ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Διαδικτυακό Σύστημα Δηλώσεων Μαθημάτων Ίδρυμα Τριτοβάθμιας Εκπαίδευσης

e-course

Του φοιτητή

Στυλιανός Τζουρέλης

Αρ. Μητρώου: 134150

Επιβλέπων καθηγητής

Αντώνης Σιδηρόπουλος

Θεσσαλονίκη 2018

ΠΡΟΛΟΓΟΣ

Η εργασία αυτή είχε ως αφορμή τον προσωπικό μου προβληματισμό όσον αφορά το δυσλειτουργικό σύστημα δηλώσεων μαθημάτων, που επικρατούσε στο τμήμα μας. Ύστερα από συζήτηση με τον πρόεδρο της σχολής σχετικά με το θέμα αυτό μου ανατέθηκε να παρουσιάσω μια πρόταση για την επίλυση του προβλήματος. Η καινούρια λοιπόν πλατφόρμα που δημιούργησα, η οποία αναπτύχθηκε σε php, javascript και για βάση δεδομένων χρησιμοποιήθηκε MySql, αποσκοπεί στην επίλυση κάποιων βασικών προβλημάτων που ταλαιπωρούν προσωπικό και φοιτητές κάθε εξάμηνο κατά την διάρκεια των δηλώσεων. Το παρόν έγγραφο περιέχει την παρουσίαση της πλατφόρμας αυτής, οδηγίες για την χρήση και την συντήρηση της.

ΠΕΡΙΛΗΨΗ

Το e-course έχει γραφτεί σε php χρησιμοποιώντας την έκδοση 7. Δεν χρησιμοποιήθηκε κάποιο framework έτσι ώστε να απαιτεί τα ελάχιστα από το hardware στο οποίο θα φιλοξενείται Έχει δοθεί ιδιαίτερη προσοχή τόσο στην εύκολη χρήση από τους διαχειριστές του (Γραμματεία και καθηγητές) όσο και από τους φοιτητές. Ενδεικτικά η πλατφόρμα περιλαμβάνει διαχείριση προγράμματος σπουδών, αντιστοίχηση παλαιών μαθημάτων σε καινούρια καρτέλα, δήλωση κατευθύνσεων, σύστημα υπενθυμίσεων και περιοχές που μπορούν οι διαχειριστές του προγράμματος να το συντηρούν και να κάνουν αλλαγές σε αυτό εύκολα. Συγκεκριμένα, για τους καθηγητές προσφέρεται η λίστα φοιτητών που εγγράφονται στα εργαστήρια τους καθώς και η προβολή προγράμματος των μαθημάτων. Τέλος, για τους φοιτητές προσφέρεται μια εύκολη και responsive λύση για την δήλωση των μαθημάτων τους, ανταλλαγή εργαστηρίων μετά το πέρας των δηλώσεων (εάν το τμήμα διαλέξει να αφήσει ενεργή αυτή την επιλογή), προβολή του προγράμματος των θεωριών και εργαστηρίων.

ABSTRACT

E-course is written in php using version 7. No frameworks were used during the development in order to keep the hardware requirements at a minimum. The ease of use for this platform was a high priority both for the faculty and the students that they will be using it. The platform contains structure of studies manipulation, old to new studies matching, registration to specific directions by the students, a simple notification system and a specific form to do maintenance and targeted changes to the core code. For the department's teachers the e-course offers students list per lab and a view of the day to day schedule. Lastly the platform offers to students an easy and responsive solution to registering to courses each semester, mutual trades to labs positions, and a view of the day to day schedule.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον κύριο Σιδηρόπουλο για την άψογη συνεργασία και βοήθεια που προσέφερε κατά την διεκπεραίωση της πτυχιακής μου εργασίας.

ΠΕΡΙΕΧΟΜΕΝΑ

Ευρετήριο σχημάτων

ΕΙΣΑΓΩΓΗ

Το e-course αναπτύχθηκε με σκοπό να αντικαταστήσει την τωρινή και προβληματική διαδικασία των δηλώσεων με μια πιο εύχρηστη και λειτουργική τόσο για τους φοιτητές όσο και για το προσωπικό του τμήματος λύση. Ο λόγος που επιλέχθηκε το συγκεκριμένο θέμα σαν πτυχιακή, είναι επειδή θεωρούσα ανεπίτρεπτο την έλλειψη εργαλείων και λύσεων από πλευράς του Τ.Ε.Ι. για την διαδικασία των δηλώσεων, μια εικόνα που δεν αρμόζει στο υψηλό επίπεδο του προσωπικού και του σπουδαστικού κοινού. Η πτυχιακή εκπονήθηκε έπειτα από πολλές ώρες σχεδιασμού και διαδοχικών συναντήσεων με το προσωπικό του Τ.Ε.Ι. έτσι ώστε να είναι προσαρμοσμένη στις ανάγκες του. Αντιμετωπίστηκαν πολλά προβλήματα κατά την διαδικασία την ανάπτυξης της πλατφόρμας όπως η προσαρμογή της σε ένα συνεχώς αναπτυσσόμενο πρόγραμμα σπουδών. Έπρεπε να σχεδιαστεί μια διαδικασία δηλώσεων όπου θα προσέφερε όλες τις επιλογές των καινούριων μαθημάτων αλλά συγχρόνως δεν θα άφηνε πίσω τους φοιτητές από παλαιότερα προγράμματα. Ένα άλλο πρόβλημα που συναντήθηκε ήταν η κατανομή των φοιτητών σε εργαστήρια καθώς και η προτεραιότητα που θα έπρεπε να επιλέξουμε. Έπειτα από πολλές δοκιμές καταλήξαμε σε έναν custom αλγόριθμο που θα αναλυθεί παρακάτω. Στο παρόν έγγραφο περιλαμβάνονται οδηγίες παραμετροποίησης του συστήματος στο εκάστοτε πρόγραμμα σπουδών, όπου αναλύεται η διαδικασία προετοιμασίας του συστήματος για την περίοδο των δηλώσεων. Επίσης περιλαμβάνεται ένα κεφάλαιο με οδηγίες και εξηγήσεις για των κώδικα και πως αυτός πρέπει να συντηρείται, οδηγίες χρήσης για φοιτητές, καθηγητές και γραμματεία. Για την ανάπτυξη της πτυχιακής χρησιμοποιήθηκε php και διάφορα site σχετικά με τον προγραμματισμό για να αντιμετωπιστούν τυχόν προβλήματα (π.χ. https://stackoverflow.com/).

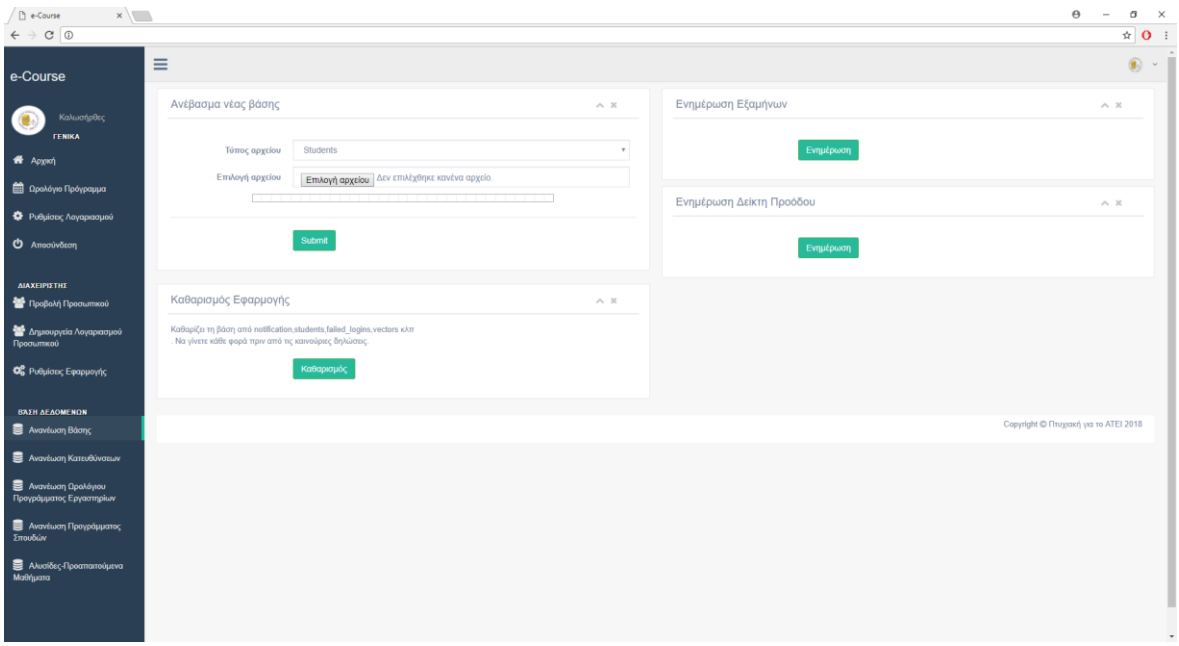
ΚΕΦΑΛΑΙΟ 1
ΔΙΑΔΙΚΑΣΙΑ ΔΗΛΩΣΕΩΝ

ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο περιγράφονται αναλυτικά οι διαδικασίες που θα οδηγήσουν στην σωστή προετοιμασία της πλατφόρμας, έτσι ώστε να είναι επιτυχημένη η διαδικασία των δηλώσεων. Πιο αναλυτικά, οι ενέργειες που απαιτούνται είναι η εισαγωγή των αναγκαίων αρχείων από την πυθία, η ανανέωση του προγράμματος σπουδών της σχολής, των προαπαιτούμενων μαθημάτων (σε περίπτωση που αυτά υπάρχουν) και η ανανέωση του ωρολογίου προγράμματος. Επίσης θα εξετάσουμε τους κωδικούς που προσφέρει η πλατφόρμα για την τροποποίηση της και προσαρμογή σε οποιοδήποτε ανάγκη της σχολής. Οι ενέργειες αυτές πρέπει να εκτελούνται από προγραμματιστές που έχουν συμβουλευτεί τις οδηγίες χρήσης και έχουν κατανοήσει πλήρως την λειτουργία του e-course.

1.1 ΡΥΘΜΙΣΗ ΤΗΣ ΒΑΣΗΣ

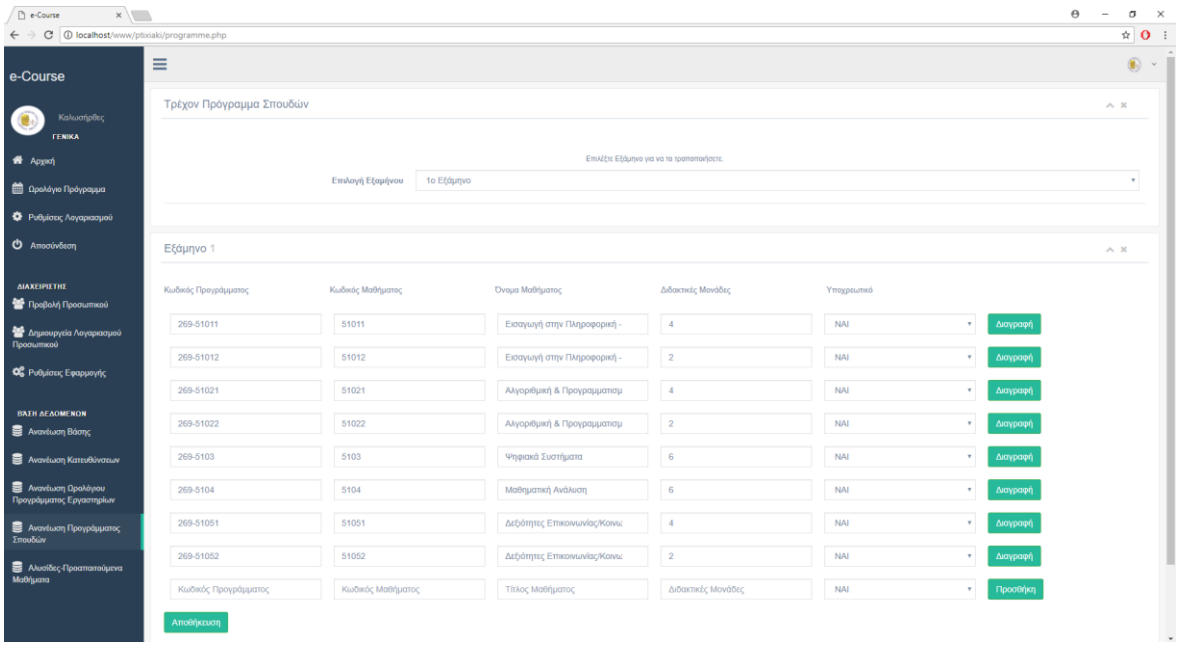
Η διαδικασία της ανανέωσης ξεκινά από τους διαχειριστές του συστήματος οι οποίοι πρέπει να προνοήσουν να υπάρξει μια καθαρή βάση. Η επιλογή σβησίματος των αντίστοιχων table δίνεται και μέσα από την εφαρμογή αλλά προς αποφυγή αστοχιών καλό θα ήταν να εκτελείται κατευθείαν από τον διακομιστή της βάσης δεδομένων. Στη συνέχεια αφού γίνει η εξαγωγή των αντίστοιχων αρχείων από την πυθία, αυτά περνιούνται στο e-course μέσα από τη σελίδα «Ανανέωση Βάσης» επιλέγοντας τον κατάλληλο τύπο αρχείου. Το αρχείο students.txt περιλαμβάνει τα στοιχεία των φοιτητών (αριθμό ειδικού μητρώου , την κατάσταση του φοιτητή, το εξάμηνο που βρίσκετε την ημερομηνία εγγραφής του κλπ.). Το αρχείο classStudents.txt περιέχει όλα τα μαθήματα που έχει δηλώσει στο παρελθόν ο φοιτητής (το έτος δήλωσης τους καθώς και το εξάμηνο, τον βαθμό της εξεταστικής τους κλπ.).Το τελευταίο αρχείο που αφορά τους φοιτητές είναι το studentCourses.txt στο οποίο περιλαμβάνονται μόνο τα περασμένα μαθήματα των σπουδαστών σε αντίθεση με το class που τα περιλαμβάνει όλα (εκτός αυτής της ιδιαιτερότητας, τα indexes των δυο αρχείων είναι ίδια). Το τελευταίο αρχείο που μπορεί να φορτωθεί σε αυτήν την σελίδα είναι το schedule.txt , το οποίο περιλαμβάνει το ωρολόγιο πρόγραμμα της σχολής. Το συγκεκριμένο txt φορτώνει στη σελίδα «Ωρολόγιο Πρόγραμμα» τις ώρες των εργαστηρίων και θεωριών έτσι ώστε να είναι προσβάσιμες από όλους τους χρήστες της εφαρμογής. Επίσης, με αυτόν τον τρόπο περνιούνται και οι επιλογές εργαστηριακών τμημάτων που εμφανίζονται αργότερα στους φοιτητές κατά την διάρκεια δηλώσεων των μαθημάτων.



Σχήμα 1 " Σελίδα Ανανέωση Βάσης"

Μετά το φόρτωμα των αρχείων από την πυθία στο e-course, πρέπει να γίνει η ενημέρωση εξαμήνου των φοιτητών και η ενημέρωση του δείκτη προόδου των φοιτητών. Ο δείκτης προόδου ανανεώνεται ξεχωριστά για τον κάθε φοιτητή και είναι ένας σημαντικός παράγοντας για την προτεραιότητα κατανομής σε εργαστηριακά τμήματα. Ο δείκτης υπολογίζεται παίρνοντας υπόψη τα δηλωμένα μαθήματα, τα περασμένα μαθήματα καθώς και το εξάμηνο στο οποίο βρίσκεται ο φοιτητής.

