

ΤΕΙ ΑΘΗΝΑΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.
ΔΙΚΤΥΑΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Εργασία εξαμήνου (Θ+Ε):

Στοιχεία Ομάδας:

- Αξιώτης Θεόφιλος – 131011
- Τσίκο Γιούλι – 131027

Τμήμα: Παρασκευή 12-2

Ως παραδοτέα στον φάκελο περιέχονται τα ακόλουθα:

- Script δημιουργίας της βάσης καθώς και το σχήμα
- Τα αρχεία προς εγκατάσταση στον server
- Τα αρχεία της desktop εφαρμογής
- Ο πηγαίος κώδικας όλων των εφαρμογών

Γενικά Στοιχεία Εργασίας:

Στα πλαίσια του μαθήματος Δικτυακός Προγραμματισμός κληθήκαμε , να δημιουργήσουμε ένα μικρό μέρος ενός φοιτητολογίου για ένα τμήμα ΑΕΙ. Η λειτουργία του συστήματος αυτού ακολουθεί τις βασικές κατευθυντήριες γραμμές που δόθηκαν στις διαλέξεις , τόσο του εργαστηριακού μέρους του μαθήματος , όσο και του θεωρητικού. Πιο συγκεκριμένα η εφαρμογή που αναπτύξαμε , είναι προσβάσιμη για τους φοιτητές του τμήματος , μόνο μέσω της αντίστοιχης διαδικτυακής εφαρμογής και οι δυνατότητες του περιορίζονται μόνο στην εμφάνιση των βαθμών τους αλλά και την επιλογή μαθημάτων. Οι φοιτητές **δεν** έχουν δυνατότητα εισόδου στην desktop εφαρμογή. Στην προαναφερθείσα εφαρμογή έχουν δυνατότητα εισόδου μόνο οι διαχειριστές και γραμματείς της σχολής. Αναλόγως με τα εκάστωτε δικαιώματα του χρήστη , για τα οποία δίνεται η δυνατότητα δυναμικής επεξεργασίας, εμφανίζονται οι αντίστοιχες λειτουργίες στο γραφικό περιβάλλον της εφαρμογής. Αμφότερες οι δύο εφαρμογές (διαδικτυακή και μη) για την λειτουργία τους κάνουν χρήση των υλοποιημένων web services.

Πιο αναλυτικά , η εφαρμογή μας έχει :

- Δημιουργηθεί σε βάση δεδομένων mySQL και η διαχείριση αυτής γίνεται μέσω του εργαλείου phpmyadmin
- Σχεδιαστεί με τέτοιο τρόπο ώστε να υποστηρίζεται η multi – tier φύση της.
- Σχεδιαστεί και αναλυθεί ώστε να υποστηρίζει το role based access.
- Κάνει χρήση , για την καθ'ολοκλήρου λειτουργία της, των δημιουργημένων απο εμάς web services.
- Κάνει χρήση του προτύπου JSON για τη μεταφορά των δεδομένων.

Οι βασικές λειτουργίες της εφαρμογής αναπαρίστανται ως εξής:

- Εισαγωγή ενός νέου φοιτητή (Α.Μ., Επώνυμο, Όνομα, Εξάμηνο)
- Εισαγωγή/τροποποίηση του βαθμού ενός φοιτητή σε ένα μάθημα.
- Τροποποίηση των βαθμών πολλών φοιτητών σε ένα μάθημα (στοιχεία από αρχείο “.xlsx”).
- Προβολή των προσωπικών στοιχείων ενός φοιτητή και των βαθμών του.
- Εισαγωγή των στοιχείων ενός νέου χρήστη του συστήματος.

Τέλος , το σύστημα αναγνωρίζει τριών ειδών κατηγορίες (ρόλους) χρηστών:

- **Διαχειριστή**, ο οποίος έχει τη δυνατότητα να εκτελεί όλες τις λειτουργίες του συστήματος.
- **Γραμματέα**, ο οποίος εκτελεί όλες τις λειτουργίες του συστήματος εκτός της λειτουργίας εισαγωγής νέου χρήστη.
- **Φοιτητή**, ο οποίος μπορεί να δει μόνο τα στοιχεία τα οποία τον αφορούν.

