'My Bloggie';

$aLangContent = array(
    '_me_blgg' => 'My Bloggie',
    '_me_blgg_server_time' => 'Server time:',
    '_me_blgg_user_time' => 'User time:',
);
```

Έπειτα, αλλάζουμε το αρχείο του Template ώστε να ανταποκρίνεται στις αλλαγές μας. Το αρχείο **bloggie/templates/base/main.html** μετατρέπεται από:

```
<div style="text-align:center;">
    <bx_text:_me_blgg_text />
</div>
```

σε:

```
<div style="text-align:center;">
    <bx_text:_me_blgg_server_time /> __server_time__
    <br />
    <bx_text:_me_blgg_user_time />
    <script type="text/javascript">
        document.write(new Date());
    </script>
</div>
```

Προσοχή στα σημεία:

- **Το bx_text** είναι μια ειδική οδηγία προς την πλατφόρμα, η οποία εξάγει την μετάφραση ενός κλειδιού γλώσσας του συστήματός μας.
- **Το server_time** είναι μια μεταβλητή της template μας την οποία ακόμη δεν έχουμε ορίσει.
- Για την εμφάνιση του χρόνου χρήσης του χρήστη χρησιμοποιούμε ένα τμήμα κώδικα javascript, επειδή τα javascripts εκτελλούντε στον περιηγητή του χρήστη και έτσι θα εμφανίσουμε τον σωστό χρόνο χρήσης.

Μόλις ανανεώσουμε την σελίδα μας θα δούμε το παρακάτω:

```
Server time: __server_time__
User time: Fri Jan 22 2010 13:46:07 GMT+1100
```

Τώρα το μόνο που μας λείπει είναι ο χρόνος χρήσης του Εξυπηρετητή μας, τον οποίο μπορούμε να εξάγουμε με χρήση php γλώσσας. Τα underscoresεξακολουθούν να εμφανίζονται επειδή δεν έχουμε ορίσει ακόμα την αντίστοιχη μεταβλητή.

Επειδή πρόκειται για δυναμική μεταβλητή και όχι στατικό αρτικόλεξο, θα τροποποιήσουμε το αρχείο **bloggie/classes/MeBlggModule.php** καταλλήλως:

```
function actionHome () {
    $this->_oTemplate->pageStart();
    $aVars = array ();
    echo $this->_oTemplate->parseHtmlByName('main', $aVars);
    $this->_oTemplate->pageCode(_t('_me_blgg'), true);
}
```

Με τις κατάλληλες προσθήκες:

```
function actionHome () {
    $this->_oTemplate->pageStart();
    $aVars = array (
        'server_time' => date('r'),
    );
    echo $this->_oTemplate->parseHtmlByName('main', $aVars);
    $this->_oTemplate->pageCode(_t('_me_blgg'), true);
}
```

Έτσι πλεόν έχουμε το επιθυμητό αποτέλεσμα στο τελικό μας αρχείο.

```
Server time: Fri, 22 Jan 2010 13:56:54 +1100
User time: Fri Jan 22 2010 13:56:55 GMT+1100
```

Ας δούμε όμως κάποια σημεία ενδιαφέροντος εδώ:

- **_oTemplate** - πρόκειται για μια κλάση τύπου template που ορίζεται αυτόματα από το σύστημά μας, στην περίπτωσή μας πρόκειται για instance της κλάσης **MeBlggTemplate**

- **pageStart** –είναι οδηγία προς την πλατφόρμα πως τα δεδομένα που θα εμφανιστούν ακολουθούν

- **pageCode** επεξηγεί πως η εξαγωγή των δεδομένων έχει τελειώσει, οπότε εμφανίζει τα δεδομένα σύμφωνα με τους κανόνες που έχουμε καθορίσει.
    - Η πρώτη παράμετρος είναι το όνομα της σελίδας html που θα χρησιμοποιηθεί, το οποίο λαμβάνουμε από το αντίστοιχο αρχείο γλώσσας (χρησιμοποιούμε την λειτουργία _t)
- Η δεύτερη παράμετρος επικοινωνεί με την πλατφόρμα μας και εγκλωβίζει όλη την έξοδο των δεδομένων σε ένα block

- **parseHtmlByName**- η λειτουργία που εξάγει το αρχείο **main.html** κάνοντας όλες τις αναγκαίες αντικαταστάσεις. Δέχεται την παράμετρο **$aVars** (μεταβλητή όπου αποθηκεύσαμε τον χρόνο του εξυπηρετητή) και δηλώσαμε στο κλειδί **server_time**.

3.4 Προσθήκη σελίδας κέντρου Διαχείρισης.

Για να προσθέσουμε μια λειτουργεία ενεργοποίησης και απενεργοποίησης του προγραμματιδίου μας, εργαζόμαστε με την παρακάτω μέθοδο.

Το κέντρο διαχείρισης, αποτελείται από τα παρακάτω σημεία:

- Δημιουργία σελίδας Διαχείρισης του προγραμματιδίου
- Μεταβλητές εγκατάστασης/απεγκατάστασης του προγραμματιδίου
- Εμφάνιση δεδομένων των νεο-δημιουργημένων σελίδων
- Σύνδεσμος προς την σελίδα διαχείρισης

3.4.1 Σελίδα διαχείρισης Προγραμματιδίου

Προσθέτουμε τα παρακάτω στο αρχείο **bloggie/classes/MeBlggModule.php**ως μέθοδο κλάσης:

```
function actionAdministration () {

    if (!$GLOBALS['logged']['admin']) {
        $this->_oTemplate->displayAccessDenied ();
        return;
    }

    $this->_oTemplate->pageStart();

    echo DesignBoxAdmin (_t('_me_blgg'), 'It works!');

    $this->_oTemplate->pageCodeAdmin (_t('_me_blgg'));
}
```

3.2 Μεταβλητές διαχείρισης

Τροποποιούμε τα αρχεία εγκατάστασης SQL και ρουτίνων SQL:

```
'install' => array(
    'update_languages' => 1,
),
'uninstall' => array (
    'update_languages' => 1,
),
```

to the following:

```
'install' => array(
    'update_languages' => 1,
    'execute_sql' => 1,
),
'uninstall' => array (
    'update_languages' => 1,
    'execute_sql' => 1,
),
```

Αρχεία SQL:

```
SET @iMaxOrder = (SELECT `menu_order` + 1 FROM `sys_options_cats`
ORDER BY `menu_order` DESC LIMIT 1);
INSERT INTO `sys_options_cats` (`name`, `menu_order`) VALUES ('My
Bloggie', @iMaxOrder);
SET @iCategId = (SELECT LAST_INSERT_ID());
INSERT INTO `sys_options` (`Name`, `VALUE`, `kateg`, `desc`, `Type`,
`check`, `err_text`, `order_in_kateg`, `AvailableValues`) VALUES
('me_blgg_date_format', 'Y-m-d H:i', @iCategId, 'Format for server
date/time', 'digit', '', '', '1', ''),
('me_blgg_enable_js_date', 'on', @iCategId, 'Show user time',
'checkbox', '', '', '2', '');
```

uninstall.sql:

```
-- settings
SET @iCategId = (SELECT `ID` FROM `sys_options_cats` WHERE `name` =
'My Bloggie' LIMIT 1);
DELETE FROM `sys_options` WHERE `kateg` = @iCategId;
DELETE FROM `sys_options_cats` WHERE `ID` = @iCategId;
```

Σημαντικότατα εδώ, η δήλωση κατηγορίας στον πίνακα επιλογών του προγραμματιδίου, ώστε η απεγκατάστασή του να είναι εύκολη και διακριτή. Εδώ τροποποιούμε τους πίνακες **sys_options** και **sys_options_cats**

Η λειτουργία **actionHome** τροποποιείται ως εξής:

```
    function actionHome () {
        $this->_oTemplate->pageStart();

        $sDateFormat = getParam ('me_blgg_date_format');
        $isShowUserTime = getParam('me_blgg_enable_js_date') ? true :
false;
        $aVars = array (
            'server_time' => date($sDateFormat),
            'bx_if:show_user_time' => array(
                'condition' => $isShowUserTime,
                'content' => array(),
            ),
        );
        echo $this->_oTemplate->parseHtmlByName('main', $aVars);
        $this->_oTemplate->pageCode(_t('_me_blgg'), true);
    }
```

Τροποποίηση αρχείου template:

```
<div style="text-align:center;">
    <bx_text:_me_blgg_server_time /> __server_time__
    <bx_if:show_user_time>
        <br />
        <bx_text:_me_blgg_user_time />
        <script type="text/javascript">
            document.write(new Date());
        </script>
    </bx_if:show_user_time>
</div>
```

Σημεία ενδιαφέροντος:

- **getParam** – λειτουργία που χρησιμεύει στο να δημιουργήσουμε επιλογή με το συγκεκριμένο όνομα, τύπου checkbox,που θα επιστρέψει την τιμή **on** ως τιμή true προς το προγραμματίδιο.
- **bx_if** ειδική οδηγία του templating system όπου έχουμε ροή δεδομένων χωρίς κατάλληλη δυναμική επεξεργασία.

3.4.3 Φόρμα επιλογών στο Κέντρο Διαχείρισης

Η ρουτίνα **actionAdministration** θα είναι η παρακάτω:

```
    function actionAdministration () {

        if (!$GLOBALS['logged']['admin']) { // check access to the
page
            $this->_oTemplate->displayAccessDenied ();
            return;
        }

        $this->_oTemplate->pageStart(); // all the code below will be
wrapped by the admin design

        $iId = $this->_oDb->getSettingsCategory(); // get our setting
category id
        if(empty($iId)) { // if category is not found display page
not found
            echo MsgBox(_t('_sys_request_page_not_found_cpt'));
            $this->_oTemplate->pageCodeAdmin (_t('_me_blgg'));
            return;
        }

        bx_import('BxDolAdminSettings'); // import class

        $mixedResult = '';
        if(isset($_POST['save']) && isset($_POST['cat'])) { // save
settings
            $oSettings = new BxDolAdminSettings($iId);
            $mixedResult = $oSettings->saveChanges($_POST);
        }

        $oSettings = new BxDolAdminSettings($iId); // get display
form code
        $sResult = $oSettings->getForm();

        if($mixedResult !== true && !empty($mixedResult)) // attach
any resulted messages at the form beginning
            $sResult = $mixedResult . $sResult;

        echo DesignBoxAdmin (_t('_me_blgg'), $sResult); // dsiplay
box

        $this->_oTemplate->pageCodeAdmin (_t('_me_blgg')); // output
is completed, admin page will be displaed here
    }
```

Και έπειτα προσθέτουμε την παρακάτω μέθοδο στο αρχείο **modules/me/bloggie/classes/MeBlggDb.php**:

```
    function getSettingsCategory() {
        return $this->getOne("SELECT `ID` FROM `sys_options_cats`
WHERE `name` = 'My Bloggie' LIMIT 1");
    }
```

Επεξηγήσεις:

- η μέθοδος **getSettingsCategory** τοποθετείται μέσα στο αρχείο **MeBlggDb**.
- Η μέθοδος **getOne** επιστρέφει την τιμή του κλειδιού του πίνακα **ID** μέσω ενός **SELECT** SQL query. Άλλες εξίσου χρήσιμες ρουτίνες υπάρχουν στο αρχείο **inc/classes/BxDolDb.php**.
- **Η _oDb** δηλώνεται αυτόματα στην κλάση **MeBlggDb**
- Η λειτουργία **MsgBox** εσωκλείει σε ένα block ειδοποίησης το κείμενο που θέλουμε.
- Η λειτουργία **bx_import** εισάγει αυτόματα μια κλάση με βάση το όνομά της και ενεργοποιεί κλάσεις με αρχικό πρόσθετο **BxDol**, **BxBase**, **BxTempl**
- Η κλάση **BxDolAdminSettings** χρησιμοποιείται για να εμφανίσει και να αποθηκεύσει επιλογές μας στο Κέντρο Διαχείρισης

3.4.4 Σύνδεσμος προς σελίδα Διαχείρισης του Προγραμματιδίου

Προσθέτουμε τα εξείς στο αρχείο **install.sql**:

```
-- settings
SET @iMaxOrder = (SELECT `menu_order` + 1 FROM `sys_options_cats`
ORDER BY `menu_order` DESC LIMIT 1);
INSERT INTO `sys_options_cats` (`name`, `menu_order`) VALUES ('My
Bloggie', @iMaxOrder);
SET @iCategId = (SELECT LAST_INSERT_ID());
INSERT INTO `sys_options` (`Name`, `VALUE`, `kateg`, `desc`, `Type`,
`check`, `err_text`, `order_in_kateg`, `AvailableValues`) VALUES
('me_blgg_permalinks', 'on', 26, 'Enable friendly permalinks in
Bloggie', 'checkbox', '', '', '0', ''),
('me_blgg_date_format', 'Y-m-d H:i', @iCategId, 'Format for server
date/time', 'digit', '', '', '1', ''),
('me_blgg_enable_js_date', 'on', @iCategId, 'Show user time',
'checkbox', '', '', '2', '');