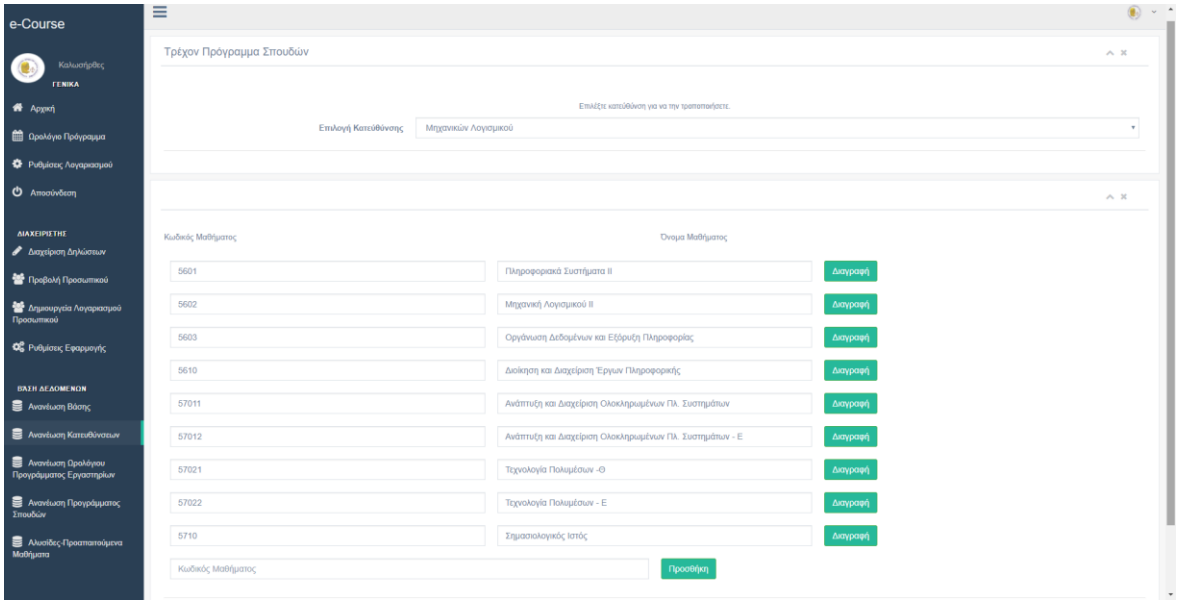
Επόμενο βήμα στην παραμετροποίηση του συστήματος είναι η ενημέρωση του προγράμματος σπουδών (σε περίπτωση που αυτό δεν έχει αλλάξει θα γίνεται μόνο την πρώτη φορά που χρησιμοποιείται η εφαρμογή).Η συγκεκριμένη ενέργεια γίνεται από την σελίδα «Ανανέωση Προγράμματος Σπουδών». Επιλέγοντας συγκεκριμένο εξάμηνο σπουδών εμφανίζονται τα μαθήματα που είναι αποθηκευμένα μέσω της τεχνολογίας AJAX χωρίς να γίνει ανανέωση της σελίδας. Εκτός από τα μαθήματα που είναι αποθηκευμένα δίνεται η επιλογή διαγραφής τους καθώς και η προσθήκη καινούριου μαθήματος. Τα πεδία που πρέπει να συμπληρωθούν είναι τα εξής, Κωδικός Προγράμματος, Κωδικός Μαθήματος, Όνομα Μαθήματος, Διδακτικές Μονάδες (διδακτικές μονάδες που πιστώνουμε στον φοιτητή κατά την δήλωση του), Υποχρεωτικό (αν το μάθημα είναι αναγκαστικό η είναι επιλογής)



Σχήμα 2 " Σελίδα Ανανέωση Προγράμματος Σπουδών **"**

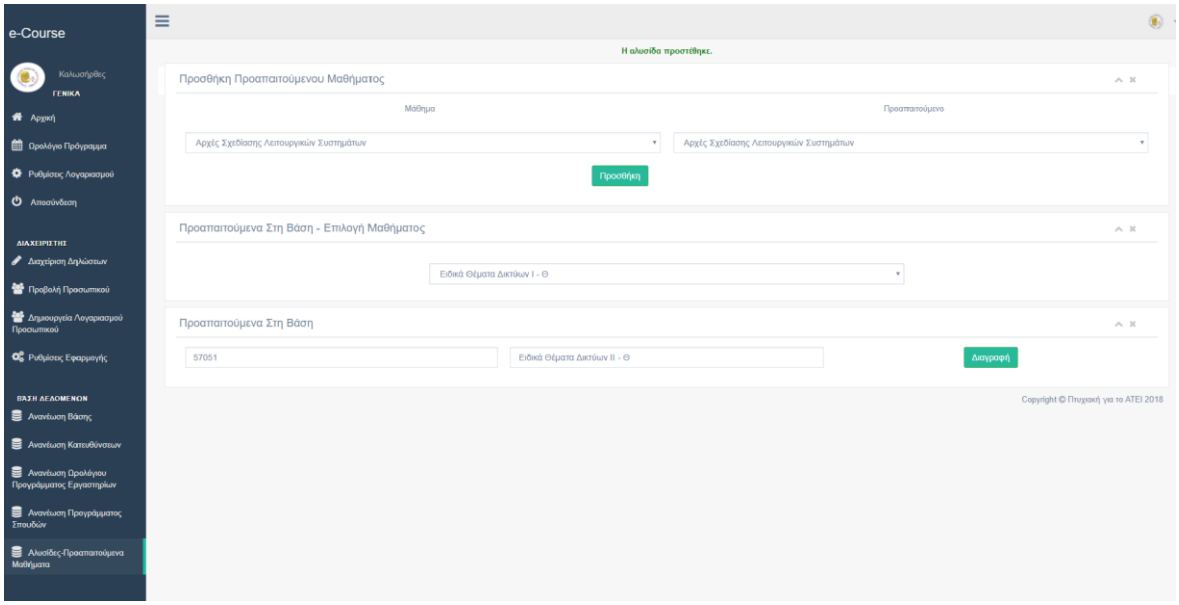
1.2 ΡΥΘΜΙΣΗ ΚΑΤΕΥΘΥΣΕΩΝ ΚΑΙ ΠΡΟΑΠΑΙΤΟΥΜΕΝΩΝ

Το επόμενο βήμα στη ρύθμιση του e-course πριν από τις δηλώσεις είναι η ανανέωση των κατευθύνσεων. Ή επιλογή κατεύθυνσης γίνεται στο 6ο εξάμηνο φοίτησης. Το εξάμηνο επιλογής μπορεί να αλλάξει ανά πάσα στιγμή από τις ρυθμίσεις συστήματος. Πηγαίνοντας στην σελίδα «Ανανέωση Κατευθύνσεων» μπορούμε να επιλέξουμε μια από τις διαθέσιμες-αποθηκευμένες κατευθύνσεις (καινούριες μπορούν να προστεθούν μόνο από την βάση προς το παρόν).Μετά την επιλογή μια κατεύθυνσης , εμφανίζονται τα μαθήματα που είναι υποχρεωτικά για αυτήν την κατεύθυνση στα πεδία κωδικός μαθήματος και όνομα μαθήματος,, καθώς επίσης και η επιλογή να διαγράψουμε ένα από αυτά. Στο κάτω μέρος της σελίδας δίνεται η επιλογή να προστεθεί κάποιο μάθημα στη συγκεκριμένη κατεύθυνση συμπληρώνοντας τον κωδικό του και πατώντας προσθήκη.



Σχήμα 3 " Σελίδα Ανανέωση Κατευθύνσεων "

Τελευταίο βήμα στη ρύθμιση του συστήματος πριν από τις δηλώσεις είναι η ρύθμιση των προαπαιτούμενων μαθημάτων σε περίπτωση που αυτό επιλεγεί από το τμήμα. Πηγαίνοντας στην σελίδα «Αλυσίδες-Προαπαιτούμενα Μαθήματα» στο επάνω μέρος δίνεται η δυνατότητα προσθήκης ενώ στο κάτω μέρος επιλέγοντας ένα συγκεκριμένο μάθημα εμφανίζονται τα αποθηκευμένα προαπαιτούμενα. Για παράδειγμα προσθέτοντας ως προαπαιτούμενο τα “Ειδικά Θέματα Δικτύων 1” για τα “Ειδικά Θέματα Δικτύων 2”, το μάθημα αποθηκεύεται στη βάση και δεν επιτρέπει τους φοιτητές να το δηλώσουν αν δεν έχουν περάσει το πρώτο.



Σχήμα 4 " Σελίδα Αλυσίδες-Προαπαιτούμενα Μαθήματα "

1.3 ΡΥΘΜΙΣΗ ΚΑΙ ΠΑΡΑΔΕΙΓΜΑΤΑ ΧΡΗΣΗΣ ΚΩΔΙΚΩΝ

Σε αυτό το τμήμα της πτυχιακής θα αναλύσουμε τους 4 σημαντικούς κώδικες που αναφέρονται σε άλλα κεφάλαια και πως να χρησιμοποιούνται καθώς και παραδείγματα χρήσης. Οι κώδικες αυτοί προσφέρουν μέρη όπου ο προγραμματιστής που συντηρεί την πλατφόρμα μπορεί με λίγες γραμμές κώδικα να την προσαρμόσει σε μελλοντικές ανάγκες της σχολής.

1.3.1 ΚΩΔΙΚΑΣ ΑΠΟΔΟΧΗΣ ΜΑΘΗΜΑΤΟΣ

Ο πρώτος κώδικας που θα αναλυθεί είναι Κώδικας Αποδοχής Μαθήματος.Ο συγκεκριμένος κώδικας "τρέχει" για κάθε μάθημα που δήλωσε ο φοιτητής. Αν βάλουμε return true; σε αυτόν τον κώδικα δεν δέχεται το μάθημα που τρέχει στο loop και μπορούμε να εμφανίσουμε μήνυμα λάθους με την μεταβλητή $_SESSION['form_error']. Μπορούν να χρησιμοποιηθούν οι εξής μεταβλητές, $passed_courses Πίνακας με τα course_ID όλων των περασμένων μαθημάτων του φοιτητή, $registered_courses Πίνακας με τα course_ID όλων των δηλωμένων μαθημάτων του φοιτητή, $study Το μάθημα που ελέγχουμε σε κάθε iteration του loop, $student Ο μαθητής που στέλνει τη δήλωση. Πίνακας με τα εξής indexes: student_ID, aem, first,last, fname, in_year, in_period_ID

student_status, deletion_reason, in_semester_ID, semester_ID, progress_indicator, $requested Πίνακας με τα course_ID όλων των μαθημάτων που δήλωσε ο φοιτητής. Ένα χρήσιμο παράδειγμα καθώς και έξυπνη χρήση του

συγκεκριμένου εργαλείου θα ήταν η εξής: Έστω ότι δημιουργούμε ένα μάθημα και το περνάμε στο πρόγραμμα σπουδών με id 75643 και όνομα “Συστήματα ελέγχου έκδοσης”. Αν θέλουμε αυτό το μάθημα να μπορούν να το δηλώσουν μόνο φοιτητές που έχουν γραφτεί μετά το 2014 στη σχολή, τότε το μόνο που έχουμε να κάνουμε είναι να αποθηκεύσουμε τον παρακάτω κώδικα στις ρυθμίσεις της εφαρμογής.

```
if($study==' 75643' && $student['in_year'] < 2014)
{
    $_SESSION['form_error']= 'Το μάθημα αυτό είναι διαθέσιμο για φοιτητές με έτος εγγραφής μεγαλύτερο του 2014';
    return true;
}
```

Αυτό σημαίνει ότι αν ο φοιτητής έχει έτος εισαγωγής μικρότερο του 2014 τότε αρχικοποιούμε $_SESSION['form_error'] για να ενημερωθεί ο φοιτητής και κάνουμε return true (που σημαίνει πως ο κώδικας αποδοχής μαθημάτων απέτυχε δηλαδή break). Ένας άλλος τρόπος χρήσης θα ήταν το εξής, για το ίδιο μάθημα που φτιάξαμε στο προηγούμενο παράδειγμα, θα θέλαμε να μπορούν να το δηλώσουν μόνο οι φοιτητές που έχουν περάσει το εργαστήριο “Εισαγωγή στην Πληροφορική” με κωδικό μαθήματος 51012. Τότε όπως και στο προηγούμενο παράδειγμα το μόνο που έχουμε να κάνουμε είναι:

```
if($study==' 75643' && !in_array('51012', $passed_courses))
{
    $_SESSION['form_error']="'Πρέπει να έχετε περάσει το εργαστήριο Εισαγωγή στην Πληροφορική για να δηλώσετε το μάθημα.";
    return true;
}
```

Αυτό είναι ένα use case που δεν συνιστάται καθώς μέσα στην εφαρμογή υλοποιείται ήδη η διαδικασία των αλυσίδων, απλά είναι ένας τρόπος να δείξουμε τις δυνατότητες της εφαρμογής. ‘Ενα εύστοχο ερώτημα που προκύπτει είναι: εφόσον δεν δεχόμαστε το μάθημα γιατί δεν επιστρέφουμε false αντί για true. Η απάντηση διαμορφώνεται από τον τρόπο λειτουργείας της eval. Πριν την έκδοση 7.0 της php η eval επιστρέφει false σε περίπτωση συντακτικού λάθους. Επομένως αν επιστρέφαμε false και εμείς σαν αποτέλεσμα του κώδικα σε περίπτωση λάθους δεν θα αναγνωρίζαμε αν ο κώδικας ολοκληρώθηκε με επιτυχία ή έχουμε κάποιο συντακτικό λάθος. Αξίζει να σημειωθεί ότι μετά την έκδοση 7 της php η eval πετάει μήνυμα λάθους σε περίπτωση συντακτικού αντί για false. Με αυτό το σκεπτικό διαμορφώθηκαν και οι υπόλοιποι κώδικες που χρησιμοποιούν την eval.

1.3.2 ΚΩΔΙΚΑΣ ΠΡΟΒΟΛΗΣ ΜΑΘΗΜΑΤΟΣ

Ο κώδικας προβολής μαθημάτων έχει ακριβώς τον ίδιο τρόπο χρήσης με τον κώδικα αποδοχής. Μπορούν να χρησιμοποιηθούν οι ίδιοι ακριβώς πίνακες με με

την διαφορά ότι το μάθημα που ελέγχουμε δεν είναι η μεταβλητή $study του τύπου string, αλλά η μεταβλητή row που είναι τύπου array με τα έξεις indexes curriculum_code, course_ID, course_name, type enum('oblig', 'option'), units,semester_ID. Με return true ΔΕΝ εμφανίζεται το μάθημα που έχουμε στο loop εκείνη την στιγμή (το $study δηλαδή).Για παράδειγμα, έστω ότι περνάμε ένα μάθημα με κωδικό 75643 όπου θέλουμε να εμφανίζεται μόνο στους μαθητές που μπήκαν το 2018 και μετά στην σχολή. Αυτό γίνεται με τον εξής τρόπο :

```
if($row['course_ID'] == '75643' && $student['in_year'] < 2018)
    return true;
```

Η συνθήκη ελέγχει αν το μάθημα που ελέγχουμε είναι αυτό με τον κωδικό 75643 και αν η εισαγωγή του φοιτητή πραγματοποιήθηκε πριν το 2018, τότε δεν το εμφανίζει ως επιλογή.

1.3.3 ΚΩΔΙΚΑΣ ΠΡΟΒΟΛΗΣ ΜΑΘΗΜΑΤΟΣ

Στη συνέχεια έχουμε τον κώδικα αναγκαίων μαθημάτων και αυτός όπως και οι 2 προηγούμενοι χρησιμοποιούν τις ίδιες μεταβλητές και όπως και τον κώδικα προβολής το μάθημα που ελέγχουμε είναι της μορφής row (array). Ο κώδικας αυτός με return true εμφανίζει το μάθημα επιλεγμένο και αναγκαίο στην δήλωση ,ο φοιτητής επομένως είναι αναγκασμένος να το δηλώσει. Εάν θέλαμε για παράδειγμα να "αναγκάσουμε" τους φοιτητές να δηλώσουν 30 διδακτικές ώρες τουλάχιστον κάθε εξάμηνο το μόνο που έχουμε να κάνουμε είναι:

```
if($hoursFromRequired <30 && $row['type'] == 'oblig')
    return true;
```

Στην συνθήκη ελέγχουμε, αν οι ώρες που έχει μαζέψει ο φοιτητής από τις μέχρι τώρα δηλώσεις είναι μικρότερες τον 30 και το μάθημα είναι τύπου υποχρεωτικό, τότε το μαρκάρουμε σαν υποχρεωτικό.

1.3.4 ΚΩΔΙΚΑΣ ΠΡΟΒΟΛΗΣ ΜΑΘΗΜΑΤΟΣ

Επόμενος είναι ο κώδικας κατοχύρωσης. Αυτός ο κώδικας αποφασίζει αν θα εμφανιστεί η επιλογή κατοχύρωση σε κάποιο εργαστήριο που πρέπει να δηλώσει ο φοιτητής. Ο κώδικας αυτός χρησιμοποιεί τις ίδιες μεταβλητές με τους προηγούμενους κώδικες. Ως προεπιλεγμένες ρυθμίσεις αυτός ο κώδικας περιλαμβάνει τα εξής

```
if(in_array($row['course_ID'] ,$registered_courses))
    return true;
```

Αν δηλαδή το μάθημα που ελέγχουμε τώρα βρίσκεται σε μάθημα που έχει δηλώσει παλαιότερα τότε εμφανίζει την επιλογή κατοχύρωση. Ο κώδικας αυτός μπορεί να επεκταθεί ελέγχοντας με τι βαθμό κόπηκε ο φοιτητής την προηγούμενη φορά που το δήλωσε κλπ.

Τελευταίος κώδικας είναι ο αυτός της αντιστοίχησης μαθημάτων. Ο κώδικας αυτός τρέχει κάθε φορά που περνάμε περασμένα και δηλωμένα μαθήματα των φοιτητών και τα αντιστοιχεί με τα καινούρια (αυτά που βρίσκονται στο τρέχων πρόγραμμα σπουδών). Μερικά παραδείγματα αντιστοίχησης μαθημάτων από P3 σε P4 και P4 σε P5.

```
$conn->query("UPDATE registered_courses SET course_ID=4303 WHERE
course_ID=3202;");
$conn->query("UPDATE registered_courses SET course_ID=5401 WHERE
course_ID=4402;");
```

ΕΠΙΛΟΓΟΣ

Σε αυτό το κεφάλαιο είδαμε όλες τις ενέργειες που πρέπει να ολοκληρωθούν προκειμένου να προχωρήσουμε στη διαδικασία των δηλώσεων καθώς και τους κώδικες που μπορούμε να χρησιμοποιήσουμε Με τον συνδυασμό των παραπάνω κωδίκων μπορούν να γίνουν πολλές αλλαγές στην πλατφόρμα και μπορεί να προσαρμοστεί στης ανάγκες της σχολής είτε αυτό αφορά τις αλυσίδες μαθημάτων είτε διαφορετικά προγράμματα σπουδών για διαφορετικά έτη εισαγωγής φοιτητών κλπ. Στο επόμενο κεφάλαιο θα δούμε την διαδικασία των δηλώσεων από την πλευρά την φοιτητών, των διαχειριστών και των καθηγητών.