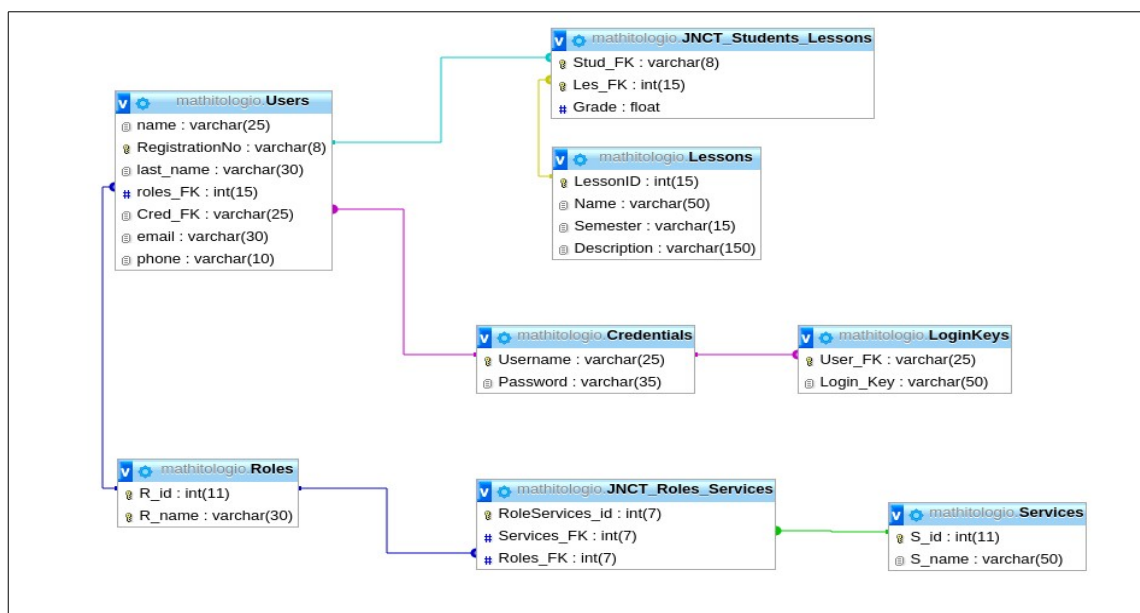
1. Βάση Δεδομένων:

Για να υποστηριχθεί και να λειτουργήσει το role based σύστημα , κρίνεται απαραίτητο να δημιουργηθεί κατάλληλη βάση δεδομένων που θα περιέχει αφενώς τους τύπους χρηστών που μπορούν να χρησιμοποιήσουν την εφαρμογή , αφετέρου τις επιτρεπόμενες υπηρεσίες /ενέργειες που εκτελούνται στο σύστημα , αλλά φυσικά και την αλληλεξάρτηση μεταξύ αυτών σε έναν ειδικά διαμορφωμένο πίνακα.

Πιο αναλυτικά , η βάση περιέχει τους αντίστοιχους πίνακες :

