



## Εισαγωγή

Το photosharing είναι μία πλήρης online εφαρμογή διαμοίρασης φωτογραφιών ανάμεσα σε χρήστες και επισκέπτες. Η αναπτυξή του έγινε με βάση την αρχιτεκτονική 3 επιπέδων με αντικειμενοστραφή PHP, ενώ η παρουσίαση πραγματοποιείται με το Smarty Template Engine.

## 3 Layer Architecture

Η 3 layer architecture βασίζεται στο Data Layer, στο Logic Layer, και στο Presentation Layer.

Το Data Layer είναι υπεύθυνο για οποιαδήποτε επικοινωνία με την Βάση δεδομένων, ενώ καλύπτει και κάποιες ακόμα βασικές επεξεργασίες δεδομένων, πριν αυτά σταλθούν στο Logic Layer. Συνήθως το Data Layer επιστρέφει αντικείμενα ή πίνακες αντικειμένων του ζητούμενου τύπου δεδομένων.

Το Logic Layer είναι το layer που γίνεται όλη η επεξεργασία πριν γίνει η προβολή στο χρήστη. Εδώ έρχονται όλα τα ζητούμενα αντικείμενα, επεξεργάζονται και 'δημιουργούν' προβολές για τον χρήστη.

Το Presentation Layer είναι αυτό που έχει όλα τα templates τα οποία παράγονται ανάλογα κάθε φορά με τις απαιτήσεις και τα αιτήματα του χρήστη της εφαρμογής.

## 3 Layer Architecture( Αναλυτικά )

### To Data Layer

Εδώ γίνεται η σύνδεση με τη βάση. Την πραγματοποιεί η PHP με την κλάση PDO. Επίσης δημιουργήτε μια βασική κλάση Database\_Object που περιέχει βασικές λειτουργίες της PDO κάθε μία ελεφρά προσαρμοσμένη στα δικά μας δεδομένα. Είναι η κλάση που θα κληρονομήσουν όλες οι επόμενες. Είναι η κλάση που θα επιτρέψει το extend των λειτουργιών της PDO με ιδιαίτερη ευκολία. Κάθε κλάση περιέχει βασικές μεθόδους που βοηθούν στην ανάκτηση των δεδομένων. Τέτοιες μπορεί να είναι η find, η select, η insert, η update, η delete που οι παραμετροί τους θα ορίζουν το ζητούμενο SQL query. Οι κλάσεις που δημιουργήσαμε είναι: Member, Photo, Tag, Comment και Search.

Ιδιαίτερο ενδιαφέρον παρουσιάζει η κλάση Photo. Πέρα από τις βασικές μεθόδους που αναφέραμε παραπάνω περιέχει και μεθόδους οι οποίες

πραγματοποιούν την μεταφόρτωση των φωτογραφιών στο σύστημα καθώς και το `resize` όπου αυτό είναι απαραίτητο (`attach_file`, `upload_and_save`). Χειρίζετε και το `multi upload` με την μέθοδο `unzip_and_instanciate` οι οποία αποσυμπίεζει ένα `zip` αρχείο σε ένα προσωρινό φάκελο και δημιουργεί ένα πίνακα αντικειμένων τύπου `photo` με όλες τις απαραίτητες ιδιότητες για την μεταφόρτωση καθώς και επιπλέον ιδιότητες αν αυτές υπάρχουν στο `info.xml` αρχείο. Στο `data layer` γίνεται επίσης ο συνδυασμός των `tags` με τις `photos` με την βοήθεια φυσικά τις `InnoDB engine`. Μέθοδοι όπως η `attach_to_photo`, `photo_tags` είναι αυτές που πραγματοποιούν αυτήν την σύνδεση.

### To Logic Layer

Είναι το `layer` που ενώνει το `data layer` με το `presentation layer` ή αλλιώς είναι αυτό το οποίο χρησιμοποιώντας τα επιστρεφόμενα αντικείμενα από το `data` θα δημιουργήσει τα ζητούμενα `smarty templates`.

Το `logic layer` καλεί όλες τις απαραίτητες μεθόδους για την εισαγωγή, επεξεργασία και διαγραφή όλων των οντοτήτων του συστήματος. Ελέγχει τις ενέργειες του χρήστη είτε αυτές είναι μέσω απλής `html` φόρμας είτε είναι αιτησεις `ajax`.