ΚΕΦΑΛΑΙΟ 2

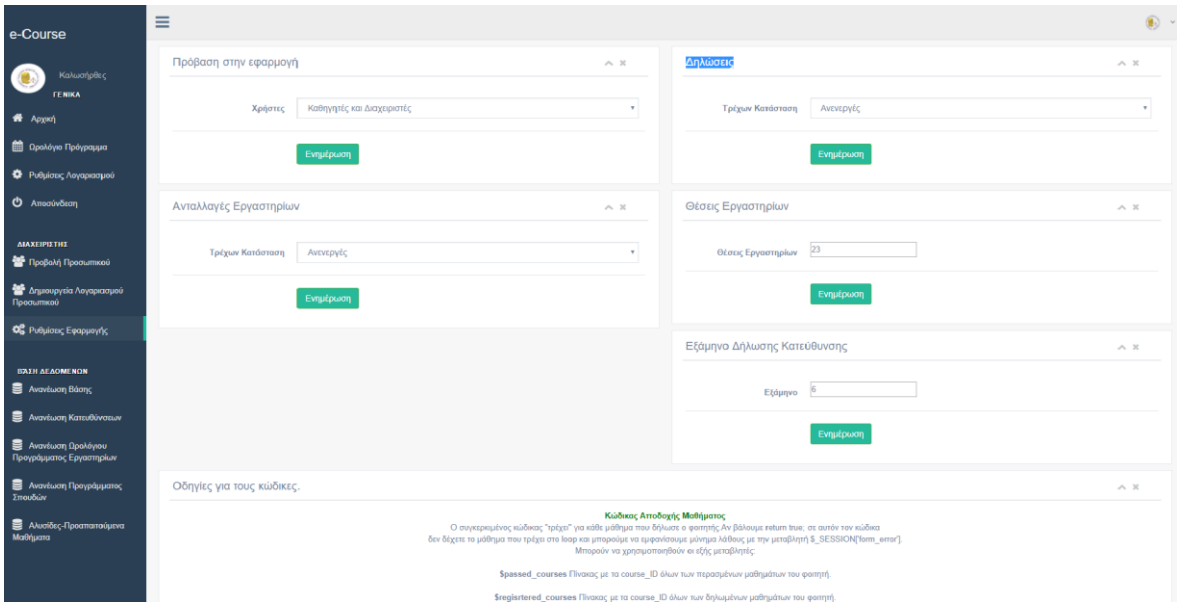
ΔΙΑΔΙΚΑΣΙΑ ΔΗΛΩΣΕΩΝ

ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο περιγράφεται αναλυτικά η διαδικασία των δηλώσεων καθώς και τα βήματα που πρέπει να ακολουθήσουν οι διαχειριστές, οι καθηγητές καθώς και οι φοιτητές του τμήματος ώστε να έχουμε μια επιτυχημένη διαδικασία. Στο πρώτο μέρος του κεφαλαίου αναλύεται η λειτουργία των διαχειριστών ως και πιο κρίσιμη για την περίοδο των δηλώσεων. Στη συνέχεια περιγράφεται η διαδικασία από την πλευρά των καθηγητών και των φοιτητών καθώς και επιλογές που έχουν όσον αναφορά τα εργαστήρια, τις λίστες μαθημάτων κλπ.

2.1 ΕΝΑΡΞΗ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΤΩΝ ΔΗΛΩΣΕΩΝ

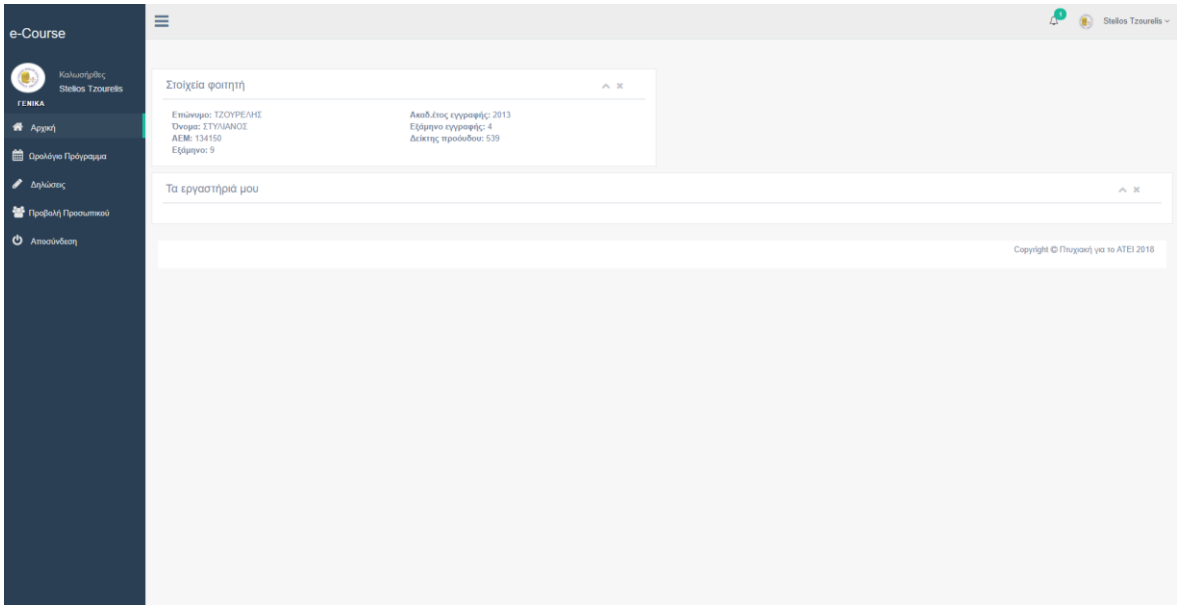
Οι δηλώσεις ξεκινούν από τους διαχειριστές αφού πρώτα έχουν ολοκληρωθεί όλες οι ενεργείς που περιγράφονται στο πρώτο κεφάλαιο .Ο διαχειριστής πρέπει να κατευθυνθεί στην σελίδα «Ρυθμίσεις Εφαρμογής». Βεβαιωνόμαστε ότι στην κατηγορία «Πρόσβαση στην εφαρμογή» έχουμε επιλεγμένο «Όλοι» έτσι ώστε οι φοιτητές να μπορούν να εισέλθουν στην εφαρμογή (Καλό είναι ενώ γίνονται εργασίες η ρυθμίσεις στην εφαρμογή , πρόσβαση να έχουν μόνο οι διαχειριστές και οι καθηγητές).Επιπλέον στην καρτέλα «Δηλώσεις» επιλέγουμε κατάσταση ενεργές. Μετά από αυτό οι φοιτητές μπορούν να ξεκινήσουν της δηλώσεις.



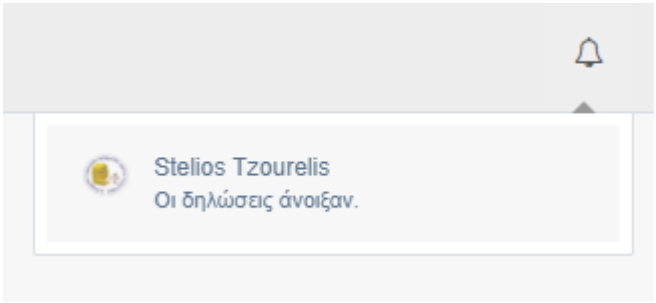
Σχήμα 5 " Σελίδα Ρυθμίσεις Εφαρμογής "

2.2 ΔΗΛΩΣΕΙΣ ΜΑΘΗΜΑΤΩΝ ΑΠΟ ΤΟΥΣ ΦΟΙΤΗΤΕΣ

Οι φοιτητές συνδέονται μέσα από το apps όπως σε όλες τις καινούριες εφαρμογές του ΑΤΕΙ. Μετά την είσοδο στην εφαρμογή, ο φοιτητής βλέπει την αρχική σελίδα όπου εμφανίζονται βασικά στοιχεία φοιτητή όπως εξάμηνο, ο δείκτης προόδου έτος εγγραφής κλπ. (Σχήμα 6).Στο επάνω μέρος υπάρχει το σύστημα υπενθυμίσεων όπου ο φοιτητής ενημερώνεται ότι οι διαδικασία των δηλώσεων έχει ξεκινήσει (Σχήμα 7).

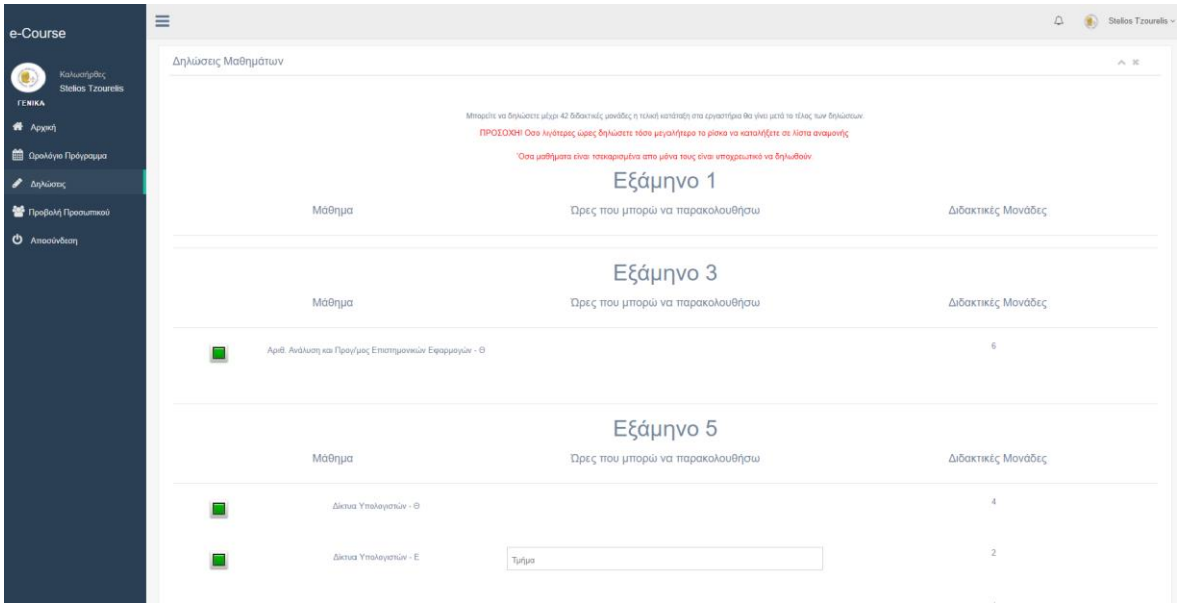


Σχήμα 6 " Αρχική Σελίδα Φοιτητή"



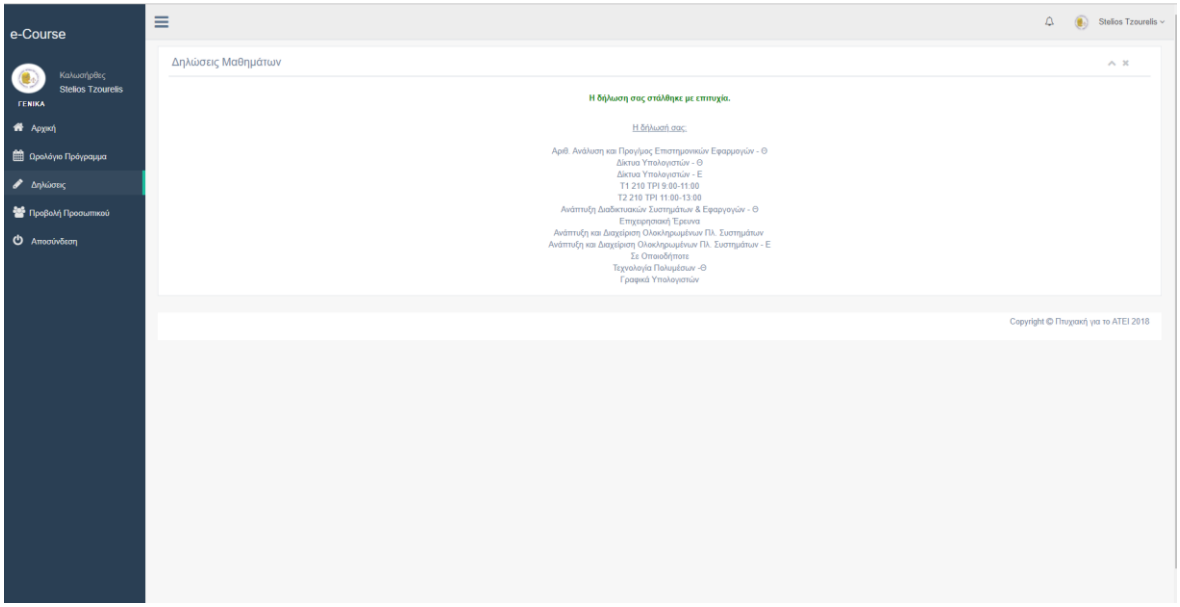
Σχήμα 7 " Σύστημα Υπενθυμίσεων"

Όταν οι διαχειριστές αλλάξουν την κατάσταση των δηλώσεων σε ενεργές στο menu των φοιτητών εμφανίζεται η επιλογή «Δηλώσεις» (Σχήμα 5). Επιλέγοντας την πηγαίνουν στην σελίδα των δηλώσεων (Σχήμα 8). Σε αυτή την σελίδα παρόμοια με το τρέχον σύστημα της πυθίας, εμφανίζονται τα εξάμηνα φοίτησης με τα μαθήματα που δεν έχει περάσει ο φοιτητής. Τα μαθήματα που έχουν και εργαστηριακό τμήμα έχουν την επιλογή του τμήματος. Ο φοιτητής μπορεί να επιλέξει πολλαπλά τμήματα που ενδιαφέρεται να καταταχθεί, αυτό όμως δεν σημαίνει και ότι η είσοδος του σε κάποιο από αυτά είναι δεδομένη (Η ανάλυση του αλγορίθμου κατάταξης αναλύετε σε παρακάτω κεφάλαιο). Μια επιπλέον επιλογή στα εργαστηριακά τμήματα είναι το "Σε Οποιοδήποτε Τμήμα”. Η συγκεκριμένη επιλογή σημαίνει ότι ο φοιτητής δεν έχει προτίμηση για το σε πιο τμήμα θα μπει είτε από αυτά που του εμφανίζονται σε επιλογή είτε από καινούρια που θα δημιουργηθούν. Σημειώνεται ότι φοιτητές που επιλέξουν "Σε Οποιοδήποτε Τμήμα" έχουν μικρότερες πιθανότητες να καταλήξουν σε λίστα αναμονής κάποιου εργαστηρίου.

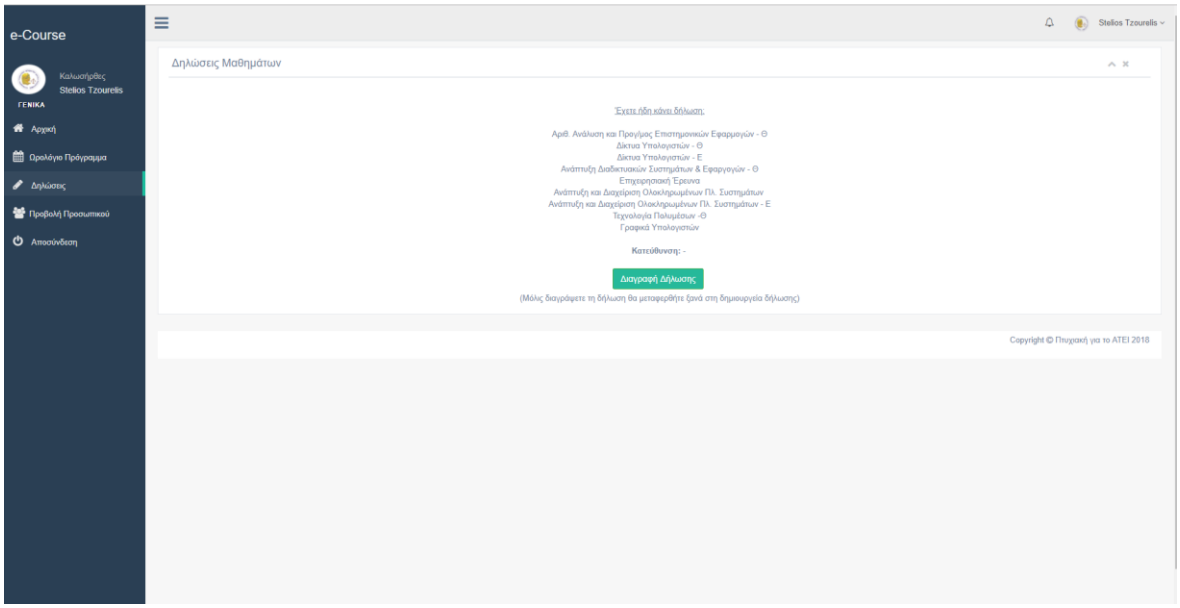


Σχήμα 8 " Σελίδα Δηλώσεων"

Αφού ο φοιτητής επιλέξει τα μαθήματα που τον ενδιαφέρουν και τα αντίστοιχα εργαστηριακά τμήματα που θέλει να παρακολουθήσει, επιλέγει την αποστολή δήλωσης. Το σύστημα ελέγχει τη δήλωση του (Αν όλα τα προαπαιτούμενα είναι δηλωμένα, αν έχει επιλέξει μαθήματα που δεν θα έπρεπε κλπ). Στην συνέχεια εμφανίζεται μια επιβεβαίωση με της επιλογές του χρήστη. Κάνοντας ανανέωση της σελίδας των δηλώσεων ο φοιτητής έχει την επιλογή να την διαγράψει και να την ξανακάνει από την αρχή σε περίπτωση που έκανε κάτι λάθος η έχει αλλάξει γνώμη (Σχήμα 9,10).