-- permalinks
INSERT INTO `sys_permalinks` VALUES (NULL, 'modules/?r=bloggie/',
'm/bloggie/', 'me_blgg_permalinks');

-- admin menu
SET @iMax = (SELECT MAX(`order`) FROM `sys_menu_admin` WHERE
`parent_id` = '2');
INSERT IGNORE INTO `sys_menu_admin` (`parent_id`, `name`, `title`,
`url`, `description`, `icon`, `order`) VALUES
(2, 'me_blgg', '_me_blgg',
'{siteUrl}modules/?r=bloggie/administration/', 'My Bloggie by Me',
'modules/me/bloggie/|icon.png', @iMax+1);
```

Και στο **uninstall.sql**:

```
-- settings
SET @iCategId = (SELECT `ID` FROM `sys_options_cats` WHERE `name` =
'My Bloggie' LIMIT 1);
DELETE FROM `sys_options` WHERE `kateg` = @iCategId;
DELETE FROM `sys_options_cats` WHERE `ID` = @iCategId;
DELETE FROM `sys_options` WHERE `Name` = 'me_blgg_permalinks';

-- permalinks
DELETE FROM `sys_permalinks` WHERE `standard` =
'modules/?r=bloggie/';

-- admin menu
DELETE FROM `sys_menu_admin` WHERE `name` = 'me_blgg';
```

Επίσης, τροποποιούμε το αρχείο **modules/me/bloggie/install/config.php**ακριβώς επειδή πρέπει να το επικαιροποιήσουμε με βάση τον πίνακα **sys_permalinks** της βάσης δεδομένων μας:

```
'install' => array(
    'update_languages' => 1,
    'execute_sql' => 1,
    'recompile_permalinks' => 1,
),
'uninstall' => array (
    'update_languages' => 1,
    'execute_sql' => 1,
    'recompile_permalinks' => 1,
),
```

the **recompile_permalinks** instruction was added above.

3.5 Λειτουργία Ιστολογίου

Το προγραμματίδιο Bloggie θα περιέχει τις εξείς σελίδες:

- Σελίδα τελευταίων δημοσιεύσεων Ιστολογίων
- Σελίδα προβολής Ιστολογίου
- Σελίδα Προσθήκης Δημοσιεύματος
- Σελίδα Τροποποίησης Δημοσιεύματος

Το πλέον συμαντικό αυτού του τμήματος κώδικα είναι η χρήση φορμών μέσα από το εγγενές σύστημα της πλατφόρμας

Δομή της βάσης δεδομένων μας

Εδώ θα φιλοξενούνται τα δεδομένα των ιστολογίων μας:

```
CREATE TABLE `me_blgg_posts` (
    `id` INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    `title` VARCHAR( 255 ) NOT NULL ,
    `text` TEXT NOT NULL ,
    `author_id` INT UNSIGNED NOT NULL ,
    `added` INT UNSIGNED NOT NULL ,
    INDEX ( `author_id` )
);
```

Τα παραπάνω θα βρίσκονται στο αρχείο **install.sql**. Επίσης θα προσθέσουμε μια οδηγία τύπου "drop" στην αρχή του αρχείου **uninstall.sql**:

```
DROP TABLE `me_blgg_posts`;
```

Πρόκειται για βασικές λειτουργίες σύνταξης SQL, οπότε δεν θα επεκταθούμε περεταίρω σε αυτές. Γενικά όμως,κάθε δημοσίευμα του προγραμματιδίου **Bloggie** θα περιέχει τίτλο, και κείμενο της εν λόγω δημοσίευσης:

- **id** – μοναδικό κλειδί στον πίνακα της βάσης δεδομένων
- **author_id** –το μοναδικό κλειδί του συγγραφέα
- **added** – ημερομηνία και ώρα της δημοσίευσης

3.5.1 Σελίδα Προσθήκης δημοσιεύματος

Τοποθετούμε την παρακάτω μέθοδο στην κλάση **MeBlggModule**:

```
    function actionAdd () {

        if (!$GLOBALS['logged']['member'] &&
!$GLOBALS['logged']['admin']) { // check access to the page
            $this->_oTemplate->displayAccessDenied ();
            return;
        }

        $this->_oTemplate->pageStart(); // all the code below will be
wrapped by the user design

        bx_import ('BxTemplFormView'); // import forms class

        $oForm = new BxTemplFormView ($this->aForm); // create foprms
class

        $oForm->initChecker(); // init form checker

        if ($oForm->isSubmittedAndValid ()) { // if form is submitted
and not form errors were found, save form data

            $aValsAdd = array ( // add additional fields
                'added' => time(),
                'author_id' => $_COOKIE['memberID'],
            );
            $iEntryId = $oForm->insert ($aValsAdd); // insert data to
database

            if ($iEntryId) { // if post was successfully added
                $sRedirectUrl = BX_DOL_URL_ROOT . $this->_oConfig-
>getBaseUri() . 'view/' . $iEntryId;
                header ('Location:' . $sRedirectUrl); // redirect to
created post view page
                exit;
            } else {
                MsgBox(_t('_Error Occured')); // if error occured
display erroro message
            }

        } else {

            echo $oForm->getCode (); // display form, if the form is
not submiyyed or data is invalid

        }

        $this->_oTemplate->pageCode(_t('_me_blgg_page_title_add'),
true); // output is completed, display all output above data wrapped
by user design
    }
```

Τέλος τροποποιούμε τον class constructor της κλάσης **MeBlggModule**:

```
function MeBlggModule(&$aModule) {
    parent::BxDolModule($aModule);

    $this->aForm = array(

        'form_attrs' => array(
            'name'     => 'form_bloggie',
            'action'   => '',
            'method'   => 'post',
        ),

        'params' => array (
            'db' => array(
                'table' => 'me_blgg_posts',
                'key' => 'id',
                'submit_name' => 'submit_form',
            ),
        ),

        'inputs' => array(

            'title' => array(
                'type' => 'text',
                'name' => 'title',
                'caption' => _t('_me_blgg_form_caption_title'),
                'required' => true,
                'checker' => array (
                    'func' => 'length',
                    'params' => array(3,255),
                    'error' => _t ('_me_blgg_form_err_title'),
                ),
                'db' => array (
                    'pass' => 'Xss',
                ),
            ),