Επιπλέον αναλαμβάνει τον έλεγχο σύνδεσης ενός μέλους, καθώς και την προβολή μηνυμάτων λάθους, προειδωποίησης ή επιτυχίας. Για την υλοποίηση αυτών υπάρχει η κλάση `session` μέσα στο `library` του `logic layer` του συστήματος. Η κλάση `session` ανοίγει το `session` του `browser` στον `constructor` και ελέγχει την σύνδεση του μέλους με μεταβλητές `boolean(is_logged_in)`. Επίσης για λόγους ασφαλείας καλεί την `session_regenerate_id()` κάθε φορά που ένα μέλος θα κάνει σύνδεση.

Μέσα στο `library` υπάρχει η κλάση `image`. Η `image` περιέχει όλες τις μεθόδους που μας επιτρέπουν το `resize` μιας φωτογραφίας με διατήρηση του `ratio`, όπως και το `crop` με βάση το ποσοστό της φωτογραφίας που θέλουμε να φαίνεται στο τελικό αποτέλεσμα και φυσικά τις διαστάσεις αυτού. Η `crop` είναι αυτή που όχι μόνο δημιουργεί τα `thumbnails` μέσα στο `file system` αλλά και με την βοήθεια του `thumbs.php` δημιουργεί 'thumbnails on the fly'.

Επίσης υπάρχει και η κλάση `pagination` της οποίας κύρια λειτουργία είναι η παραγωγή του `offset` ανάλογα με το `current page` και φυσικά τα νούμερα της προηγούμενης και επόμενης σελίδας. Όλα αυτά χρησιμοποιούνται από την μέθοδο `select` της κλάσης `photo` και ως αποτέλεσμα υπάρχει η σελιδοποίηση των φωτογραφιών.

Τέλος, έχουμε κάνει **extend** το `Smarty Template System`, δημιουργώντας το "Tag Cloud" **plugin**. Με αυτό το τρόπο το `tag cloud` μπορεί και παραγεται καλώντας απλά την εντολή `Smarty`

**{tagcloud}**

Αυτη θα επιστρέψει ένα σύνολο απο **<anchor>** elements με **class='tag'** που θα αποτελούν το tag cloud μας.

### **To Presentation Layer (Smarty)**

Το layer που περιέχει όλα τα .tpl files, αποτελείτε από τον φάκελο themes που οι υποφάκελοί του αποτελούν όλα τα θέματα της σελίδας.

### **Theme Development**

Το κάθε θέμα της εφαρμογής αποτελείτε από templates.

Αναλυτικά :

Τα θέματα βασίζονται πάνω σε layouts, τα οποία χρησιμοποιούνται για ανάλογη παρουσίαση της εφαρμογής.

Συγκεκριμένα στη γενική περίπτωση υπάρχει το

#### **-page.layout.tpl**

*Δέχεται ένα σύνολο από Smarty μεταβλητές σχετικά με το album και τη γραφική απεικόνιση του quota αν και μόνο αν υπάρχει συνδεδεμένο μέλος,αλλιώς απλά εμφανίζει μια login φόρμα.*

Στην προβολή της αναζήτησης υπάρχει το

#### **-search.layout.tpl**

*Δέχεται τις Smarty μεταβλητές για την τρέχουσα αναζήτηση,και παρουσιάζει την extended φόρμα.*

ενώ στην προβολή μιας φωτογραφίας το

#### **-view.layout.tpl**

*Δέχεται ένα πίνακα Smarty αντικειμένων "comments" για τη φωτογραφία που γίνεται προβολή*

Το κάθε layout ανάλογα τη σελίδα που θα φορτώσει το αντίστοιχο template με τη βοήθεια της μεθόδου render() της βοηθητικής κλάσης Template() (η οποία υπάρχει στο library του logic area).

Τα templates που δέχονται το Smarty αντικείμενο της ζητούμενης φωτογραφίας είναι :

#### **-edit\_photo.tpl**

#### **-photo\_view.tpl**

Ένα πίνακα Smarty αντικειμένων photo δέχονται τα :

#### **-photos.tpl**

*ο πίνακας περιέχει της φωτογραφίες του συνδεδεμένου μέλους*

#### **-search.tpl**

*ο πίνακας περιέχει τα αποτελέσματα της τρέχουσας αναζήτησης*

#### **-home.tpl**

*ο πίνακας περιέχει τις δημοφιλήs φωτογραφίες της εφαρμογής*

Υπάρχουν επίσης τα templates που περιέχουν τις βασικές φόρμες λειτουργίας της εφαρμογής, αυτά είναι :

**-upload.tpl**

**-profile.tpl**

*δέχεται το smarty αντικείμενο με τις πληροφορίες του συνδεδεμένου μέλους*

**-login.tpl**

**-register.tpl**

*τα δύο τελευταία παράγονται με το simple.layout.tpl το οποίο απλά παρουσιάζει το ζητούμενο template σε 100% width*

Εδώ πρέπει να σημειώσουμε ότι η Javascript της εφαρμογής πραγματοποιείται με το JQuery javascript framework ενώ το Geolocation με ένα συνδυασμό της **HTML5** και **Google Maps API v3**.