Σχήμα 9 " Επιβεβαίωση Δήλωσης"



Σχήμα 10 " Επιλογή Διαγραφής Δήλωσης"

ΕΠΙΛΟΓΟΣ

Σε αυτό το κεφάλαιο αναλύθηκε η διαδικασία εκκίνησης δηλώσεων από τους διαχειριστές καθώς και οι ενέργειες που ακολουθούν οι φοιτητές προκειμένου να αποστείλουν την δήλωση εξαμήνου στη γραμματεία. Στο επόμενο κεφάλαιο περιγράφεται με λεπτομέρεια ο τερματισμός και η ερμηνεία των αποτελεσμάτων των εγγραφών καθώς και τα εργαλεία που έχει το ΑΤΕΙ στα χέρια του προκειμένου να κάνει την διαδικασία αυτή ευκολότερη.

ΚΕΦΑΛΑΙΟ 3

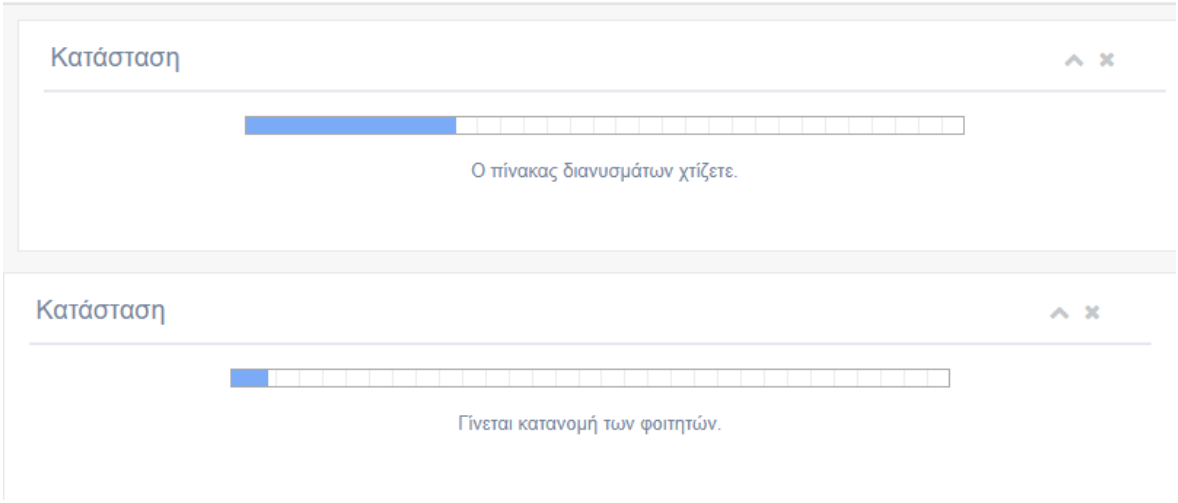
ΤΕΡΜΑΤΙΣΜΟΣ ΚΑΙ ΑΠΟΤΕΛΕΣΜΑΤΑ ΔΗΛΩΣΕΩΝ

ΕΙΣΑΓΩΓΗ

Σε αυτήν την ενότητα περιγράφεται η διαδικασία τερματισμού των δηλώσεων από τους διαχειριστές της πλατφόρμας. Είναι μια από τις σημαντικότερες και πιο πολύπλοκες διαδικασίες του e-course. Στη συνέχεια θα αναλυθούν τα αποτελέσματα αυτής της διαδικασίας από την πλευρά των φοιτητών και των καθηγητών.

3.1 ΤΕΡΜΑΤΙΣΜΟΣ ΔΗΛΩΣΕΩΝ ΑΠΟ ΤΟΥΣ ΔΙΑΧΕΙΡΙΣΤΕΣ

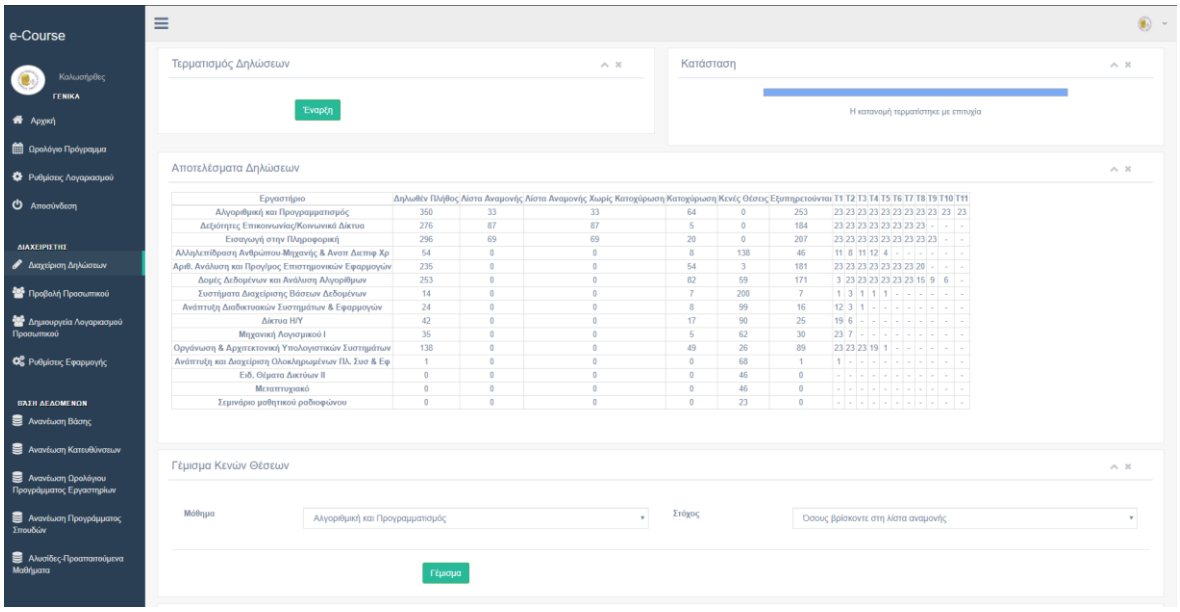
Η διαδικασία τερματισμού ξεκινάει από την σελίδα «Διαχείριση Δηλώσεων» του διαχειριστή. Πατώντας το κουμπί «Έναρξη» ο χρήστης ενημερώνεται για την κατάσταση των δηλώσεων (Σχήμα 11).Το βήμα αυτό μπορεί να πάρει αρκετά λεπτά για να ολοκληρωθεί ανάλογα με τον αριθμό των φοιτητών που ολοκλήρωσαν τις δηλώσεις τους.



Σχήμα 11 " Κατάσταση Τερματισμού"

Μετά την ολοκλήρωση αυτού του βήματος ο διαχειριστής οδηγείται σε μια σελίδα με προσωρινά αποτελέσματα δηλώσεων (κατανομή φοιτητών σε εργαστήρια) όπου του δίνεται η επιλογή διαγραφή τμήματος, προσθήκη καινούριου κλπ. (Σχήμα 12). Τα αποτελέσματα που εμφανίζονται δεν είναι αντιπροσωπευτικά πραγματικών δηλώσεων αλλά μια προσομοίωση για να δείξουμε της δυνατότητες της εφαρμογής. Το πρώτο table που βλέπουμε με τίτλο «Αποτελέσματα Δηλώσεων» είναι τα εργαστηριακά τμήματα που δήλωσαν οι φοιτητές , με την αντίστοιχη σειρά: το πρώτο πεδίο είναι πόσοι φοιτητές δήλωσαν το εργαστήριο, στο δεύτερο πεδίο καταγράφονται οι φοιτητές που κατέληξαν σε λίστα αναμονής (είτε γιατί δεν χωρούσαν στα εργαστήρια που είχαν δηλώσει κατά προτίμηση, είτε γιατί τα εργαστηριακά τμήματα αυτού του μαθήματος έχουν γεμίσει). Στο επόμενο πεδίο καταγράφεται ο αριθμός των φοιτητών που βρίσκεται σε λίστες αναμονής αλλά δεν έχουν κατοχυρωμένο το μάθημα (δηλαδή όσοι το δήλωσαν για πρώτη φορά η δεν συμπλήρωσαν τις απαραίτητες παρουσίες την προηγούμενη φορά που το παρακολούθησαν). Αμέσως μετά είναι ο αριθμός των φοιτητών που δήλωσαν το συγκεκριμένο εργαστήριο με κατοχύρωση (χωρίς να χρειάζεται δηλαδή να το παρακολουθήσουν) και στη συνέχεια ο αριθμός των κενών θέσεων, πόσους δηλαδή φοιτητές μπορούμε να εξυπηρετήσουμε ακόμα με τα διαθέσιμα εργαστήρια. Ο

τελευταίος αριθμός του table είναι ο αριθμός των φοιτητών που έχουν κατανεμηθεί σε εργαστήρια. Στην συνέχεια ακολουθείται η λίστα των εργαστηρίων με τον αριθμό των φοιτητών που έχει το καθένα.



Σχήμα 12 "Προσωρινά Αποτελέσματα Δηλώσεων"

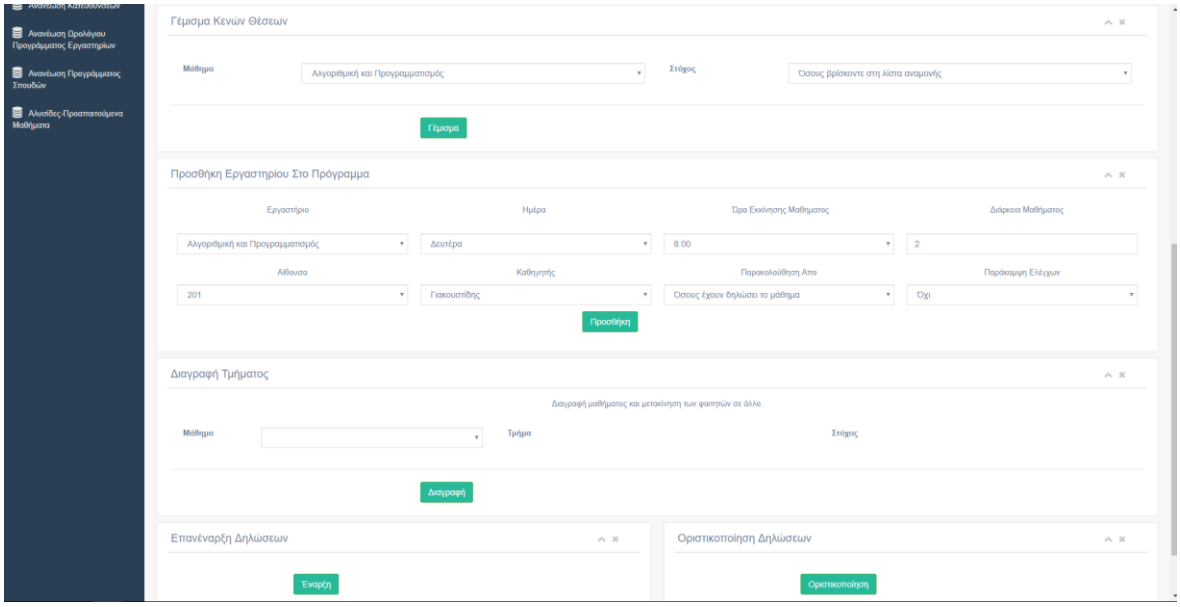
Κάτω από τα προσωρινά αποτελέσματα δηλώσεων εμφανίζονται 4 σειρές από καρτέλες με διάφορα εργαλεία και επιλογές για τον διαχειριστή της εφαρμογής όπως γέμισμα θέσεων, δημιουργία καινούριου τμήματος, διαγραφή τμήματος κλπ τα οποία θα αναλυθούν παρακάτω.

Η πρώτη καρτέλα είναι το Γέμισμα Κενών Θέσεων. Σε αυτήν δίνεται η επιλογή του διαχειριστή να "γεμίσει" τυχόν κενές θέσεις που υπάρχουν σε εργαστήρια τμήματα με συγκεκριμένους στόχους. Οι επιλογές που δίνονται είναι τα μαθήματα που έχουν εργαστήρια και σαν στόχοι υπάρχουν δύο επιλογές: λίστα αναμονής και λίστα αναμονής χωρίς κατοχύρωση. Επιλέγοντας λίστα αναμονής περνιούνται οι απαραίτητες γραμμές στη βάση ώστε να ανατεθούν οι φοιτητές από την λίστα αναμονής στα εργαστηριακά τμήματα που δεν είναι γεμάτα ενώ με την δεύτερη επιλογή, το σύστημα στοχεύει μόνο τους φοιτητές που δεν έχουν δηλώσει το συγκεκριμένο μάθημα στο παρελθόν και βρίσκονται στη λίστα αναμονής.

Στην επόμενη καρτέλα που ονομάζεται "Προσθήκη Εργαστηρίου Στο Πρόγραμμα" ο διαχειριστής έχει την επιλογή να προσθέσει καινούριο εργαστηριακό τμήμα στο ωρολογιακό πρόγραμμα και με το “γέμισμα” να στοχεύσει σε συγκεκριμένους φοιτητές. Στο καινούριο τμήμα αφού επιλεχθούν σωστά μάθημα , ημέρα, ώρα, αίθουσα και καθηγητής έχουμε την δυνατότητα παρακολούθησης όπου προσφέρονται οι δύο επιλογές που έχουμε αναφέρει και στο «Γέμισμα κενών θέσεων» όπως και οι δυο καινούριες. Το καινούριο εργαστήριο μπορεί να ανατεθεί σε όλους τους φοιτητές που έχουν δηλώσει το μάθημα και όσους έχουν επιλέξει «Οποιοδήποτε τμήμα». Μετά την προσθήκη του εργαστηρίου το σύστημα περνάει τις απαραίτητες γραμμές στην βάση και πρέπει να γίνει επανέναρξη της διαδικασίας.

Στην τελευταία καρτέλα (Διαγραφή Τμήματος) έχουμε την επιλογή να διαγράψουμε κάποιο εργαστηριακό τμήμα αφού επιλέξουμε το μάθημα που μας ενδιαφέρει και να μεταφέρουμε τους φοιτητές του σε κάποιο άλλο.

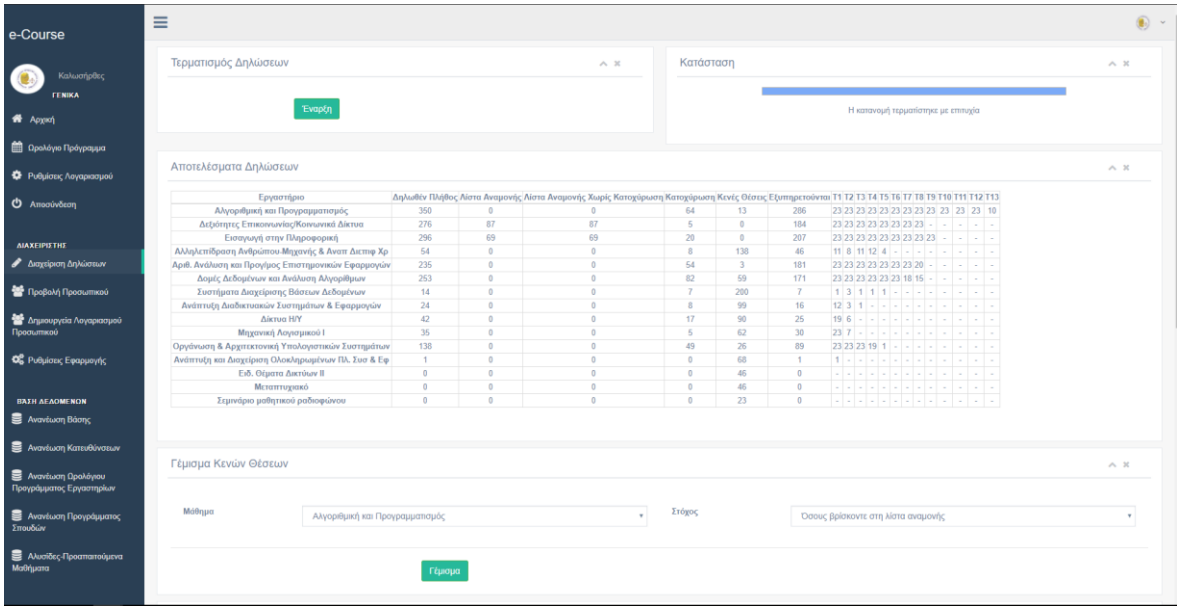
Όλα αυτά φαίνονται στο Σχήμα 13. Ένα παράδειγμα χρήσης θα ήταν το εξής. Βλέποντας ότι το μάθημα Αλγοριθμική και Προγραμματισμός έχει 33 άτομα σε λίστα αναμονής και καμιά διαθέσιμη θέση, θα δημιουργήσουμε 2 καινούρια τμήματα. Παρομοίως στο μάθημα Δομές Δεδομένων και Ανάλυση Αλγορίθμων παρατηρούμε ότι τα τμήματα Τ10 (με 6 μαθητές) και Τ1 (με 3 μαθητές) μπορούν να συγχωνευτούν με το Τ8 (9 μαθητές) και Τ7 (15 μαθητές) αντίστοιχα. Επομένως από την καρτέλα διαγραφή τμήματος κάνουμε τις απαραίτητες ενέργειες (βλέπε σχήμα 13).Αφού τελειώσουμε επιλέγουμε επανέναρξη δηλώσεων.