            'text' => array(
                'type' => 'textarea',
                'name' => 'text',
                'caption' => _t('_me_blgg_form_caption_text'),
                'required' => true,
                'html' => 2,
                'checker' => array (
                    'func' => 'length',
                    'params' => array(20,64000),
                    'error' => _t ('_me_blgg_form_err_text'),
                ),
                'db' => array (
                    'pass' => 'XssHtml',
                ),
            ),

            'submit' => array (
                'type' => 'submit',
                'name' => 'submit_form',
                'value' => _t('_Submit'),
```

```
                    'colspan' => true,
                ),

            ),
        );
    }
```

Η αλφαριθμιτική τιμή του constructor είναι η περιγραφή της φόρμας μας. Την τοποθετούμε εκεί για λόγους τάξης και επαναχρησιμοποίησης.

Μπορούμε να περιγράψουμε το αλφαριθμιτικό με τον παρακάτω τρόπο:

form_attrs – πρόκειται για HTML μεταβλητές που προστίθενται στην ετικέτα **<form>**.

params – πρόκειται για ειδικές παραμέτρους της λειτουργίας. Οι πιο μσημαντικές είναι ο υποκατάλογος **db** που περιγράφει τον πίνακα όπου αποθηκεύονται τα δημοσιεύματα.

- **Το table** είναι το όνομα του πίνακα
- **Το key** είναι το πρωτεύων κλειδί του πίνακα
- **Το submit_name** είναι το όνομα του κουμπιού που θα έχει η φόρμα.

Το αλφαριθμητικό **inputs** περιγράφει κάθε πεδίο του πίνακα όπου αποθηκεύονται τα δημοσιεύματα:

- **type** – είδος δεδομένωνόπως **text**, **textarea κτλ**
- **name** - Το όνομα του πεδίου της φόρμας που πρέπει να είναι ίδιο με το όνομα του πεδίου της βάσης δεδομένων
- **caption** – Η επικεφαλίδα της φόρμας μας
- **required** – Ελεγκτής για το αν το πεδίο πρόκειται για υποχρεωτικό
- **checker** Δηλώνει τον έλεγχο που γίνεται στο αντιστοιχισμένο πεδίο.Εδώ γίνεται ο πραγματικός έλεγχος του πεδίου.
    - **Func** – η λειτουργία ελέγχου όπως **length**, **preg**, **avail**, κτλ.
    - **params** – οι παραμετροι της λειτουργίας ελέγχου.
    - **error** – το μύνημα σφάλματος αν τα εισαγόμενα δεδομένα δεν περάσουν τον έλεγχο επιτυχώς.
- Το db περιγράφει το πεδίο της βάσης όπου αποθηκεύεται το εκάστοτε δεδομένο:
    - **pass** – πρόκειται για λειτουργία της πλατφόρμας όπου έχουμε έλεγχο του τύπου των δεδομένων του πεδίουόπως π.χ. **Xss**, **XssHtml?**, **Int**, **Float**, **Date**, κτλ.

3.5.2 Προβολή Δημοσιεύματος

Προσθέτουμε τον παρακάτω κώδικα στην κλάση **MeBlggModule**:

```
    function actionView ($iEntryId) {

        $aEntry = $this->_oDb->getEntryById ((int)$iEntryId);
        if (!$aEntry) { // check if entry exists
            $this->_oTemplate->displayPageNotFound ();
            return;
        }

        $this->_oTemplate->pageStart(); // all the code below will be
wrapped by the user design

        $aVars = array (
            'title' => $aEntry['title'],
            'text' => $aEntry['text'],
            'author' => getNickName($aEntry['author_id']),
            'added' => defineTimeInterval($aEntry['added']),
        );
        echo $this->_oTemplate->parseHtmlByName('view', $aVars); //
display post

        $this->_oTemplate->pageCode($aEntry['title'], true); //
output is completed, display all output above data wrapped by user
design
    }
```

Εδώ η λειτουργία μας δέχεται για πρώτη φορά παραμέτρους:

- **$iEntryId** – παράμετρος λειτουργίας που περνάει μόλις καλέσουμε την διεύθυνση: **http://www.your-site.com/m/bloggie/view/15**,σε αυτή την περίπτωση ίση με «15».
- **getNickName** είναι λειτουργία συστήματος που θα μας επιστρέψει το user id.
- **defineTimeInterval** – λειτουργία που δημιουργεί timestamp.
- **displayPageNotFound** εμφανίζει μήνυμα τύπου 404μαζί με τους κατάλληλους headers.
- **getEntryById** είναι η λειτουργία που συγχρονίζει την βάση δεδομένων μας.

Προσθέτουμε την παρακάτω μέθοδο στην κλάση **MeBlggDb**:

```
    function getEntryById ($iEntryId) {
        return $this->getRow("SELECT * FROM `" . $this->_sPrefix .
"posts` WHERE `id` = '$iEntryId'");
    }
```

Τέλος ορίζουμε τα δεδομένα του αρχείου **bloggie/templates/base/view.html**:

```
<div>
    <h1>__title__</h1>
    <p>__text__</p>
    <p style="text-align:right; font-
style:italic;"><bx_text:_me_blgg_posted_by /> __author__
(__added__)</p>
</div>
```

3.6 Αρχική Σελίδα

Τροποποιούμε την λειτουργία **actionHome** καταλλήλως:

```
    function actionHome () {
        $this->_oTemplate->pageStart(); // all the code below will be
wrapped by the user design

        $aPosts = $this->_oDb->getAllPosts
(getParam('me_blgg_max_posts_to_show')); // get all posts from
database
        foreach ($aPosts as $sKey => $aRow) { // add human readable
values to the resulted array
            $aPosts[$sKey]['url'] = BX_DOL_URL_ROOT . $this-
>_oConfig->getBaseUri() . 'view/' . $aRow['id'];
            $aPosts[$sKey]['added'] =
defineTimeInterval($aRow['added']);
            $aPosts[$sKey]['author'] =
getNickName($aRow['author_id']);
        }
        $aVars = array ( // define template variables
            'bx_repeat:posts' => $aPosts,
        );
        echo $this->_oTemplate->parseHtmlByName('main', $aVars); //
output posts list

        $this->_oTemplate->pageCode(_t('_me_blgg'), true); // output
is completed, display all output above data wrapped by user design
    }
```

Και αντιστοίχως το αρχείο template **bloggie/templates/base/main.html**:

```
    <bx_repeat:posts>
        <h2>
            <a href="__url__">__title__</a>
        </h2>
        <p>__text__</p>
        <p style="text-align:right; font-
style:italic;"><bx_text:_me_blgg_posted_by /> __author__
(__added__)</p>
        <hr />
    </bx_repeat:posts>
```

Η νέα μέθοδός μας:

```
    function getAllPosts ($iLimit) {
        return $this->getAll("SELECT *, LEFT(`text`, 256) AS `text`
FROM `" . $this->_sPrefix . "posts` ORDER BY `added` DESC LIMIT
$iLimit");
    }
```

Χρησιμοποιούμε την οδηγία **bx_repeat:posts** για να έχουμε επανάληψη της εμφάνισης των δημοσιευμάτων. Ιδιαίτερη προσοχή στην μέθοδο κλάσης **getBaseUri** που επιστρέφει το **modules/?r=bloggie/** or **m/bloggie/** αναλόγως με το αν τα permalinks και το **mod_rewrite** είναι ενεργοποιημένα ή όχι.

3.7 Επεξεργασία δημοσιεύσεων Ιστολογίου

Προσθέτουμε τα παρακάτω στστην κλάση **MeBlggModule**:

```
    function actionEdit ($iEntryId) {

        $aEntry = $this->_oDb->getEntryById ((int)$iEntryId);
        if (!$aEntry) { // check if entry exists
            $this->_oTemplate->displayPageNotFound ();
            return;
        }

        if ((!$GLOBALS['logged']['member'] &&
!$GLOBALS['logged']['admin']) || $aEntry['author_id'] !=
$_COOKIE['memberID']) { // check access to the page
            $this->_oTemplate->displayAccessDenied ();
            return;
        }

        $this->_oTemplate->pageStart(); // all the code below will be
wrapped by the user design

        bx_import ('BxTemplFormView'); // import forms class

        $oForm = new BxTemplFormView ($this->aForm); // create forms
class

        $oForm->initChecker($aEntry); // init form checker

        if ($oForm->isSubmittedAndValid ()) { // if form is submitted
and there is no form errors were found, save form data

            $iRes = $oForm->update ($iEntryId); // update data in
database

            if ($iRes) { // if post was successfully added
                $sRedirectUrl = BX_DOL_URL_ROOT . $this->_oConfig-
>getBaseUri() . 'view/' . $iEntryId;
                header ('Location:' . $sRedirectUrl); // redirect to
updated post view page
                exit;
            } else {
                MsgBox(_t('_Error Occured')); // if error occured
display error message
            }

        } else {

            echo $oForm->getCode (); // display form, if the form is
not submitted or data is invalid

        }

        $this->_oTemplate->pageCode(_t('_me_blgg_page_title_edit'),
true); // output is completed, display all output above data wrapped
by user design
    }
```

Η λειτουργία είναι ίδια με την actionAdd method – με τις παρακάτω διαφοροποιήσεις:

- Έχουμε επιπρόσθετο έλεγχο της ασφάλειας ώστε να επιτρέπουμε μόνο στον χρήστη που έκανε την δημοσίευση, να έχει δικαίωμα τροποποίησης της
- Ο **initChecker** δέχεται την μεταβλητή **$aEntry** ως παράμετρο για να γεμίσει τα πεδία με δεδομένα της δημοσίευσης
- Αντί για **save** καλέσαμε την λειτουργία της φόρμας **update**με το id του χρήστη.

Ο σύνδεσμος δημοσίευσης τοποθετείται κοντά στον τίτλο του δημοσιεύματος, για συνοχικότητα της όψης, και για να το επιτύχουμε, τροποποιούμε την μέθοδο **actionView** όπως περιγράφεται ακολούθως:

```
    function actionView ($iEntryId) {

        $aEntry = $this->_oDb->getEntryById ((int)$iEntryId);

        if (!$aEntry) { // check if entry exists
            $this->_oTemplate->displayPageNotFound ();
            return;
        }

        $this->_oTemplate->pageStart(); // all the code below will be
wrapped by the user design

        $aVars = array (
            'title' => $aEntry['title'],
            'text' => $aEntry['text'],
            'author' => getNickName($aEntry['author_id']),
            'added' => defineTimeInterval($aEntry['added']),
            'bx_if:edit' => array(
                'condition' => ($GLOBALS['logged']['member'] ||
$GLOBALS['logged']['admin']) && $aEntry['author_id'] ==
$_COOKIE['memberID'],
                'content' => array(
                    'edit_url' => BX_DOL_URL_ROOT . $this->_oConfig-
>getBaseUri() . 'edit/' . $aEntry['id'],
                ),
            ),
        );
        echo $this->_oTemplate->parseHtmlByName('view', $aVars); //
display post

        $this->_oTemplate->pageCode($aEntry['title'], true); //
output is completed, display all output above data wrapped by user
design
    }
```

Ο παρακάτω κώδικας εφαρμόζεται στο αρχείο **bloggie/templates/base/view.html**:

```
<div>
    <h1>
        __title__
```

```
        <bx_if:edit>
            <a href="__edit_url__"><bx_text:_me_blgg_edit /></a>
        </bx_if:edit>
    </h1>
    <p>__text__</p>
    <p style="text-align:right; font-
style:italic;"><bx_text:_me_blgg_posted_by /> __author__
(__added__)</p>
</div>
```

3.x Τελικά αρχεία

Το Αρχείο request.php

```
<?
require_once(BX_DIRECTORY_PATH_INC . 'profiles.inc.php');

check_logged();

bx_import('BxDolRequest');
BxDolRequest::processAsAction($GLOBALS['aModule'], $GLOBALS['aRequest']);

?>
```

Οι κλάσεις μας και τα αντίστοιχα αρχεία στο προγραμματίδιο:

1)MeBlggConfig.php

```
<?
bx_import('BxDolConfig');

class MeBlggConfig extends BxDolConfig {

        function MeBlggConfig($aModule) {
           parent::BxDolConfig($aModule);
        }
}

?>
```

2)MeBlggDB

```
<?