Ιδιαίτερο ενδιαφέρον έχει το input **validation** που παράγεται με ένα αξιολάπητο JQuery plugin προσαρμοσμένο στο δικό μας γραφικό περιβάλλον.

## **Δομή φακέλων και ασφάλεια**

Όλες οι φωτογραφίες του συστήματος αποθηκεύονται στο root folder `"/uploads"` ο οποίος χρησιμοποιεί για την επαρκή ασφάλεια δεδομένων ένα `".htaccess"` file με την εντολή **"deny from all"** οι οποία αποτρέπει με σφάλμα `"403 / forbidden"` σε κάθε πιθανή αίτηση ενός url φωτογραφίας.

Οι φωτογραφίες όμως, προβάλλονται με τη βοήθεια δύο αρχείων *php* με όνομα `"photo.php"` και `"thumbs.php"` που προβάλλουν αυτούσια την φωτογραφία και προβάλλουν ένα thumbnail αυτής *"on-the-fly"* αντίστοιχα με τη βοήθεια της εντολής **header("content-type:...")** και της βοηθητικής κλάσης `Image()` που βρίσκεται στο library του logic layer. Και τα δύο παραπάνω αρχεία έχουν μοναδική παράμετρο (είσοδο) το μοναδικό id της αντίστοιχης φωτογραφίας.

Ο έλεγχος που πραγματοποιείται μέσα σε αυτά τα αρχεία τόσο για την υπάρχουσα σύνδεση ενός μέλους όσο και για τον έλεγχο ιδιοκτησίας της φωτογραφίας σε περίπτωση που αυτή είναι ιδιωτική καθιστά αυτά τα αρχεία απαραίτητα στην υλοποίηση αυτής της εφαρμογής

## **Τεχνολογία Ajax**

Οι αιτήσεις Ajax μέσα στην εφαρμογή δημιουργούνται με την χρήση της JQuery function `$.ajax()` (*Write less, Do more :P*)

Μια βασική χρήση της **ajax** γίνεται στο **autocomplete** όπου στέλνετε σε κάθε "**keyup** event" το περιεχόμενο του αντίστοιχου text input σε ένα αρχείο "tag.php" στο φάκελο ajax που με τη βοήθεια της κλάσης Tag() ανακτούμε τις πιθανές προτάσεις για το current string του text input. Επίσης υπάρχει τεχνολογία ajax για την διαγραφή των tags μιας φωτογραφίας όπως και για την live προβολή των πληροφοριών της μέσα στο album του μέλους.

## Geolocation

Όπως αναφέραμε παραπάνω το geolocation πραγματοποιείται με το google maps API v3 και χρησιμοποιείται σε δύο σημεία

1. Στην επεξεργασία μιας φωτογραφίας
2. Στον αρχικό χάρτη της εφαρμογής

Αναλυτικά, στην επεξεργασία της εικόνας μετά από κάθε "click" ή "dragend" event γράφονται δύο **hidden** inputs με το *latitude* και *longitude* αντίστοιχα, ενώ ένας **Geocoder** μεταφράζει την αντίστοιχη διεύθυνση σε ένα τρίτο input

Στον αρχικό χάρτη η πρώτη προσπάθειά μας να παρουσιάσουμε μικρά thumbnails των φωτογραφιών γύρω από την τοποθεσία του επισκέπτη ή μέλους έγινε με επιτυχία, αλλά παρατηρήσαμε μια έντονη μείωση της απόδοσης του χάρτη.

Έτσι αποφασίσαμε να δημιουργήσουμε ένα custom marker icon στατικό για όλες τις φωτογραφίες και με τη βοήθεια ενός info box προβάλαμε τα thumbnails των φωτογραφιών on-the-fly βελτιώνοντας έτσι σημαντικά την απόδοση.