Ανάπτυξη & Σχεδίαση Λογισμικού

Project STF Σύστημα Τροφοδοσίας Φαρμακείων

Ομάδα: Μεταπτυχιακοί:

Λορέντζος Δημήτριος-Σωτήριος ΑΕΜ: 421

Μπακρατσάς Μάριος ΑΕΜ: 426

Ψαθά Άννα ΑΕΜ: 415

Προπτυχιακοί:

Βλάχος Ξάνθος ΑΕΜ: 1072

Καλφούντζος Δημήτριος ΑΕΜ: 485

Χατζηκωτούλας Σπυρίδων ΑΕΜ: 300

Σχεδιαστικές επιλογές

Μενού επιλογών ανά χρήστη (User's, Admin's) και sessions

Για το διαχωρισμό των διάφορων λειτουργιών που εκτελούνται από το διαχειριστή και από το χρήστη έχουν σχεδιαστεί δυο διαφορετικά μενού επιλογών. Ανάλογα με το είδος του χρήστη που συνδέεται στο σύστημα εμφανίζεται και το ανάλογο μενού. Σε αυτό έχουμε χρησιμοποιήσει sessions ελέγχοντας κάθε φορά το 'role' του χρήστη ($_SESSION['role']=="A" για τον διαχειριστή ή $_SESSION['role']=="U" για το χρήστη-φαρμακοποιό).

Κρυπτογράφηση ευαίσθητων δεδομένων

Οι διάφοροι κωδικοί των χρηστών, όταν κάνουμε εγγραφή, κρυπτογραφούνται με τον αλγόριθμο SHA-256 αφού πρώτα έχουμε προσθέσει μπροστά από το κωδικό ένα μοναδικό αλφαριθμητικό (SALT) που δημιουργείται. Με αυτή μας την επιλογή επιτυγχάνουμε μεγαλύτερη ασφάλεια από διάφορες κακόβουλες επιθέσεις.

Δημιουργία μοναδικού αλφαριθμητικού για εικόνες

Στην εισαγωγή προϊόντος εισάγουμε εικόνα. Για την αποφυγή ύπαρξης ίδιων ονομάτων στα αρχεία εικόνων καλούμε τη συνάρτηση create_guid() η οποία δημιουργεί ένα μοναδικό αλφαριθμητικό εισάγοντας τόσο στο φάκελο upload_images την εικόνα όσο και στη βάση δεδομένων μας το μοναδικό αλφαριθμητικό.

Κλήση αρχείων στο κώδικα για εξέλιξη και συντήρηση

Για την καλύτερη διαχείριση του Project μας από άποψη ανάπτυξης λογισμικού έχουμε χωρίσει σε αρχεία PHP το κώδικα μας καλώντας τον κάθε φορά όπου χρειάζεται. Μερικά παραδείγματα είναι τα require 'header.php' , require 'menu.php' , require 'main.php', & require 'footer.php' που καλούμε στην αρχική σελίδα index.php , καθώς επίσης και το require ' db_connect.php ' για να δημιουργήσουμε ένα νέο αντικείμενο της βιβλιοθήκης PDO ,να επιλέξουμε τη βάση δεδομένων που χρησιμοποιούμε ,όνομα χρήστη και κωδικό στη MySQL. Με αυτό τον τρόπο μπορούμε να κάνουμε τις αλλαγές σε ένα αρχείο κι αυτές με τη σειρά τους να γίνονται όπου έχουμε κάνει τα διάφορα require. Ένα ακόμα πλεονέκτημα είναι και η μελλοντική εξέλιξη του κώδικα μας που συμβάλλουν τα διάφορα require. Λίγο παραπάνω αναφερθήκαμε στη βιβλιοθήκη PDO η οποία αποτελεί επέκταση της PHP και μας παρέχει μια τυπική προγραμματιστική διεπαφή μεταξύ της PHP και του RDBMS που χρησιμοποιείται. Κάθε ερώτημα που πριν εκτελεστεί «προετοιμάζεται» (πχ $statement-> $pdoObject ->prepare($sql_query);) δηλαδή γίνεται compile κι έτσι αποφεύγονται κακόβουλες ενέργειες όπως το SQL Injection .