Σχήμα 13 " Εργαλεία Τερματισμού Δηλώσεων"

Όπως βλέπουμε και στην καινούρια σελίδα (Σχήμα 14) τα αποτελέσματα είναι αυτά που περιμέναμε. Έχουν δημιουργηθεί 2 καινούρια τμήματα στο μάθημα Αλγοριθμική και Προγραμματισμός και η λίστα αναμονής πλέον είναι άδεια, ενώ στο μάθημα Δομές Δεδομένων και Ανάλυση Αλγορίθμων τα 2 τμήματα που επιλέξαμε έχουν πλέον συγχωνευτεί και οι φοιτητές στο Τ7 και Τ8 είναι 18 και 15 αντίστοιχα. Η παραπάνω διαδικασία ακολουθείται μέχρι να είναι ευχαριστημένος ο διαχειριστής από τα αποτελέσματα των δηλώσεων. Όταν αυτό έχει γίνει ο χρήστης επιλέγει

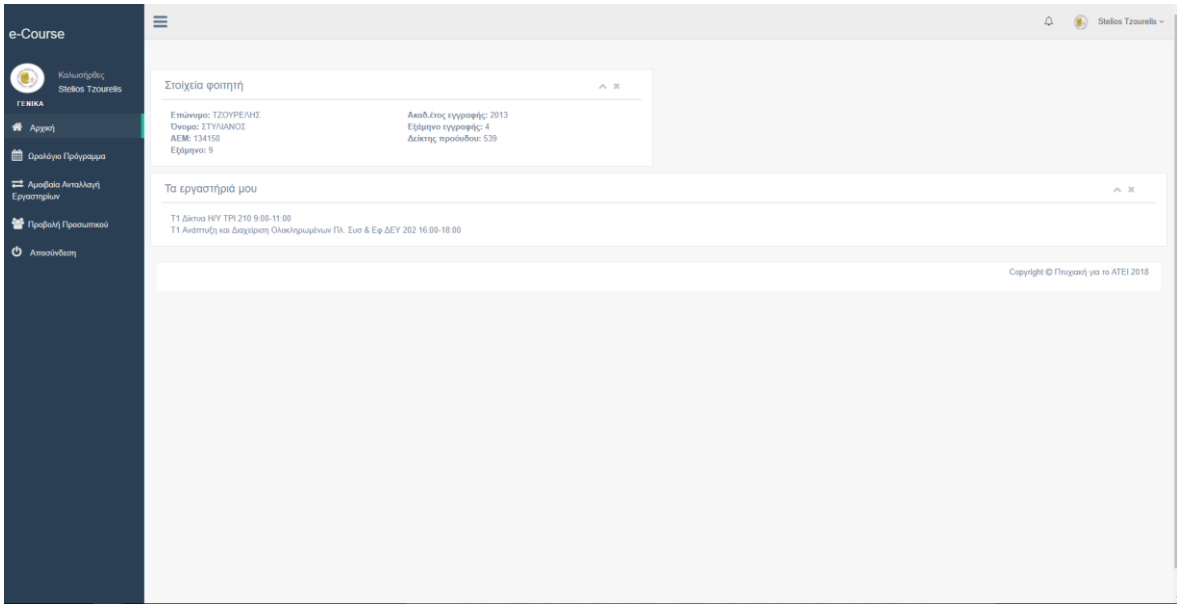
οριστικοποίηση των δηλώσεων



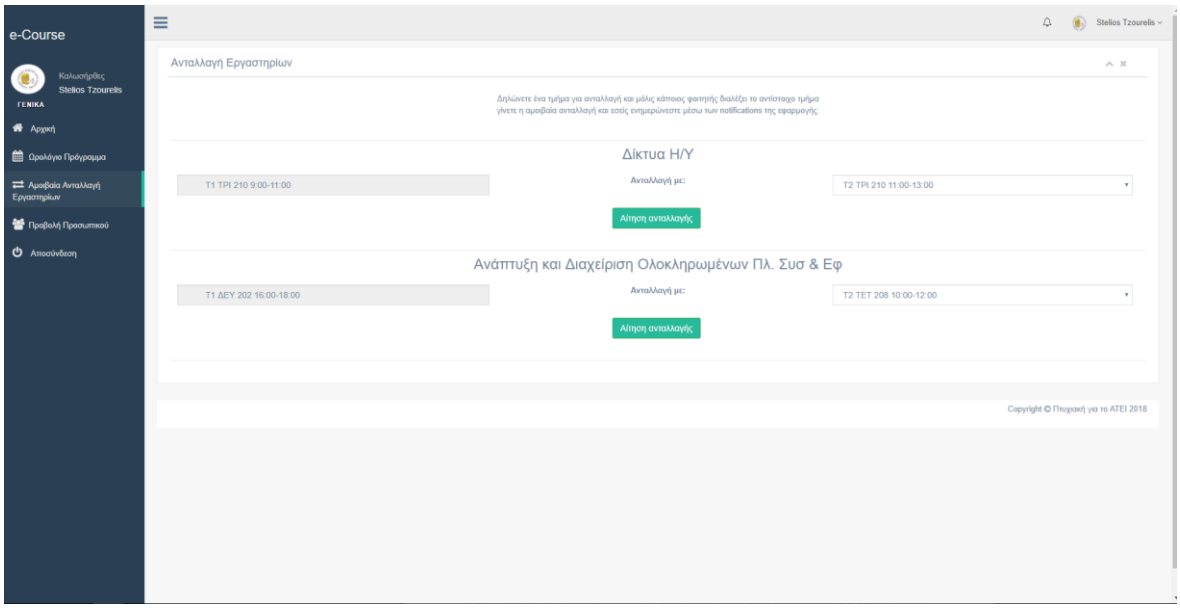
Σχήμα 14 " Καινούρια Αποτελέσματα Δηλώσεων"

3.2 ΑΠΟΤΕΛΕΣΜΑΤΑ ΔΗΛΩΣΕΩΝ ΣΤΟΥΣ ΦΟΙΤΗΤΕΣ

Μετά την επιτυχή οριστικοποίηση των δηλώσεων οι φοιτητές μπορούν να συνδεθούν στην εφαρμογή για να δουν σε ποια εργαστηριακά τμήματα έχουν καταταχθεί. Στην αρχική οθόνη φαίνονται όλα τα εργαστήρια και οι ώρες που είναι γραμμένος ο φοιτητής ενώ σε περίπτωση που η διοίκηση του τμήματος το επιλέξει ενεργοποιείται η επιλογή «Αμοιβαία Ανταλλαγή Εργαστηρίων».Σε αυτήν την σελίδα επιτρέπεται στον φοιτητή να δηλώσει ενδιαφέρον για ανταλλαγή της θέσης του σε κάποιο άλλο τμήμα. Όταν ένας φοιτητής δηλώσει ενδιαφέρον για την θέση του τότε πραγματοποιείται η αμοιβαία ανταλλαγή. Σημειώνεται ότι στις ανταλλαγές ισχύει σειρά προτεραιότητας, οι φοιτητές που δήλωσαν πρώτοι θα εξυπηρετηθούν νωρίτερα (Σχήμα 15,16).



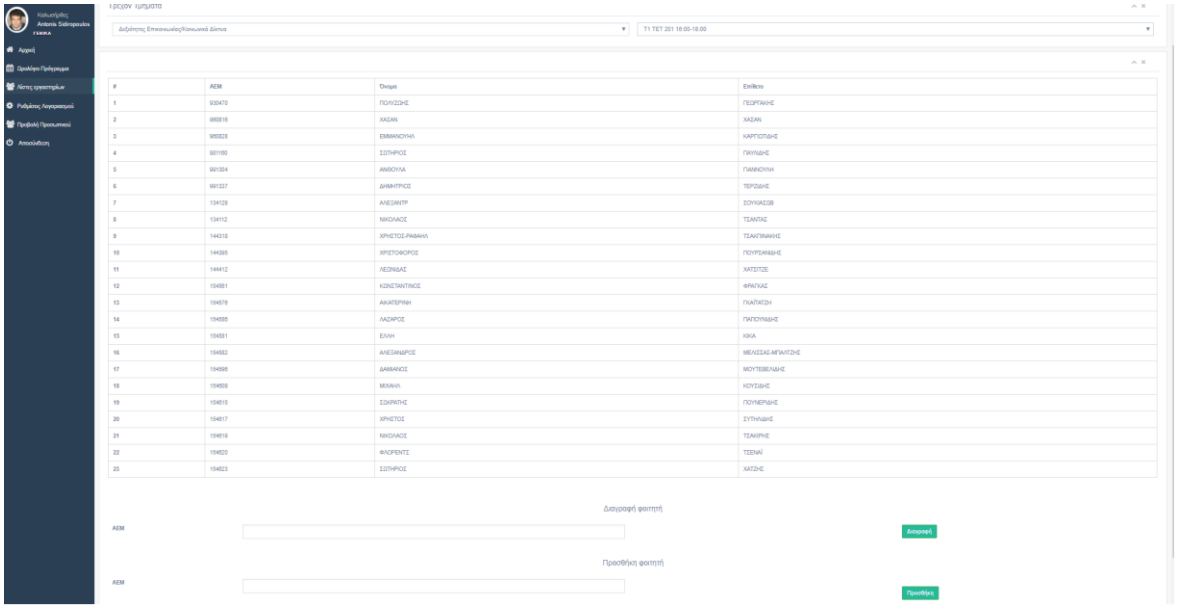
Σχήμα 15 " Αρχική Σελίδα Φοιτητή Μετά τις Δηλώσεις"



Σχήμα 16 " Σελίδα Αμοιβαίων Ανταλλαγών"

3.3 ΑΠΟΤΕΛΕΣΜΑΤΑ ΔΗΛΩΣΕΩΝ ΣΤΟΥΣ ΚΑΘΗΓΗΤΕΣ

Μετά το πέρας των δηλώσεων οι καθηγητές μπορούν να δουν τις λίστες εργαστηρίων (Όνομα επώνυμο και αριθμό ειδικού μητρώου των φοιτητών).Επίσης δίνεται η επιλογή στον καθηγητή να επέμβει άμεσα στην λίστα του εργαστηρίου προσθέτοντας και διαγράφοντας φοιτητές με τον αριθμό ειδικού μητρώου τους(Βλέπε σχήμα 17).Αξίζει να σημειωθεί ότι οι ενέργειες των καθηγητών καταγράφονται σε ειδικό table (logs) έτσι ώστε να μπορούν να εμφανιστούν σε περίπτωση που χρειαστούν.



Σχήμα 17 " Σελίδα Λίστα Εργαστηρίου"

ΕΠΙΛΟΓΟΣ

Παρατηρούμε λοιπόν ότι το e-course προσφέρει μια σειρά από εργαλεία για τους καθηγητές και τους διαχειριστές του ΑΤΕΙ κάνοντας την διαδικασία των δηλώσεων ευκολότερη ενώ παράλληλα δίνει την ευκαιρία στους φοιτητές να εγγραφούν σε λίστες εργαστηρίων απροβλημάτιστα. Επιπλέον μια λειτουργία που προσφέρεται είναι η αμοιβαία ανταλλαγή εργαστηρίων στους φοιτητές κάτι που κάθε εξάμηνο προκαλεί σύγχυση τόσο στους καθηγητές όσο και στους φοιτητές. Στο επόμενο κεφάλαιο θα αναλυθούν οι τεχνικές προγραμματισμού καθώς και οι τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη της πλατφόρμας.

ΚΕΦΑΛΑΙΟ 4

ΣΧΕΔΙΑΣΜΟΣ ΤΗΣ ΒΑΣΗΣ

ΕΙΣΑΓΩΓΗ

Ένα πολύ σημαντικό κομμάτι της κάθε διαδικτυακής εφαρμογής είναι η βάση δεδομένων. Ο σχεδιασμός της βάσης είναι μια χρονοβόρα και δύσκολη διαδικασία. Σε αυτό το κεφάλαιο αναλύεται ο σχεδιασμός της βάσης της πλατφόρμας καθώς και οι πίνακες που την απαρτίζουν.

4.1 ΑΝΑΛΥΣΗ ΤΩΝ ΠΙΝΑΚΩΝ ΤΗΣ ΒΑΣΗΣ

Η βάση του ecourse αποτελείται από 23 πίνακες, άλλοι πολύ σημαντικοί για την λειτουργεία της πλατφόρμας (π.χ. πίνακας students) και άλλοι συμπληρωματικοί (π.χ. πίνακας logs που καταγράφει τις κινήσεις των καθηγητών/διαχειριστών).Ο πρώτος πίνακας που θα αναλύσουμε είναι αυτός που αποθηκεύει τα στοιχεία των φοιτητών.

```
CREATE TABLE `students` (
  `student_ID` int(11) NOT NULL,
  `aem` varchar(35) COLLATE utf8_unicode_ci NOT NULL,
  `first` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `last` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `fname` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `in_year` int(4) NOT NULL,
  `in_period_ID` int(3) NOT NULL,
  `student_status` enum('active','deleted','graduated','on hold','transfered','declared
graduate','to be registered') COLLATE utf8_unicode_ci NOT NULL DEFAULT 'active',
  `deletion_reason` text COLLATE utf8_unicode_ci NOT NULL,
  `in_semester_ID` int(3) NOT NULL,
  `semester_ID` int(3) NOT NULL,
  `direction` int(2) NOT NULL,
  `progress_indicator` int(5) NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

ALTER TABLE `students`
  ADD PRIMARY KEY (`student_ID`),
  ADD KEY `aem` (`aem`);
COMMIT;
```

Το παραπάνω σχέδιο του πίνακα students μας δείχνει τα πεδία που τον απαρτίζουν καθώς και τα κλειδιά του. Τα πεδία περιλαμβάνουν το student_ID που είναι ένας εσωτερικός κωδικός μοναδικό για κάθε φοιτητή που χρησιμοποιείται αποκλειστικά από την εφαρμογή, τον αριθμό ειδικού μητρώου που χρησιμοποιείται κατά την είσοδο του φοιτητή στην εφαρμογή ώστε να κάνουμε την αντιστοίχηση με τον εσωτερικό κωδικό. Στην συνέχεια έχουμε τα προσκοπικά στοιχεία του φοιτητή καθώς και τον δείκτη προόδου και το εξάμηνο σπουδών. Για τον συγκεκριμένο πίνακα επιλέχθηκε σαν βασικό κλειδί το student_ID ενώ προστέθηκε ο α.ε.μ σαν ευρετήριο αφού γίνονται ερωτήματα στην βάση με αυτό το πεδίο.

Ο επόμενος πίνακας που θα αναλύσουμε είναι ο πίνακας ο registered_courses που περιέχει όλα τα δηλωμένα μαθήματα από όλους τους φοιτητές. Ο registered_courses περιλαμβάνει τον κωδικό του φοιτητή, τον A.E.M, το πρόγραμμα σπουδών και κωδικό μαθήματος, βαθμολογία καθώς και πότε αυτό δηλώθηκε. Ο πίνακας αυτός δεν έχει άλλο κλειδί παρά μόνο ένα ευρετήριο, τον κωδικό φοιτητή, καθώς είναι και το μόνο που χρησιμοποιείται για την αναζήτηση αποτελεσμάτων. Τέλος από το σχέδιο του βλέπουμε ότι ο κωδικός φοιτητή του πίνακα έχει συσχετιστεί με αυτόν του πίνακα students.

```
CREATE TABLE `registered_courses` (
  `student_ID` int(11) NOT NULL,
  `aem` varchar(35) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL,
  `curriculum_code` varchar(35) CHARACTER SET utf8 COLLATE utf8_unicode_ci
NOT NULL,
  `course_ID` varchar(10) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT
NULL,
  `course_name` varchar(250) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT
NULL,
  `year` int(4) NOT NULL,
  `period_ID` int(2) NOT NULL,
  `grade` float NOT NULL,
  `epshort` varchar(8) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `registered_courses`
  ADD KEY `student_ID` (`student_ID`);

ALTER TABLE `registered_courses`
  ADD CONSTRAINT `registered_courses_ibfk_1` FOREIGN KEY (`student_ID`)
REFERENCES `students` (`student_ID`);
COMMIT;
```

Ο πίνακας registered_course είναι πανομοιότυπος με αυτόν των περασμένων μαθημάτων (passed_courses), με την διαφορά ότι ο τελευταίος περιέχει μόνο μαθήματα που οι φοιτητές έχουν περάσει.