bx_import('BxDolModuleDb');

class MeBlggDb extends BxDolModuleDb {

        function MeBlggDb(&$oConfig) {
                parent::BxDolModuleDb();
      $this->_sPrefix = $oConfig->getDbPrefix();
   }

   function getSettingsCategory() {
      return $this->getOne("SELECT `ID` FROM `sys_options_cats` WHERE `name`
= 'My Bloggie' LIMIT 1");
   }

   function getEntryById ($iEntryId) {
      return $this->getRow("SELECT * FROM `" . $this->_sPrefix . "posts` WHERE
`id` = '$iEntryId'");
   }

   function getAllPosts ($iLimit) {
      return $this->getAll("SELECT *, LEFT(`text`, 256) AS `text` FROM `" . $this-
>_sPrefix . "posts` ORDER BY `added` DESC LIMIT $iLimit");
   }
}

?>
```

3)MeBlggTemplate.php

```
<?
bx_import ('BxDolTwigTemplate');

class MeBlggTemplate extends BxDolTwigTemplate {
```

```
        function MeBlggTemplate(&$oConfig, &$oDb) {
            parent::BxDolTwigTemplate($oConfig, $oDb);
    }
}

?>
```

4)MeBlggModule.php

```
<?php

bx_import('BxDolModule');

class MeBlggModule extends BxDolModule {

    function MeBlggModule(&$aModule) {
        parent::BxDolModule($aModule);

        $this->aForm = array(

            'form_attrs' => array(
                'name'     => 'form_bloggie',
                'action'   => '',
                'method'   => 'post',
            ),

            'params' => array (
                'db' => array(
                    'table' => 'me_blgg_posts',
                    'key' => 'id',
                    'submit_name' => 'submit_form',
                ),
            ),

            'inputs' => array(

                'title' => array(
                    'type' => 'text',
                    'name' => 'title',
                    'caption' => _t('_me_blgg_form_caption_title'),
                    'required' => true,
                    'checker' => array (
                        'func' => 'length',
                        'params' => array(3,255),
                        'error' => _t ('_me_blgg_form_err_title'),
                    ),
                    'db' => array (
                        'pass' => 'Xss',
                    ),
                ),

                'text' => array(
                    'type' => 'textarea',
                    'name' => 'text',
                    'caption' => _t('_me_blgg_form_caption_text'),
                    'required' => true,
                    'html' => 2,
```

```
                'checker' => array (
                    'func' => 'length',
                    'params' => array(20,64000),
                    'error' => _t ('_me_blgg_form_err_text'),
                ),
                'db' => array (
                    'pass' => 'XssHtml',
                ),
            ),

            'submit' => array (
                'type' => 'submit',
                'name' => 'submit_form',
                'value' => _t('_Submit'),
                'colspan' => true,
            ),

        ),
    );
  }

  function actionHome () {
    $this->_oTemplate->pageStart(); // all the code below will be wrapped by the
user design

    $aPosts = $this->_oDb->getAllPosts
(getParam('me_blgg_max_posts_to_show')); // get all posts from database
    foreach ($aPosts as $sKey => $aRow) { // add human readable values to the
resulted array
        $aPosts[$sKey]['url'] = BX_DOL_URL_ROOT . $this->_oConfig-
>getBaseUri() . 'view/' . $aRow['id'];
        $aPosts[$sKey]['added'] = defineTimeInterval($aRow['added']);
        $aPosts[$sKey]['author'] = getNickName($aRow['author_id']);
    }
    $aVars = array ( // define template variables
        'bx_repeat:posts' => $aPosts,
    );
    echo $this->_oTemplate->parseHtmlByName('main', $aVars); // output posts list

    $this->_oTemplate->pageCode(_t('_me_blgg'), true); // output is completed,
display all output above data wrapped by user design
  }

  function actionView ($iEntryId) {

    $aEntry = $this->_oDb->getEntryById ((int)$iEntryId);
    if (!$aEntry) { // check if entry exists
        $this->_oTemplate->displayPageNotFound ();
        return;
    }
```

```
        $this->_oTemplate->pageStart(); // all the code below will be wrapped by the
user design

        $aVars = array (
            'title' => $aEntry['title'],
            'text' => $aEntry['text'],
            'author' => getNickName($aEntry['author_id']),
            'added' => defineTimeInterval($aEntry['added']),
            'bx_if:edit' => array(
                'condition' => ($GLOBALS['logged']['member'] ||
$GLOBALS['logged']['admin']) && $aEntry['author_id'] ==
$_COOKIE['memberID'],
                'content' => array(
                    'edit_url' => BX_DOL_URL_ROOT . $this->_oConfig->getBaseUri() .
'edit/' . $aEntry['id'],
                ),
            ),
        );
        echo $this->_oTemplate->parseHtmlByName('view', $aVars); // display post

        $this->_oTemplate->pageCode($aEntry['title'], true); // output is completed,
display all output above data wrapped by user design
    }

    function actionAdd () {

        if (!$GLOBALS['logged']['member'] && !$GLOBALS['logged']['admin']) { //
check access to the page
            $this->_oTemplate->displayAccessDenied ();
            return;
        }

        $this->_oTemplate->pageStart(); // all the code below will be wrapped by the
user design

        bx_import ('BxTemplFormView'); // import forms class

        $oForm = new BxTemplFormView ($this->aForm); // create foprms class

        $oForm->initChecker(); // init form checker

        if ($oForm->isSubmittedAndValid ()) { // if form is submitted and not form
errors were found, save form data

            $aValsAdd = array ( // add additional fields
                'added' => time(),
                'author_id' => $_COOKIE['memberID'],
            );
            $iEntryId = $oForm->insert ($aValsAdd); // insert data to database
```

```
            if ($iEntryId) { // if post was successfully added
                $sRedirectUrl = BX_DOL_URL_ROOT . $this->_oConfig->getBaseUri() .
'view/' . $iEntryId;
                header ('Location:' . $sRedirectUrl); // redirect to created post view page
                exit;
            } else {
                MsgBox(_t('_Error Occured')); // if error occured display erroro message
            }

        } else {

            echo $oForm->getCode (); // display form, if the form is not submitted or data
is invalid

        }

        $this->_oTemplate->pageCode(_t('_me_blgg_page_title_add'), true); // output is
completed, display all output above data wrapped by user design
    }

    function actionEdit ($iEntryId) {

        $aEntry = $this->_oDb->getEntryById ((int)$iEntryId);
        if (!$aEntry) { // check if entry exists
            $this->_oTemplate->displayPageNotFound ();
            return;
        }

        if ((!$GLOBALS['logged']['member'] && !$GLOBALS['logged']['admin']) ||
$aEntry['author_id'] != $_COOKIE['memberID']) { // check access to the page
            $this->_oTemplate->displayAccessDenied ();
            return;
        }