Έλεγχοι

Κάτι πολύ σημαντικό που λάβαμε υπόψη μας ήταν να κάνουμε αρκετούς ελέγχους κατά την εισαγωγή και επεξεργασία χρηστών και προϊόντων για αρνητικές τιμές ή μηδέν ή ακόμα για πεδία που δεν είχαν συμπληρωθεί αλλά και γενικά σε όλα τα πεδία στις διάφορες φόρμες συμπλήρωσης. Ορίστηκαν έλεγχοι στα διάφορα inputs της html μέσω patterns (regular expressions) όσον αφορά τον τύπο των τιμών και το μήκος τους (π.χ. email address κατά την εισαγωγή νέου χρήστη από το διαχειριστή). Επιπρόσθετα οι ίδιοι έλεγχοι γίνανε

και μέσω της php (validStrLen συνάρτηση και έλεγχοι τύπων). Σημαντικοί έλεγχοι που έγιναν ήταν και στις παραγγελίες όπου για να παραγγείλει ένα προϊόν ο χρήστης θα έπρεπε πρώτα να υπάρχει η διαθέσιμη ποσότητα. Επίσης έλεγχοι έγιναν και για τυχόν εκπτώσεις σε παραγγελίες χρηστών με βάση τον αριθμό παραγγελιών που έχει κάνει ο χρήστης και ελέγχοντας το ranking του εκάστοτε πελάτη.

PayPal

Για τη δημιουργία της φόρμας παραγγελίας (οπτικά βλέπεις απλά ένα Button "Checkout Order", όλα τα υπόλοιπα πεδία είναι hidden) δημιουργήσαμε τη συνάρτηση *createPaymentFormButton($products_list, $batchOrderID)* στο αρχείο class/helper.php που κατασκευάζει δυναμικά τα πεδία της φόρμας για όλα τα προϊόντα που έχουμε στο shopping cart. Επίσης παίρνει σαν όρισμα έναν «τυχαίο» αριθμό που αντιπροσωπεύει τον αριθμό της συγκεκριμένης παραγγελίας. Το PayPal κατά την αποστολή μιας φόρμας μπορεί να «κουβαλήσει» μαζί της και μια custom μεταβλητή.

Αφού ολοκληρωθεί η διαδικασία της συναλλαγής, επιστρέφει το PayPal ένα String (IPN notification) με τις πληροφορίες της συναλλαγής συμπεριλαμβανομένης και αυτής της μεταβλητής, σε ένα συγκεκριμένο link (που δείχνει στον server της εφαρμογής μας). Με αυτόν τον τρόπο μπορούμε να γνωρίζουμε το αποτέλεσμα μιας συγκεκριμένης συναλλαγής. Κάθε προϊόν της παραγγελίας έχει και το δικό του order_ID, οπότε σε περίπτωση περισσότερων του ενός προϊόντος στο shopping cart χρειαζόμαστε ένα *batch_order_id* για την ενοποίηση. Το αρχείο IPNreturn.php λειτουργεί σαν *listener* του IPN notification που επιστρέφει το PayPal, ενώ το class/paypal.php το διασπά σε επιμέρους μεταβλητές μέσω της συνάρτησης validateTransaction() επιστρέφοντας, το αποτέλεσμα της συναλλαγής (verified ή failed) καθώς επίσης και την custom μεταβλητή.

Μέσω του session περνάμε έναν πίνακα με πληροφορίες για κάθε προϊόν του shopping cart (id κάθε προϊόντος, και λοιπές πληροφορίες) και σε περίπτωση που τη αποτέλεσμα της validateTransaction() είναι verified τότε τα συγκεκριμένα προϊόντα λαμβάνουν τον χαρακτηρισμό Paid.