Στη συνέχεια θα αναλύσουμε έναν από τους πιο σημαντικούς πίνακες της βάσης με όνομα vectors. Ο πίνακας αυτός όπως αναφέρεται και στο επόμενο κεφάλαιο χρησιμοποιείται από τον αλγόριθμο κατανομής φοιτητών σε εργαστηριακά τμήματα. Οι εγγραφές του χτίζονται κάθε φορά κατά τον τερματισμό των δηλώσεων και στην συνέχεια γίνεται η κατανομή τους με βάση συγκεκριμένα πεδία.

```
CREATE TABLE `vectors` (
  `student_ID` int(11) NOT NULL,
  `course_ID` varchar(10) NOT NULL,
  `lab_ID` int(10) NOT NULL,
  `lab_day` int(2) NOT NULL,
  `lab_starting_hour` int(2) NOT NULL,
  `lab_priority`
enum('direction_obligatory','obligatory','direction_normal','normal','obligat
ory_direction_attended','obligatory_attended','direection_attended','attended
') NOT NULL DEFAULT 'normal',
  `progress_indicator` int(5) NOT NULL DEFAULT '0',
  `status` enum('unchecked','done','skip') NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `vectors`
  ADD PRIMARY KEY (`student_ID`,`lab_ID`),
  ADD KEY `lab_ID` (`lab_ID`);

ALTER TABLE `vectors`
  ADD CONSTRAINT `vectors_ibfk_1` FOREIGN KEY (`student_ID`) REFERENCES
`students` (`student_ID`),
  ADD CONSTRAINT `vectors_ibfk_2` FOREIGN KEY (`lab_ID`) REFERENCES
`schedule_labs` (`lab_ID`);
COMMIT;
```

Οι χρήσεις των πεδίων του πίνακα αναλύονται με λεπτομέρεια στο επόμενο κεφάλαιο όπου περιγράφουμε τον αλγόριθμο κατανομής. Τα κλειδιά που χρησιμοποιούμε είναι στο student_id μαζί με το lab_id (κάθε φοιτητής μπορεί να δηλώσει ένα εργαστήριο από μια φορά). Επίσης υπάρχουν οι συσχετισμοί των πρωτεύων κλειδιών με τους αντίστοιχους πίνακες. των δηλώσεων και στην συνέχεια γίνετε η κατανομή τους με βάση συγκεκριμένα πεδία.

Ο τελευταίος πίνακας που θα αναλύσουμε είναι ο schedule_labs.Ο συγκεκριμένος πίνακας περιλαμβάνει όλα τα εργαστηριακά τμήματα που είναι προγραμματισμένα για το εξάμηνο τις ώρες έναρξης, την μέρα καθώς και την αίθουσα. Ο λόγος που χρειαζόμαστε όλη αυτήν την πληροφορία είναι για να μπορούμε να ελέγχουμε αν ο φοιτητής που θέλει να κατανεμηθεί σε ένα εργαστηριακό τμήμα έχει ήδη εκείνη την ώρα και κάποιο άλλο μάθημα. Επίσης όταν ο διαχειριστής προσπαθεί να περάσει ένα καινούριο εργαστήριο πρώτα ελέγχεται αν η αίθουσα που έχει δηλώσει είναι κατειλημμένη από άλλο τμήμα.

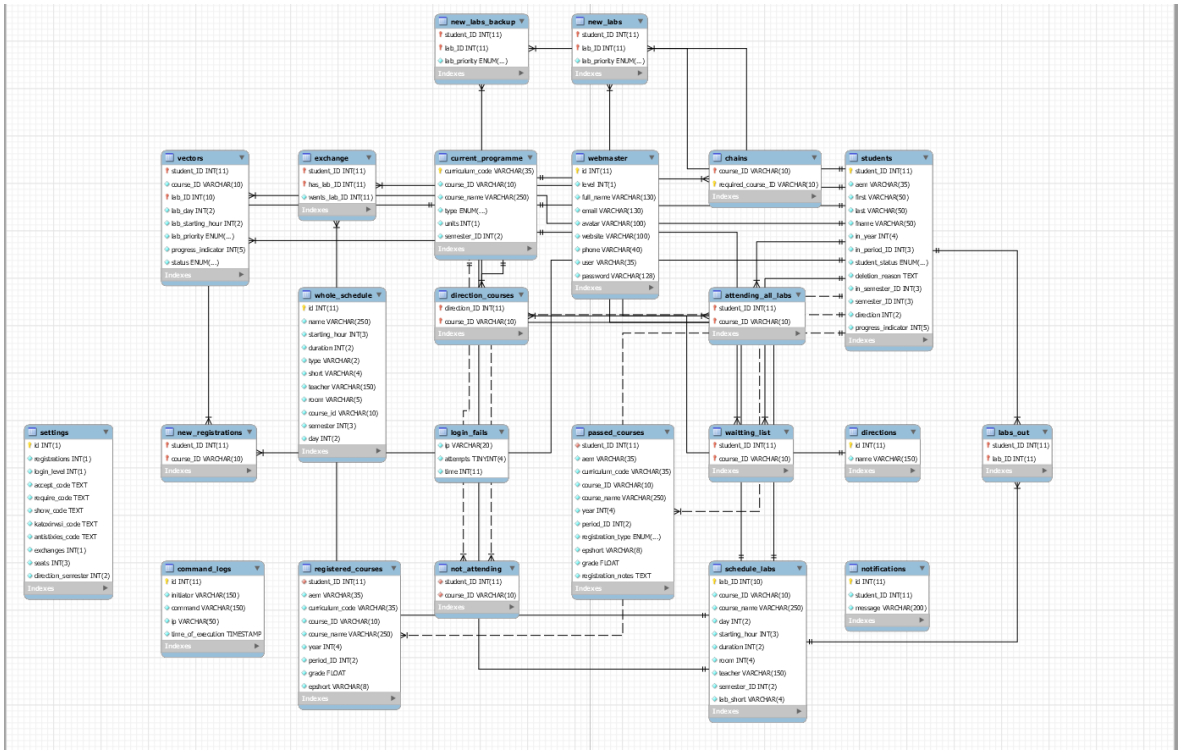
```
CREATE TABLE `schedule_labs` (
  `lab_ID` int(10) NOT NULL,
  `course_ID` varchar(10) CHARACTER SET utf8 NOT NULL,
  `course_name` varchar(250) CHARACTER SET utf8 NOT NULL,
  `day` int(2) NOT NULL,
  `starting_hour` int(3) NOT NULL,
  `duration` int(2) NOT NULL,
  `room` int(4) NOT NULL,
  `teacher` varchar(150) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL,
  `semester_ID` int(2) NOT NULL,
  `lab_short` varchar(4) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
--
ALTER TABLE `schedule_labs`
  ADD PRIMARY KEY (`lab_ID`);

ALTER TABLE `schedule_labs`
  MODIFY `lab_ID` int(10) NOT NULL AUTO_INCREMENT ;
```

Αξίζει να σημειωθεί ότι υπάρχει ο και πίνακας whole_schedule που περιλαμβάνει όλο το ωρολόγιο πρόγραμμα της σχολής και η δομή του είναι πανομοιότυπη με αυτή των εργαστηρίων.

4.2 ΣΧΕΣΙΑΚΟ ΜΟΝΤΕΛΟ ΒΑΣΗΣ

Το σχεσιακό μοντέλο μιας βάσης παρουσιάζει μια συλλογή από σχέσεις, μεταξύ των οντοτήτων (πινάκων) της. Όπως προαναφέρθηκε η βάση του ecourse περιλαμβάνει 23 πίνακες. Μερικοί από αυτούς δεν έχουν σχέσεις με άλλους όπως για παράδειγμα ο πίνακας settings που αποθηκεύει τις ρυθμίσεις της εφαρμογής. ‘Αλλοι πίνακες συσχετίζονται όπως προαναφέρθηκε στην ανάλυσή τους (students registered_courses,passed_courses).



Σχήμα 18 " Σχεσιακό Μοντέλο της Βάσης"

ΕΠΙΛΟΓΟΣ

Όπως παρατηρούμε η σχεδίαση και υλοποίηση μιας βάσης δεδομένων αν και εκ πρώτης όψεως είναι μια απλή διαδικασία, έχει μεγαλύτερο βάθος και πολυπλοκότητα καθώς ο σχεδιαστής πρέπει να δώσει μεγάλη βαρύτητα στα κλειδιά που θα χρησιμοποιήσει, στην δομή του κάθε πίνακα ξεχωριστά καθώς και στην μεταξύ τους σχέση. Στο επόμενο κεφάλαιο θα αναλυθεί τεχνικά η πλατφόρμα και τα επιμέρους συστήματα που την απαρτίζουν.

ΚΕΦΑΛΑΙΟ 5

ΤΕΧΝΙΚΗ ΑΝΑΛΥΣΗ ΤΗΣ ΠΛΑΤΦΟΡΜΑΣ

ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο θα αναλυθούν τεχνικά τα επιμέρους συστήματα της εφαρμογής όπως το σύστημα σύνδεσης, η βάση δεδομένων, οι διάφοροι αλγόριθμοι κατανομής των φοιτητών σε εργαστήρια κλπ. Για την κατασκευή του ecourse χρησιμοποιήθηκε η γλώσσα προγραμματισμού php σε έκδοση 7.0.Ο διακομιστής της βάσης δεδομένων είναι MariaDb σε έκδοση 10.1.31. Επίσης χρησιμοποιήθηκε ο apache στην έκδοση 2.4 για webserver.

ΑΝΑΛΥΣΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΣΥΝΔΕΣΗΣ 5.1

Το σύστημα χρηστών της πλατφόρμας είναι συνδεδεμένο με το apps του ΑΤΕΙ. Αυτό σημαίνει ότι οι φοιτητές χρησιμοποιούν τον ίδιο λογαριασμό για να συνδεθούν στο ecourse. Για προφανής λόγους δεν θα γίνει ανάλυση τις διασύνδεσης αλλά μια βασική περιγραφή του login από πλευράς χρηστών και στη συνέχεια διαχειριστών.

Το σύστημα σύνδεσης λειτουργεί κάνοντας χρήση του πίνακα $_SESSION της php. Αφού ο χρήστης συνδεθεί επιτυχώς στην εφαρμογή apps του ΑΤΕΙ από τα στοιχεία που παίρνουμε σαν επιστροφή το βασικότερο είναι ο αριθμός ειδικού μητρώου και η ομάδα χρηστών που ανήκει. Αν είναι φοιτητής κάνουμε μια ερώτηση στην βάση δεδομένο SELECT student_ID,first,last,semester_ID FROM students WHERE aem=? . Έτσι παίρνουμε τα βασικά στοιχεία του φοιτητή όπως εσωτερικό id , εξάμηνο ονοματεπώνυμο κλπ. Σημειώνεται ότι ο παραπάνω κώδικας τρέχει μόνο αν εξακριβωθεί ότι οι φοιτητές έχουν πρόσβαση στην εφαρμογή την δεδομένη στιγμή. Αυτό επιτυγχάνεται με την παρακάτω ερώτηση SELECT login_level FROM `settings` WHERE id='0'; . Ο δείκτης login_level μας δείχνει ποια ομάδα χρηστών έχει πρόσβαση στην εφαρμογή. Το 3 υποδηλώνει ότι όλοι μπορούν να συνδεθούν, το 2 ότι μόνο καθηγητές και διαχειριστές έχουν πρόσβαση στην εφαρμογή ενώ το 1 ότι μόνο οι διαχειριστές έχουν πρόσβαση στην εφαρμογή.

Με παρόμοιο τρόπο λειτουργεί και σύνδεση του διαχειριστή με κάποια επιπλέον βήματα κυρίως για λόγους ασφαλείας. Τα στοιχεία του διαχειριστή αποθηκεύονται σε διαφορετικό table από τους φοιτητές. Σε κάθε αποτυχημένη προσπάθεια σύνδεσης (είτε λάθος κωδικού είτε όνομα χρήστη) , αποθηκεύεται η διεύθυνση σύνδεσης του αιτώντα εισόδου (ip) και σε ένα table που ονομάζεται login_fails. Σε αυτό τον πίνακα αποθηκεύεται η ip , οι αποτυχημένες προσπάθειες του και ο χρόνος της τελευταίας προσπάθειας. Αν ο χρήστης κάνει πάνω από 3 αποτυχημένες προσπάθειες το σύστημα δεν τον αφήνει να ξανά προσπαθήσει για 5 λεπτά. Έπειτα από την επιτυχημένη προσπάθεια σύνδεσης η γραμμή που περιέχει την ip του χρήστη διαγράφεται.

ΑΝΑΛΥΣΗ ΤΗΣ ΕΙΣΑΓΩΓΗΣ ΔΕΔΟΜΕΝΩΝ 5.2

Η εισαγωγή δεδομένων στην πλατφόρμα γίνεται από τους διαχειριστές μια φορά το εξάμηνο πριν την διαδικασία των δηλώσεων. Η ενέργεια αυτή γίνεται από την σελίδα «Ανανέωση Βάσης». Ο χρήστης αφού επιλέξει τον τύπο του αρχείου αλλά και το αρχείο που βρίσκεται στον υπολογιστή του γίνεται το ανέβασμα του και στην συνέχεια η ανανέωση των δεδομένων. Ο διαχειριστής ενημερώνεται μέσω της τεχνολογίας ajax για την πορεία του ανεβάσματος.

Ο κώδικας που επισυνάπτεται παρακάτω είναι γραμμένος σε javascript και είναι υπεύθυνος για το ανέβασμα του αρχείου στον Server. Αφού επιλέξει το αρχείο του ο χρήστης δημιουργεί ένα καινούριο αντικείμενο XMLHttpRequest και περνάει συγκεκριμένα options. Η συγκεκριμένη γραμμή ajax.addEventListener("load", completeHandler, false); περνάει έναν handler για την ολοκλήρωση του ajax request, επόμενο η function με όνομα completeHandler θα τρέξει όταν δεχθούμε απάντηση από την server. Σε περίπτωση λάθους ή κώδικα "abort" από τον server έχουμε δηλώσει σαν callback το errorHandler και το abortHandler αντίστοιχα. Στις τελευταίες γραμμές βλέπουμε ότι το post της φόρμας γίνεται στην σελίδα updateDb.

```
function uploadFile()
{
    var file = document.getElementById("file").files[0];
    // alert(file.name+" | "+file.size+" | "+file.type);
    var formdata = new FormData();
    formdata.append("file", file);
    var e = document.getElementById("db_options");
    var db_options = e.options[e.selectedIndex].value;
    formdata.append("db_options",db_options);
    var ajax = new XMLHttpRequest();
    ajax.upload.addEventListener("progress", progressHandler, false);
    ajax.addEventListener("load", completeHandler, false);
    ajax.addEventListener("error", errorHandler, false);
    ajax.addEventListener("abort", abortHandler, false);
    ajax.open("POST", "updateDb.php");
    ajax.send(formdata);
    $(':button').prop('disabled', true);
}
```

Κώδικας 1 " Ανέβασμα αρχείων με AJAX"

Στη συνέχεια είναι ο κώδικας από πλευράς server που διαχειρίζεται το ανέβασμα του αρχείου και την αποθήκευση των στοιχείων του στη βάση. Πρώτα θα αναλύσουμε τον κώδικα που διαχειρίζεται το ανέβασμα του αρχείου των φοιτητών (παράρτημα Δ).

```php
<?php
if(isset($_FILES["file"]) && isset($_POST['db_options']) && is_uploaded_file(
$_FILES["file"]["tmp_name"]))
{
    set_time_limit(0);
    $enc = mb_detect_encoding($_FILES["file"]["tmp_name"], mb_list_encodings(), true);
    //if(mb_detect_encoding($str) == 'ISO-8859-7')
        //  $str = iconv("ISO-8859-7","UTF-8", $str);
    if ($str = file_get_contents($_FILES["file"]["tmp_name"]))
    {
        if(is_utf8($str) || $str = iconv("ISO-8859-7","UTF-8", $str))
            switch($_POST['db_options'])
            {
                //students
                case 1:
                $str = str_replace("student_ID  spec_aem    first   last    fname
in_year in_period_ID    cond_ID dgr_logos   in_exam_ID  exam_ID","",$str);
                $str = explode("\n",$str);
                $query = "INSERT INTO students
(`student_ID`,`aem`,`first`,`last`,`fname`,`in_year`,`in_period_ID`,`student_status`,`
deletion_reason`,`in_semester_ID`,`semester_ID`,`direction`,`progress_indicator`)
VALUES ";
                $index=0;
                $conn->query("DELETE * FROM `students`");
                $conn->query("ALTER TABLE `students` DROP INDEX aemIndex;");
                foreach($str as $tempstr)
                {
                    $tempstr = str_replace(",","",$tempstr);
                    $tempstr = str_replace("\t",",",$tempstr);
                    $tempstr = str_replace(",,",", ",$tempstr);
                    $tempstr = str_replace(",","','",$tempstr);
                    $tempstr = "('".$tempstr."','-1','0'),";
                    $arrayFields = explode(",",$tempstr);

...

                    if($index >1500)
                    {
                        $query = trim($query, ",");
                        $query = $query.";";
                        $conn->query($query);
                        $query = "INSERT INTO students
(`student_ID`,`aem`,`first`,`last`,`fname`,`in_year`,`in_period_ID`,`student_status`,`
deletion_reason`,`in_semester_ID`,`semester_ID`,`direction`,`progress_indicator`)
VALUES ";
                        $index=0;
                    }
                    $index++;
```

Κώδικας 3 " Ανέβασμα αρχείων, Server Side (Παράρτημα Δ.)"