        $this->_oTemplate->pageStart(); // all the code below will be wrapped by the
user design

        bx_import ('BxTemplFormView'); // import forms class

        $oForm = new BxTemplFormView ($this->aForm); // create forms class

        $oForm->initChecker($aEntry); // init form checker

        if ($oForm->isSubmittedAndValid ()) { // if form is submitted and not form
errors were found, save form data

            $iRes = $oForm->update ($iEntryId); // insert data to database

            if ($iRes) { // if post was successfully added
```

```
            $sRedirectUrl = BX_DOL_URL_ROOT . $this->_oConfig->getBaseUri() .
'view/' . $iEntryId;
            header ('Location:' . $sRedirectUrl); // redirect to updated post view page
            exit;
        } else {
            MsgBox(_t('_Error Occured')); // if error occured display erroro message
        }

    } else {

        echo $oForm->getCode (); // display form, if the form is not submitted or data
is invalid

    }

    $this->_oTemplate->pageCode(_t('_me_blgg_page_title_edit'), true); // output is
completed, display all output above data wrapped by user design
  }

  function actionAdministration () {

    if (!$GLOBALS['logged']['admin']) { // check access to the page
        $this->_oTemplate->displayAccessDenied ();
        return;
    }

    $this->_oTemplate->pageStart(); // all the code below will be wrapped by the
admin design

            $iId = $this->_oDb->getSettingsCategory(); // get our setting category id
            if(empty($iId)) { // if category is not found display page not found
        echo MsgBox(_t('_sys_request_page_not_found_cpt'));
        $this->_oTemplate->pageCodeAdmin (_t('_me_blgg'));
        return;
    }

    bx_import('BxDolAdminSettings'); // import class

    $mixedResult = '';
    if(isset($_POST['save']) && isset($_POST['cat'])) { // save settings
                $oSettings = new BxDolAdminSettings($iId);
        $mixedResult = $oSettings->saveChanges($_POST);
    }

    $oSettings = new BxDolAdminSettings($iId); // get display form code
    $sResult = $oSettings->getForm();

    if($mixedResult !== true && !empty($mixedResult)) // attach any resulted
messages at the form beginning
        $sResult = $mixedResult . $sResult;
```

```
        echo DesignBoxAdmin (_t('_me_blgg'), $sResult); // dsiplay box

        $this->_oTemplate->pageCodeAdmin (_t('_me_blgg')); // output is completed,
admin page will be displaed here
    }
}
?>
```

Το αρχείο των αρτικόλεξων που θα προστεθούν στην βάση δεδομένων μας:

```
<?

$sLangCategory = 'My Bloggie';

$aLangContent = array(

    '_me_blgg' => 'My Bloggie',

    '_me_blgg_posted_by' => 'Posted by:',
    '_me_blgg_edit' => 'Edit',

    '_me_blgg_page_title_add' => 'Add Bloggie Post',
    '_me_blgg_page_title_edit' => 'Edit Bloggie Post',

    '_me_blgg_form_caption_title' => 'Title',
    '_me_blgg_form_err_title' => 'Title length must be from 3 to 255 characters long',
    '_me_blgg_form_caption_text' => 'Text',
    '_me_blgg_form_err_text' => 'Text must be from 20 to 64000 characters long',

);

?>
```

Το αρχείο διαμόρφωσης της εγκατάστασης:

```
<?
$aConfig = array(
        'title' => 'Bloggie',

   'version' => '1.0.0',
        'vendor' => 'Lamda',
        'update_url' => '',


        'compatible_with' => array(
   ),

        'home_dir' => 'me/bloggie/', // folder where module files are located, it describes path from /modules/ folder.
        'home_uri' => 'bloggie', // module URI, so the module will be accessable via the following urls: m/bloggie/ or modules/?r=bloggie/

        'db_prefix' => 'me_blgg_',
   'class_prefix' => 'MeBlgg',

        'install' => array(
      'update_languages' => 1, // add languages
      'execute_sql' => 1, // process install.sql file
      'recompile_permalinks' => 1, // recompile permalinks cache
   ),

        'uninstall' => array (
      'update_languages' => 1, // remove added languages
      'execute_sql' => 1, // process uninstall.sql file
      'recompile_permalinks' => 1, // recompile permalinks cache
   ),


        'language_category' => 'My Bloggie',

        /**
         * Permissions Section, list all permissions here which need to be changed before install and after uninstall, see examples in other BoonEx modules
         */
        'install_permissions' => array(),
   'uninstall_permissions' => array(),


        'install_info' => array(
                'introduction' => '',
                'conclusion' => ''
        ),
        'uninstall_info' => array(
```

```
                'introduction' => '',
                'conclusion' => ''
        )
);

?>
```

Το αρχείο installer.php

```
<?php

bx_import("BxDolInstaller");

class MeBlggInstaller extends BxDolInstaller {
    function MeBlggInstaller($aConfig) {
        parent::BxDolInstaller($aConfig);
    }
}

?>
```

Αρχείο εγκατάστασης SQL:

```
CREATE TABLE `me_blgg_posts` (
   `id` INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY ,
   `title` VARCHAR( 255 ) NOT NULL ,
   `text` TEXT NOT NULL ,
   `author_id` INT UNSIGNED NOT NULL ,
   `added` INT UNSIGNED NOT NULL ,
   INDEX ( `author_id` )
);

-- settings
SET @iMaxOrder = (SELECT `menu_order` + 1 FROM `sys_options_cats` ORDER
BY `menu_order` DESC LIMIT 1);
INSERT INTO `sys_options_cats` (`name`, `menu_order`) VALUES ('My Bloggie',
@iMaxOrder);
SET @iCategId = (SELECT LAST_INSERT_ID());
INSERT INTO `sys_options` (`Name`, `VALUE`, `kateg`, `desc`, `Type`, `check`,
`err_text`, `order_in_kateg`, `AvailableValues`) VALUES
('me_blgg_permalinks', 'on', 26, 'Enable friendly permalinks in Bloggie', 'checkbox', '',
'', '0', ''),
('me_blgg_max_posts_to_show', '10', @iCategId, 'Max number of posts to show',
'digit', '', '', '1', '');

-- permalinks
INSERT INTO `sys_permalinks` VALUES (NULL, 'modules/?r=bloggie/',
'm/bloggie/', 'me_blgg_permalinks');