Αφού πρώτα πάρουμε το περιεχόμενο του αρχείου που μας δόθηκε σαν είσοδος, το πρώτο πράγμα που κάνουμε είναι να διαγράψουμε την επικεφαλίδα αφού δεν μας χρειάζεται. Αφού αδειάσουμε τον πίνακα students (καθώς θα περάσουμε καινούριες εγγραφές μέσα) ξεκινάμε να χτίζουμε το query. Σπάμε κάθε γραμμή του κειμένου όπου συναντάμε "," για να πάρουμε τα απαραίτητα πεδία (ονοματεπώνυμο, Α.Ε.Μ , έτος εισόδου κλπ). Κάθε 1500 γραμμές κάνουμε παύση της δημιουργίας του ερωτήματος και τα περνάμε στην βάση. Αυτό το κάνουμε για να αποφύγουμε να αλλαχτεί η ρύθμιση max_allowed_packet στον διακομιστή της βάσης δεδομένων. Στη συνέχεια περνάμε και τις υπόλοιπες γραμμές ώσπου να

τελειώσει το κείμενο. Ο επόμενος κώδικας που θα αναλύσουμε είναι τον αποθηκευμένων μαθημάτων (ο κώδικας δεν διαφέρει πολύ από αυτόν των περασμένων μαθημάτων οπότε θα αναλυθούν μαζί, Παράρτημα Ε).

Όπως και με τον κώδικα των μαθητών το πρώτο πράγμα που κάνουμε είναι να αφαιρέσουμε τον τίτλο του περιεχομένου. Στη συνέχεια ξεκινάμε να χτίζουμε το ερώτημα της βάσης όπως και πριν. Επίσης σπάμε τα ερωτήματα κάθε 1500 γραμμές προκειμένου να αποφύγουμε timeouts από τον διακομιστή της βάσης δεδομένων. Αφού όλες οι γραμμές περαστούν και πριν ενημερώσουμε τον χρήστη για την επιτυχή έκβαση της ενέργειας του συναντάμε αυτή τη γραμμή "eval($code[αντιστοιχίες_code']);".Αφού τα δεδομένα περαστούν η πλατφόρμα τρέχει ένα επιπλέον κώδικα για τις αντιστοιχίες τον μαθημάτων από παλαιότερα σε καινούρια προγράμματα σπουδών (Η χρήση του κώδικα αυτού έχει αναλυθεί στο κεφάλαιο 1.3 σελ. 13). Ο κώδικας αυτός είναι σχεδόν ίδιος με τα περασμένα μαθήματα.

ΑΝΑΛΥΣΗ ΤΗΣ ΔΗΛΩΣΗΣ ΜΑΘΗΜΑΤΩΝ 5.3

Η δήλωση των μαθημάτων γίνεται από τους φοιτητές μια φορά το εξάμηνο. Διαχειρίζεται από το αρχείο registrations.php το οποίο φορτώνει τα απαραίτητα στοιχεία του μαθητή όπως: περασμένα μαθήματα ανα εξάμηνο, κατεύθυνση κλπ. Στη συνέχεια του εμφανίζει τα μαθήματα που μπορεί να δηλώσει καθώς και τα εργαστηριακά τμήματα στα οποία μπορεί να δηλώσει ως προτιμήσεις.

```php
<?php
session_start();
require("config/db_config.php");
require("misc/util.php");
if(!isset($_SESSION['user']) || !registrationsActive($conn) || $_SESSION['level'] !=
3)
{
    header ( "location:index.php" );
    return;
}

...

$query = $conn->prepare("SELECT course_ID FROM `passed_courses` WHERE student_ID=?;");
$query->bind_param('d',$_SESSION['user']);
$query->execute();
$passed_result=$query->get_result();
$passed_courses = fetchWholeArr($passed_result);
$query = $conn->prepare("SELECT course_ID,year FROM `registered_courses` WHERE
student_ID=?;");
$query->bind_param('d',$_SESSION['user']);
$query->execute();
$registered_result=$query->get_result();
$registered_courses = fetchWholeArr($registered_result);
$query = $conn->prepare("SELECT * FROM `students` WHERE student_ID=?;");
$query->bind_param('d',$_SESSION['user']);
$query->execute();
$student_result = $query->get_result();
$student = $student_result->fetch_assoc();

...

if(isset($_POST['requested']) && !empty($_POST['requested']))
    {
        //init
        $registrationSucc=true;
        //Build required array
        $hoursFromRequired=0;
        $max = $_SESSION['semester']+1;
        $required_exams=array();
```

Κώδικας 3 " Σελίδα Δηλώσεων (Παράρτημα Β)"

Ο κώδικας ξεκινάει καλώντας το helper function session start της php. Στη συνέχεια ελέγχει αν ο χρήστης είναι συνδεδεμένος, αν είναι φοιτητής (level = 3) καθώς και αν είναι οι δηλώσεις ενεργές τη στιγμή που ανοίγει η σελίδα. Συνεχίζοντας ελέγχεται αν το άνοιγμα της σελίδας έγινε κανονικά η αν έχει γίνει post σε αυτήν από την χρήστη, δηλαδή αν έχει στείλει δεδομένα.

Σε περίπτωση αποστολής δεδομένων ελέγχεται αν ο χρήστης στέλνει την δήλωση του ή αν την διαγράφει καθώς και οι 2 περιπτώσεις διαχειρίζονται από την ίδια σελίδα. Σε περίπτωση διαγραφής το πρόγραμμα διαγράφει από τα απαραίτητα table της βάσης δεδομένων τα στοιχεία του χρήστη ώστε να μπορέσει να ξανακάνει την δήλωση του από την αρχή. Αν ο χρήστης δεν διαγράφει την δήλωση του αλλά την στέλνει, ελέγχονται οι μεταβλητές $_POST['requested'] όπου είναι ένας πίνακας που περιλαμβάνει όλα τα μαθήματα που έχει δηλώσει ο φοιτητής. Πρώτα ελέγχονται από τον αλγόριθμο αν όλες οι αλυσίδες για το συγκεκριμένο μάθημα είναι περασμένες από τον μαθητή (σε περίπτωση που έχει). Στη συνέχεια ελέγχεται αν

έχουν ξεπεραστεί οι 42 διδακτικές μονάδες που επιτρέπεται να δηλωθούν ανά εξάμηνο, αν όχι τρέχει ο κώδικας αποδοχής μαθημάτων που είναι ρυθμισμένος από τον διαχειριστή του συστήματος (Η χρήση του κώδικα αυτού έχει αναλυθεί στο κεφάλαιο 1.3 σελ. 13).

ΑΝΑΛΥΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ ΚΑΤΑΝΟΜΗΣ 5.4

Ο αλγόριθμος κατανομής είναι υπεύθυνος για την ανάθεση των φοιτητών σε εργαστηριακά τμήματα. Στην πλατφόρμα υπάρχουν 3 αλγόριθμοι κατανομής, ο maxmimum flow, απλή κατανομή με βάση τον δείκτη προόδου και ο αλγόριθμος που δημιουργήθηκε με βάση ενός template που γράφτηκε από τον κύριο Σιδηρόπουλο. Από αυτούς τους τρεις σαν τελικός επιλέχθηκε ο τρίτος. Ακολουθεί μια περιγραφή αυτών των αλγορίθμων.

Ο αλγόριθμος στο παράρτημα Α είναι αυτός που επιλέχτηκε για την τελική κατάταξη των φοιτητών σε εργαστήρια και είναι ο εξής.

```
while(true)
{
    if(time()-$lastUpdate > 3)
    {
        statusUpdate($conn);
        $lastUpdate=time();
    }
    $vector=$conn->query("SELECT * FROM vectors v WHERE v.status='unchecked' ORDER BY
v.lab_priority ASC,v.progress_indicator DESC,v.course_ID ASC LIMIT 1;");
    if($vector->num_rows == 0)
        break;

    $vector=$vector->fetch_assoc();


if(canRegister($vector['lab_ID'],$vector['student_ID'],$vector['lab_day'],$vector['lab
_starting_hour'],$max_seats,$conn))

doRegister($vector['student_ID'],$vector['lab_ID'],$vector['course_ID'],$conn);
    else
    {
        markAsSkip($vector['student_ID'],$vector['lab_ID'],$conn);
        //Check if he is registered in another lab with the same hour and day

if(hourAndDayUsed($vector['student_ID'],$vector['lab_day'],$vector['lab_starting_hour'
],$conn))
            continue;

        //Try to free a seat
        if(freeASeat($vector['course_ID'],$vector['lab_ID'],$conn,$max_seats))

doRegister($vector['student_ID'],$vector['lab_ID'],$vector['course_ID'],$conn);
    }
}
```

Κώδικας 2 " Ανέβασμα αρχείων με AJAX (Παράρτημα Α)"

Αρχικά δημιουργούνται διανύσματα στο πίνακα vectors όπου περιλαμβάνονται το id του μαθητή, το μάθημα το id του εργαστηρίου, την ώρα και την μέρα που ξεκινάει το εργαστήριο, το priority, τον δείκτη προόδου του φοιτητή

καθώς και το status του διανύσματος. Το priority παίρνει τις εξής τιμές-ανάλογα με την κατάσταση του φοιτητή, αν το μάθημα το έχει παρακολουθήσει στο παρελθόν (παρατηρητής) το status ορίζεται ως υποχρεωτικό κατεύθυνσης (αν είναι εργαστήριο κατεύθυνσης) που έχει παρακολουθήσει, υποχρεωτικό που έχει παρακολουθήσει, κατεύθυνσης που έχει παρακολουθήσει, και απλός παρατηρητής. Αν το μάθημα δεν έχει ξανά δηλωθεί από τον φοιτητή τότε παίρνει τις εξής τιμές: υποχρεωτικό κατεύθυνσης, υποχρεωτικό κανονικό κατεύθυνσης και κανονικό. Το status του διανύσματος παίρνει μια από τις εξής τιμές: είτε δεν έχει ελεγχθεί (unchecked),είτε περασμένο (skip), είτε έτοιμο (done). Στην τελευταία περίπτωση ο φοιτητής είναι κατοχυρωμένος στο εργαστήριο που εμφανίζεται στο διάνυσμα. Αφού δημιουργηθεί ο πίνακας των διανυσμάτων, τότε επιλέγεται μια γραμμή την φορά με lab_priority αύξουσα , φθίνουσα δείκτη προόδου και αύξουσα κωδικό μαθήματος. Αν ο μαθητής μπορεί να γραφτεί σε αυτό το εργαστήριο (δηλαδή το εργαστήριο δεν είναι γεμάτο) και ο φοιτητής δεν έχει άλλο εργαστήριο την ίδια ώρα και μέρα, τότε τον καταχωρούμε σε αυτό και μαρκάρουμε το διάνυσμα ως done. Αν αυτό δεν είναι εφικτό τότε μαρκάρουμε το διάνυσμα ως skip. Στη συνέχεια ελέγχουμε αν μπορούμε να αδειάσουμε κάποια θέση για τον φοιτητή αν δηλαδή μπορεί κάποιος από τους εγγεγραμμένους να περαστεί σε άλλο εργαστήριο. Αν ναι κάνουμε την μετακίνηση και γράφουμε τον φοιτητή που μαρκάραμε σαν "skip" στο εργαστήριο και αλλάζουμε το status του διανύσματος σε done. Η παραπάνω διαδικασία εκτελείται μέχρις ότου να μην υπάρχουν διανύσματα που να μην έχουν ελεγχθεί από τον αλγόριθμο.

Ο επόμενος αλγόριθμος που υπάρχει στην πλατφόρμα (σε ξεχωριστό πρόγραμμα) ονομάζεται Maximum flow του Edmond Karps. Ο αλγόριθμος αυτός ψάχνει την μέγιστη ροή από την πηγή στον στόχο. Στη συγκεκριμένη περίπτωση πηγή θεωρούνται οι μαθητές που πρέπει να καταχωρηθούν στα εργαστήρια. Το κάθε μάθημα-εργαστήριο έχει 23 θέσεις οπότε ο κάθε στόχος του μαθήματος έχει από 23 συνδέσεις. Ο κώδικας αυτός έχει ως σκοπό την μέγιστη απόδοση-εξυπηρέτηση φοιτητών ανάλογα με τις δηλώσεις τους και τις διαθέσιμες θέσεις των εργαστηριακών τμημάτων. Ο αλγόριθμος είναι υλοποιημένος σε java και λόγο τις πολυπλοκότητας του εγκαταλείφθηκε σαν λύση , υπάρχουν όμως λειτουργικά πρωτότυπα και η επιλογή για εξαγωγή/εισαγωγή των αποτελεσμάτων στην πλατφόρμα σε περίπτωση που κάποιος φοιτητής θέλει να τον ολοκληρώσει.

Τέλος ο πιο απλός αλγόριθμος από τους τρεις διαθέσιμους, είναι η κατανομή με βάση των δείκτη προόδου. Η ίδια διαδικασία ακολουθείται ως προς τον τερματισμό των δηλώσεων όσον αναφορά το μέρος του διαχειριστή. Ο αλγόριθμος αυτός κατατάσσει τους φοιτητές σε εργαστήρια που έχουν επιλέξει δίνοντας προτεραιότητα στους μαθητές με τον μεγαλύτερο δείκτη προόδου. Αν και εύκολος στην υλοποίηση του καθώς και "λογική" λύση εκ πρώτης όψεως, ο αλγόριθμος αυτός δεν λαμβάνει υπόψιν το εξάμηνο φοίτησης, τα μαθήματα βαρύτητας, την κατεύθυνση όπως επίσης και τις κατοχυρώσεις των σπουδαστών με αποτέλεσμα να τον καθιστά μη επαρκή για τους στόχους του ΑΤΕΙ.

ΕΠΙΛΟΓΟΣ

Σε αυτό το κεφάλαιο αναλύθηκε ο πηγαίος κώδικας της εφαρμογής καθώς και οι τεχνικές που χρησιμοποιήθηκαν για την ανάπτυξη του. Επίσης δόθηκε βαρύτητα στον τρόπο κατανομής των φοιτητών σε εργαστηριακά τμήματα κάτι που όχι μόνο αποτελεί μεγάλο μέρος της εφαρμογής αλλά είναι και πολύ σημαντικό για την λειτουργία του ΑΤΕΙ.

ΣΥΜΠΕΡΑΣΜΑΤΑ

Από όλα τα παραπάνω συμπεραίνουμε ότι μια λύση που βασίζεται στις νέες τεχνολογίες για την διαδικασία των δηλώσεων δεν είναι αδύνατη. Το e-course ή μια οποιαδήποτε πλατφόρμα παρόμοιου σκοπού θα βοηθούσε στην αντικατάσταση της Πυθίας για την συγκεκριμένη εργασία ενώ παράλληλα θα σταματούσε την ταλαιπωρία τόσο για τους φοιτητές όσο και το διδακτικό προσωπικό του ΑΤΕΙ. Βελτιώσεις που θα μπορούσαν να γίνουν στην πλατφόρμα είναι καθάρισμα του κώδικα ώστε να είναι πιο ευανάγνωστος και μετονομασία συγκεκριμένων μεταβλητών ώστε να αντιπροσωπεύουν καλύτερα τον σκοπό τους. Επίσης όπως αναφέρθηκε και νωρίτερα για τον αλγόριθμο maximum flow, αν και το λειτουργικό πρωτότυπο μπορεί να φέρει αποτέλεσμα υπάρχουν edge cases που δεν έχει ελεγχθεί η αποτελεσματικότητα του. Επίσης, ο αλγόριθμος του Edmond Karp είναι αρκετά πολύπλοκος και θα μπορούσε να αποτελέσει αυτοτελή πτυχιακή εργασία ενός ενδιαφερόμενου φοιτητή.

ΑΝΑΦΟΡΕΣ

ΒΙΒΛΙΟΓΡΑΦΙΑ

Luke Welling, Laura Thomson, PHP and My SQL Web Development 4th Edition.

Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems, 3rd Edition.

ΠΑΡΑΡΤΗΜΑΤΑ

Παράρτημα Α.

```
<?php
//Dimiourgia Dianismatwn
$result = $conn->query("SELECT * FROM new_labs;");
while($row=$result->fetch_assoc()){
    if(time()-$lastUpdate > 3)  {
        statusUpdate($conn);
        $lastUpdate=time();
    }
    //Lab db
    $query = $conn->prepare("SELECT course_ID,day,starting_hour FROM
`schedule_labs` WHERE lab_ID=?;");
    $query->bind_param('d',$row['lab_ID']);
    $query->execute();
    $lab_result = $query->get_result();
    $lab = $lab_result->fetch_assoc();
    //Semester db
    $query = $conn->prepare("SELECT semester_ID FROM `current_programme`
WHERE course_ID=?;");
    $query->bind_param('s',$lab['course_ID']);
    $query->execute();
    $class_res = $query->get_result();
    $class_res = $class_res->fetch_assoc();
    $semester=$class_res['semester_ID'];
    //Student db
    $query = $conn->prepare("SELECT progress_indicator,semester_ID FROM
`students` WHERE student_ID=?;");
    $query->bind_param('d',$row['student_ID']);
    $query->execute();
    $student_result = $query->get_result();
    $student = $student_result->fetch_assoc();

    $status='unchecked';

    //Insert stin db
    $query = $conn->prepare("INSERT INTO `vectors`
(student_ID,course_ID,lab_ID,lab_day,lab_starting_hour,lab_priority,progress_
indicator,status)
                            VALUES (?,?,?,?,?,?,?,?);");
    $query-
>bind_param('dsdddsds',$row['student_ID'],$lab['course_ID'],$row['lab_ID'],$l
ab['day'],

$lab['starting_hour'],$row['lab_priority'],$student['progress_indicator'],$st
atus);
    $query->execute();
```

```php
<?php
    //Sort Dianismatwn
    $seats=$conn->query("SELECT seats FROM `settings` WHERE id=0;");
    $seats=$seats->fetch assoc();
    $max_seats=$seats['seats'];

    while(true)
    {
        if(time()-$lastUpdate > 3)
        {
            statusUpdate($conn);
            $lastUpdate=time();
        }
        $vector=$conn->query("SELECT * FROM vectors v WHERE v.status='unchecked' ORDER BY
v.lab_priority ASC,v.progress_indicator DESC,v.course_ID ASC LIMIT 1;");
        if($vector->num_rows == 0)
            break;

        $vector=$vector->fetch_assoc();


if(canRegister($vector['lab_ID'],$vector['student_ID'],$vector['lab_day'],$vector['lab_star
ting_hour'],$max_seats,$conn))
            doRegister($vector['student_ID'],$vector['lab_ID'],$vector['course_ID'],$conn);
        else
        {
            markAsSkip($vector['student_ID'],$vector['lab_ID'],$conn);
            //Check if he is registered in another lab with the same hour and day

if(hourAndDayUsed($vector['student_ID'],$vector['lab_day'],$vector['lab_starting_hour'],$co
nn))
                continue;

            //Try to free a seat
            if(freeASeat($vector['course_ID'],$vector['lab_ID'],$conn,$max_seats))

doRegister($vector['student_ID'],$vector['lab_ID'],$vector['course_ID'],$conn);
        }
    }
    $conn->query("INSERT INTO `labs_out` SELECT v.student_ID,v.lab_ID FROM `vectors` v
WHERE v.status='done';");
    $last=$conn->query("SELECT DISTINCT(student_ID) FROM new_labs;");
    while($student = $last->fetch_assoc())
    {
        $query = $conn->prepare("SELECT DISTINCT(s.course_ID) FROM new_labs n,schedule_labs
s WHERE n.student_ID=? AND n.lab_ID=s.lab_ID;");
        $query->bind_param('d',$student['student_ID']);
        $query->execute();
        $registered = fetchWholeArr($query->get_result());
        $query = $conn->prepare("SELECT DISTINCT(s.course_ID) FROM labs_out o,schedule_labs
s WHERE o.student_ID=? AND o.lab_ID=s.lab_ID;");
        $query->bind_param('d',$student['student_ID']);
        $query->execute();
        $done = fetchWholeArr($query->get_result());
        $waitting = array_diff($registered,$done);
        foreach($waitting as $course)
        {
            $query = $conn->prepare("INSERT INTO waitting_list VALUES (?,?);");
            $query->bind_param('ds',$student['student_ID'],$course);
            $query->execute();
        }
    }
```

Παράρτημα Β

```
session_start();
require("config/db config.php");
require("misc/util.php");
if(!isset($_SESSION['user']) || !registrationsActive($conn) || $_SESSION['level'] != 3)
{
    header ( "location:index.php" );
    return;
}
$hasRegistered = hasRegistered();
$hasDirection = false;
if($hasRegistered && isset($_POST['cmd']) && strcmp($_POST['cmd'],'delete') == 0)
{
    $query = $conn->prepare("DELETE FROM new_registrations WHERE student_ID=? ;");
    $query->bind param('d',$ SESSION['user']);
    $query->execute();
    $query = $conn->prepare("DELETE FROM attending all labs WHERE student ID=? ;");
    $query->bind_param('d',$_SESSION['user']);
    $query->execute();
    $query = $conn->prepare("DELETE FROM new_labs WHERE student_ID=? ;");
    $query->bind param('d',$ SESSION['user']);
    $query->execute();
    $query = $conn->prepare("DELETE FROM not attending WHERE student ID=? ;");
    $query->bind_param('d',$_SESSION['user']);
    $query->execute();

    $query = $conn->prepare("SELECT semester_ID FROM `students` WHERE student_ID=?;");
    $query->bind_param('d',$_SESSION['user']);
    $query->execute();
    $student = $query->get_result();

    $query = $conn->prepare("SELECT semester_ID FROM `students` WHERE student_ID=?;");
    $query->bind_param('d',$_SESSION['user']);
    $query->execute();
    $student_result = $query->get_result();
    $student = $student_result->fetch_assoc();

    $result = $conn->query("SELECT direction_semester FROM `settings` WHERE id='0';");
    $settingSem = $result->fetch_assoc();

    if($settingSem['direction_semester'] == $student['semester_ID'])
    {
        $defaultDir=-1;
        $query = $conn->prepare("UPDATE `students` SET direction=? WHERE student_ID=?;");
        $query->bind_param('dd',$defaultDir,$_SESSION['user']);
        $query->execute();
    }

    $hasRegistered=false;
}
if(!$hasRegistered)
{
    //Load db
    $code_result=$conn->query("SELECT
direction_semester,show_code,require_code,accept_code,katoxirwsi_code FROM `settings` WHERE
id='0';");
    $code=$code_result->fetch_assoc();
    $query = $conn->prepare("SELECT course_ID FROM `passed_courses` WHERE student_ID=?;");
    $query->bind param('d',$ SESSION['user']);
    $query->execute();
    $passed_result=$query->get_result();
    $passed_courses = fetchWholeArr($passed_result);
    $query = $conn->prepare("SELECT course_ID,year FROM `registered_courses` WHERE
student_ID=?;");
    $query->bind_param('d',$_SESSION['user']);
```

```
$query->execute();
    $registered_result=$query->get_result();
    $registered_courses = fetchWholeArr($registered_result);
    $query = $conn->prepare("SELECT * FROM `students` WHERE student_ID=?;");
    $query->bind_param('d',$_SESSION['user']);
    $query->execute();
    $student_result = $query->get_result();
    $student = $student_result->fetch_assoc();
    if($student['direction'] != -1 && !isInvalid($student['direction']))
    {
        $hasDirection = true;
        $_GET['dir'] = $student['direction'];
    }
    //An blepei dilwsi
    if(!isset($_GET['dir']) || isInvalid($_GET['dir']))
        $_GET['dir']=1;
    //An stelnei dilwsi
    if(!$hasDirection && isset($_POST['dir']) && !isInvalid($_POST['dir']))
        $_GET['dir']=$_POST['dir'];
    $query = $conn->prepare("SELECT course_ID FROM direction_courses WHERE
direction_ID=?;");
    $query->bind_param('d',$_GET['dir']);
    $query->execute();
    $direction_courses= fetchWholeArr($query->get_result());
    //
    if(isset($_POST['requested']) && !empty($_POST['requested']))
    {
        //init
        $registrationSucc=true;
        //Build required array
        $hoursFromRequired=0;
        $max = $_SESSION['semester']+1;
        $required_exams=array();
        if($max >9)
            $max=9;
        $winter = !(date('m')>6);
        for($semester=1; $semester<$max; $semester++)
        {
            if(($winter && $semester%2==0) || (!$winter && $semester%2 !=0))
                continue;

            $query="SELECT * FROM `current_programme` WHERE semester_ID='".$semester."'";
            $result=$conn->query($query);
            while($row = $result->fetch_assoc())
            {
                if(in_array($row['course_ID'],$passed_courses))
                    continue;

                $returned_show = eval($code['show_code']);
                if(isset($returned_show) && $returned_show)
                    continue;

                $returned_required = eval($code['require_code']);
                //eval to required
                if((isset($returned_required) && $returned_required && $semester <
$code['direction_semester']) ||
                ($semester >= $code['direction_semester']) && $returned_required &&
!(empty($direction_courses))
                && in_array($row['course_ID'],$direction_courses)))
                    array_push($required_exams,$row['course_ID']);
                $hoursFromRequired+=$row['units'];
            }
        }
```

```
//Verify that all the required are there
        $result = array_diff($required_exams, $_POST['requested']);
        if(!empty($result))
        {
            $_SESSION['form_error']="Δεν δηλώθηκαν όλα τα απαιτούμενα μαθήματα.";
            $registrationSucc=false;
        }
        //
        $sumUnits=0;
        foreach($_POST['requested'] as $study)
        {
            $query = $conn->prepare("SELECT units,course_name FROM current_programme WHERE
course_ID=?;");
            $query->bind_param('s',$study);
            $query->execute();
            $result = $query->get_result();
            $row =  $result->fetch_assoc();
            $accept_return = eval($code['accept_code']);
            //Load Chains
            $query = $conn->prepare("SELECT required_course_ID FROM chains WHERE course_ID=?;");
            $query->bind_param('s',$study);
            $query->execute();
            $chains_result = $query->get_result();
            $chains = $chains_result->fetch_assoc();
            //
            //Verify that all the chains are there
            if(!empty($chains))
                foreach($chains as $required_course)
                {
                    if(!in_array($required_course,$passed_courses))
                    {
                        $registrationSucc=false;
                        $query = $conn->prepare("SELECT course_name FROM current_programme WHERE
course_ID=?;");
                        $query->bind_param('s',$required_course);
                        $query->execute();
                        $result_name = $query->get_result();
                        $required_name = $result_name->fetch_assoc();
                        $_SESSION['form_error']="Για να δηλώσετε το μάθημα
".$row['course_name']." πρέπει να έχετε περάσει το μάθημα ".$required_name['course_name'].".";
                        $registrationSucc=false;
                        break;
                    }

                }
            //
            if(isset($accept_return) && $accept_return)
            {
                $registrationSucc=false;
                break;
            }
            $sumUnits+=$row['units'];
        }
        if($sumUnits >42)
        {
            $_SESSION['form_error']="'Εχετε επιλέξει πάνω απο 42 διδακτικές μονάδες.Η δήλωση δεν
στάλθηκε.";
            $registrationSucc=false;
        }
        if($registrationSucc)
        {
            foreach($_POST['requested'] as $study)
            {
                $query = $conn->prepare("INSERT INTO new_registrations VALUES (?,?);");
                $query->bind_param('ds',$_SESSION['user'],$study);
```

```
$query->execute();
                if(isset($_POST[$study]))
                {
                    if(in_array('KAT',$_POST[$study]) &&
in_array($study,$registered_courses))
                    {
                        $query = $conn->prepare("INSERT INTO not_attending VALUES (?,?);");
                        $query->bind_param('ds',$_SESSION['user'],$study);
                        $query->execute();
                    }
                    else if(in_array('ΟΠΟ',$_POST[$study]))
                    {
                        $query="SELECT lab_ID FROM `schedule_labs` WHERE
course_ID='".$study."'";
                        $result=$conn->query($query);
                        $labIds = fetchWholeArr($result);
                        foreach($labIds as $lab_ID)
                        {
                            //if(!in_array($limiter,$luck))
                                //continue;
                            //Calculate priority
                            $priority='';
                            //set priority and status
                            if(in_array($study,$registered_courses))
                            {
                                if(in_array($study,$direction_courses)
                                && in_array($study,$required_exams))
                                    $priority='direction_obligatory_attended';
                                else if(!in_array($study,$direction_courses)
                                && in_array($study,$required_exams))
                                    $priority='obligatory_attended';
                                else if(!in_array($study,$direction_courses)
                                && in_array($study,$required_exams))
                                    $priority='direction_attended';
                                else
                                    $priority='attended';
                            }
                            else
                            {
                                if(in_array($study,$direction_courses)
                                && in_array($study,$required_exams))
                                    $priority='direction_obligatory';
                                else if(!in_array($study,$direction_courses)
                                    && in_array($study,$required_exams))
                                    $priority='obligatory';
                                else if(in_array($study,$direction_courses)
                                    && !in_array($study,$required_exams))
                                    $priority='direction_normal';
                                else
                                    $priority='normal';
                            }

                            $query = $conn->prepare("INSERT INTO new_labs VALUES
(?,?,?);");
                            $query->bind_param('dss',$_SESSION['user'],$lab_ID,$priority);
                            $query->execute();
                        }
                        $query = $conn->prepare("INSERT INTO attending_all_labs VALUES
(?,?);");
                        $query->bind_param('ds',$_SESSION['user'],$study);
                        $query->execute();
                    }
                    else
                    foreach($_POST[$study] as $lab_ID)
```

```
{
                        if(strcmp($lab_ID,'KAT') == 0)
                            continue;

                        //Calculate priority
                        $priority='';
                        //set priority and status
                        if(in_array($study,$registered_courses))
                        {
                            if(in_array($study,$direction_courses)
                            && in_array($study,$required_exams))
                                $priority='direction_obligatory_attended';
                            else if(!in_array($study,$direction_courses)
                            && in_array($study,$required_exams))
                                $priority='obligatory_attended';
                            else if(!in_array($study,$direction_courses)
                            && in_array($study,$required_exams))
                                $priority='direction_attended';
                            else
                                $priority='attended';
                        }
                        else
                        {
                            if(in_array($study,$direction_courses)
                            && in_array($study,$required_exams))
                                $priority='direction_obligatory';
                            else if(!in_array($study,$direction_courses)
                                && in_array($study,$required_exams))
                                $priority='obligatory';
                            else if(in_array($study,$direction_courses)
                                && !in_array($study,$required_exams))
                                $priority='direction_normal';
                            else
                                $priority='normal';
                        }

                        $query = $conn->prepare("INSERT INTO new_labs VALUES
(?,?,?);");
                        $query-
>bind_param('dss',$_SESSION['user'],$lab_ID,$priority);
                        $query->execute();
                    }
                }
            }
            $_SESSION['form_succ']="Η δήλωση σας στάλθηκε με επιτυχία.";
        }
    }
}
```

Παράρτημα Δ.

```
case 1:
    $str = str_replace("student_ID  spec_aem    first   last    fname   in_year
in_period_ID    cond_ID dgr_logos   in_exam_ID  exam_ID","",$str);
    $str = explode("\n",$str);
    $query = "INSERT INTO students
(`student_ID`,`aem`,`first`,`last`,`fname`,`in_year`,`in_period_ID`,`student_status`,`
deletion_reason`,`in_semester_ID`,`semester_ID`,`direction`,`progress_indicator`)
VALUES ";
    $index=0;
    $conn->query("DELETE * FROM `students`");
    $conn->query("ALTER TABLE `students` DROP INDEX aemIndex;");
    foreach($str as $tempstr)
    {
        $tempstr = str_replace(",","",$tempstr);
        $tempstr = str_replace("\t",",",$tempstr);
        $tempstr = str_replace(",,",",",$tempstr);
        $tempstr = str_replace(",","','",$tempstr);
        $tempstr = "('".$tempstr."','-1','0'),";
        $arrayFields = explode(",",$tempstr);

        //giaytous pou exoun etos eisodou keno
        if(isset($arrayFields[5]) && !is_numeric(str_replace("'","",$arrayFields[5])))
            $arrayFields[5]="'-1'";

        //giaytous pou exoun etos eisodou keno
        if(isset($arrayFields[6]) && !is_numeric(str_replace("'","",$arrayFields[6])))
            $arrayFields[6]="'-1'";

        //giaytous pou exoun etos eisodou keno
        if(isset($arrayFields[9]) && empty(str_replace("'","",$arrayFields[9])))
            $arrayFields[9]="'-1'";

        $arrayFields=implode(",",$arrayFields);

        if(substr_count($arrayFields,",")>3)
            $query = $query.$arrayFields;

        //spasimo gia na min xriastei na alaksoume to max_allowed_packet
        if($index >1500)
        {
            $query = trim($query, ",");
            $query = $query.";";
            $conn->query($query);
            $query = "INSERT INTO students
(`student_ID`,`aem`,`first`,`last`,`fname`,`in_year`,`in_period_ID`,`student_status`,`
deletion_reason`,`in_semester_ID`,`semester_ID`,`direction`,`progress_indicator`)
VALUES ";
            $index=0;
        }
        $index++;
    }
    $query = trim($query, ",");
    $query = $query.";";
    $conn->query($query);
    $conn->query("ALTER TABLE `students` ADD INDEX aemIndex (`aem`);");
    echo "To table students αννανεώθηκε.";
    break;
```

Παράρτημα Ε.

```
case 2:
    $str = str_replace("student_ID  spec_aem    courseID    coursecode  title
cyear   cperiod cregtypeID  epshort cgrade  notes","",$str);
    $str = explode("\n",$str);
    $query = "INSERT INTO passed courses
(`student_ID`,`aem`,`curriculum_code`,`course_ID`,`course_name`,`year`,`perio
d_ID`,`registration_type`,`epshort`,`grade`,`registration_notes`) VALUES ";
    $index=0;
    foreach($str as $tempstr)
    {
        $tempstr = str_replace(",",".",$tempstr);
        $tempstr = str_replace("\t",",",$tempstr);
        $tempstr = str_replace(",,",", ",$tempstr);
        $tempstr = str_replace(",","','",$tempstr);
        $tempstr = "('".$tempstr."'),";
        if(substr_count($tempstr,",")>3)
            $query = $query.$tempstr;

        //spasimo gia na min xriastei na alaksoume to max_allowed_packet
        if($index >1500)
        {
            $query = trim($query, ",");
            $query = $query.";";
            $conn->query($query);
            $query = "INSERT INTO passed_courses
(`student_ID`,`aem`,`curriculum_code`,`course_ID`,`course_name`,`year`,`perio
d_ID`,`registration_type`,`epshort`,`grade`,`registration_notes`) VALUES ";
            $index=0;
        }

        $index++;
    }
    $query = trim($query, ",");
    $query = $query.";";
    $conn->query($query);
    $query = "UPDATE passed_courses SET
registration_type=registration_type+1;";
    $conn->query($query);

    $code_result=$conn->query("SELECT antistixies_code FROM `settings` WHERE
id='0';");
    $code=$code_result->fetch_assoc();
    eval($code['antistixies_code']);

    $conn->query($query);
    echo "To table student's courses αννανεώθηκε.";
    break;
```