-- admin menu
SET @iMax = (SELECT MAX(`order`) FROM `sys_menu_admin` WHERE
`parent_id` = '2');
INSERT IGNORE INTO `sys_menu_admin` (`parent_id`, `name`, `title`, `url`,
`description`, `icon`, `order`) VALUES
(2, 'me_blgg', '_me_blgg', '{siteUrl}modules/?r=bloggie/administration/', 'My Bloggie
by Me', 'modules/me/bloggie/|icon.png', @iMax+1);
```

Το αντίστοιχο αρχείο απεγκατάστασης SQL:

```
DROP TABLE `me_blgg_posts`;

-- settings
SET @iCategId = (SELECT `ID` FROM `sys_options_cats` WHERE `name` = 'My
Bloggie' LIMIT 1);
DELETE FROM `sys_options` WHERE `kateg` = @iCategId;
DELETE FROM `sys_options_cats` WHERE `ID` = @iCategId;
DELETE FROM `sys_options` WHERE `Name` = 'me_blgg_permalinks';

-- permalinks
DELETE FROM `sys_permalinks` WHERE `standard` = 'modules/?r=bloggie/';

-- admin menu
DELETE FROM `sys_menu_admin` WHERE `name` = 'me_blgg';
```

Βασικό αρχείο Templating (main.html):

```
<bx_repeat:posts>
   <h2>
      <a href="__url__">__title__</a>
   </h2>
   <p>__text__</p>
   <p style="text-align:right; font-style:italic;"><bx_text:_me_blgg_posted_by />
__author__ (__added__)</p>
   <hr />
</bx_repeat:posts>
```

Διαχειριστής Views μέσω του Templating System (view.html)

```
<div>
   <h1>
      __title__
      <bx_if:edit>
         <a href="__edit_url__"><bx_text:_me_blgg_edit /></a>
      </bx_if:edit>
   </h1>
   <p>__text__</p>
   <p style="text-align:right; font-style:italic;"><bx_text:_me_blgg_posted_by />
__author__ (__added__)</p>
</div>
```

ΕΥΧΑΡΙΣΤΙΕΣ

Ευχαριστώ θερμά για το αμέριστο ενδιαφέρον και τον πολύτιμο χρόνο που μου αφιερώθηκε από τον επιβλέποντα καθηγητή μου, κ. Γεώργιος Παπαδουράκη. Επίσης, ευχαριστώ τους καθηγητές μου ,που αποτελούν σταθμούς στην φοιτητική μου πορεία και στην κατεύθυνση των γνώσεών μου. Τέλος, ευχαριστώ την οικογένεια και τους φίλους συμφοιτητές μου για την συμπαράσταση και την βοήθειά τους.

ACKNOWLEDGEMENTS

Special thanks for the significant interest and the priceless time that was offered to me, are owned to my supervisor prof. Mr. George Papadourakis. I also thank my professors, which led my student course and the direction of my knowledge. Finally, I thank my family and my fellow students for their support and assistance.

ΠΗΓΕΣ

- Μαριάς Κώστας. “Ιατρική Απεικόνιση – Υπέρηχοι, Υπερηχοτομογραφία”. Η/Υ 571, Τμήμα Επιστήμης Υπολογιστών Πανεπιστημίου Κρήτης, www.csd.uoc.gr/~hy571/LECTURES/5_US.pdf .
- Καρπούζης Κώστας. “Ηλεκτρονικές Υπηρεσίες Υγείας”. Τμήμα Διδακτικής της τεχνολογίας και Ψηφιακών συστημάτων, dtps.unipi.gr/files/notes/2004-2005/ ...ygeias/ehealth03.ppt .
- Η διεθνής, ελεύθερου περιεχομένου και εξελισσόμενη δια συνεργασίας εγκυκλοπαίδεια Βικιπαίδεια. http://en.wikipedia.org/wiki/Segmentation_%28image_processing%29 .
- Τσαπατσούλης Νικόλας. “Αναγνώριση Προτύπων και Ανάλυση Εικόνας”. Τμήμα Επιστήμης και Τεχνολογίας Τηλεπικοινωνιών Πανεπιστημίου Πελοποννήσου 2007.
- Καβαλλιεράτου Εργίνα, Αργυρόπουλος Γιάννης.”Computer Vision - Review on Color Segmentation (Αναφορά στην Χρωματική Κατάτμηση)”. Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων Πανεπιστημίου Αιγαίου 2007.
- Andrew W. Moore. “K-means and Hierarchical Clustering - Tutorial Slides” Robotics and Computer Science , School of Computer Science, Carnegie Mellon University.
- J. B. MacQueen (1967): "Some Methods for classification and Analysis of Multivariate Observations, *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability"*, Berkeley, University of California Press, 1:281-297
- Steve Eddins. **“**The Watershed Transform - Strategies for Image Segmentation”. 1994-2009 The MathWorks Inc.
- Papadourakis George ,“Introduction To Neural Networks”.Πλατφόρμα Ασύγχρονης Τηλεκπαίδευσης GUNet e-Class, Τμήμα Εφαρμοσμένης πληροφορικής και πολυμέσων, http://eclass.epp.teiher.gr/eclass/claroline/document/document.php .
- Jianbo Shi. “Graph Based Image Segmentation” .University of Pennsylvania.

- Η ηλεκτρονική κοινωνία πληροφοριών του Matlab Central μέσω της επίσημης ιστοσελίδας ανταλλαγής ανοιχτού κώδικα Matlab και Simulink. http://www.mathworks.com/matlabcentral/fileexchange/?term=tag%3A%22segmentation%22 .

ΒΙΒΛΙΟΓΡΑΦΙΑ

ΧΡΗΣΙΜΕΣ ΙΣΤΟΣΕΛΙΔΕΣ

- http://en.wikipedia.org .Η διεθνής, ελεύθερου περιεχομένου και εξελισσόμενη δια συνεργασίας εγκυκλοπαίδεια Βικιπαίδεια.
- http://isl.epp.teicrete.gr .Το ερευνητικό Εργαστήριο Ευφυών Συστημάτων του τμήματος Εφαρμοσμένης Πληροφορικής και Πολυμέσων του ΤΕΙ Κρήτης.
- http://www.ceid.upatras.gr .Το Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστημίου Πατρών.
- http://www.ece.ntua.gr . Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών Εθνικού Μετσόβιου Πολυτεχνείου.
- http://www.teiath.gr/stef/tio .Το Τμήμα Τεχνολογίας Ιατρικών Οργάνων του Τ.Ε.Ι. Αθήνας.
- http://www.gsrt.gr .Η Γενική Γραμματεία Έρευνας και Τεχνολογίας, Υπουργείο Ανάπτυξης, Ελληνική Δημοκρατία.