- Users : Ο πίνακας users περιέχει τα στοιχεία όλων των χρηστών ανεξαρτήτως του ρόλου τους , έχοντας όμως το αντίστοιχο ξένο κλειδί στον πίνακα Roles, όπου γίνεται η συσχέτιση.
- Services : Ο πίνακας services παίζει καθοριστική σημασία στην εφαρμογή καθώς μέσω αυτού γίνεται η ταυτοποίηση των επιτρεπόμενων ενεργειών που μπορεί να εκτελέσει ο κάθε χρήστης. Οι πρώτες 5 εγγραφές περιλαμβάνουν τις βασικές ενέργειες που μπορεί να κάνει ο admin ή ο γραμματέας , και οι 3 τελευταίες αφορούν περισσότερο τις λειτουργίες που μπορεί να κάνει ο απλός χρήστης ή φοιτητής , και η χρήση τους είναι τέτοια ώστε να διασφαλίζεται μέσω των web services ότι ο χρήστης έχει δικαίωμα να κάνει κάποια ενέργεια . Περισσότερη ανάλυση υπάρχει στο κεφάλαιο των web services.
- Roles : Εδώ αποθηκεύονται οι ρόλοι των χρηστών που απαρτίζουν το σύστημα.
- LoginKeys : Σε αυτόν τον πίνακα αποθηκεύεται ένα μοναδικό τυχαίο κλειδί που δίνεται στον κάθε χρήστη κατά την επιτυχημένη είσοδό του στο σύστημα. Η συσχέτιση του κλειδιού και του χρήστη γίνεται μέσω του πεδίου user_FK όπου υπάρχει το username του χρήστη. Το κλειδί σβήνεται όποτε ο χρήστης κάνει logout απο οποιαδήποτε εφαρμογή (web/desktop).
- Lessons : Περιέχει τα μαθήματα.
- Credentials : Περιέχει τα username και password του κάθε χρήστη.
- JNCT_Students_Lessons : Ο πίνακας αυτός είναι ένα junction table για την ορθή απεικόνιση της M:N συσχέτισης μεταξύ των φοιτητών και των μαθημάτων.
- JNCT_Roles_Services : Ο πίνακας αυτός , ομοίως με παραπάνω είναι ο ενδιαμέσος πίνακας για την M:N συσχέτιση μεταξύ ρόλων και υπηρεσιών.

Παρακάτω ακολουθεί το σχήμα της βάσης δεδομένων όπως αυτή έχει δομηθεί στην εργασία μας.



2. Desktop Εφαρμογή:

Η desktop εφαρμογή όπως αλλωστε έχει ήδη αναφερθεί προηγουμένως δίνει τη δυνατότητα μόνο σε admins και γραμματείς να εκτελούν συγκεκριμένες ενέργειες. Παρακάτω θα εστιάσουμε στις κύριες κλάσεις και μια γενική περιγραφή τους , καθώς ο κώδικας είναι επαρκώς σχολιασμένος στα αρχεία που παραδίδονται.

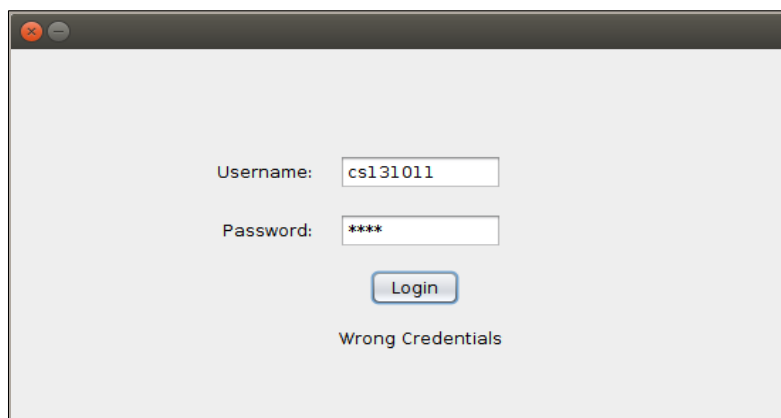
User.java:

Η κλάση User είναι βοηθητική κλάση , μέσω της οποίας αποθηκεύονται και ανακτώνται τα συγκεκριμένα χαρακτηριστικά του χρήστη αλλά και της "συνεδρίας" του στην Desktop εφαρμογή. Όλες οι μεταβλητές και οι μέθοδοι είναι static για να διασφαλίσουμε πως τα πεδία και οι μέθοδοι της κλάσης User κατα τη δημιουργία της, θα είναι μοναδικά και προσβάσιμα απο όλες τις υπόλοιπες κλάσεις χωρίς να είναι απαραίτητη η δημιουργία νέων αντικειμένων ή η παραμετρική αποστολή των references τους απο αντικείμενο σε αντικείμενο.

LoginForm.java:

Η loginForm αποτελεί το GUI της βασικής οθόνης login της εφαρμογής. Εδώ ελέγχονται μέσω συγκεκριμένων μεθόδων τα credentials του χρήστη που θέλει να κάνει login και εφόσον αυτά υπάρχουν στη βάση δίνεται πρόσβαση στην κεντρική οθόνη επιλογών.

Δεν δίνεται η δυνατότητα σε φοιτητές (ρόλος=3) να κάνουν login.

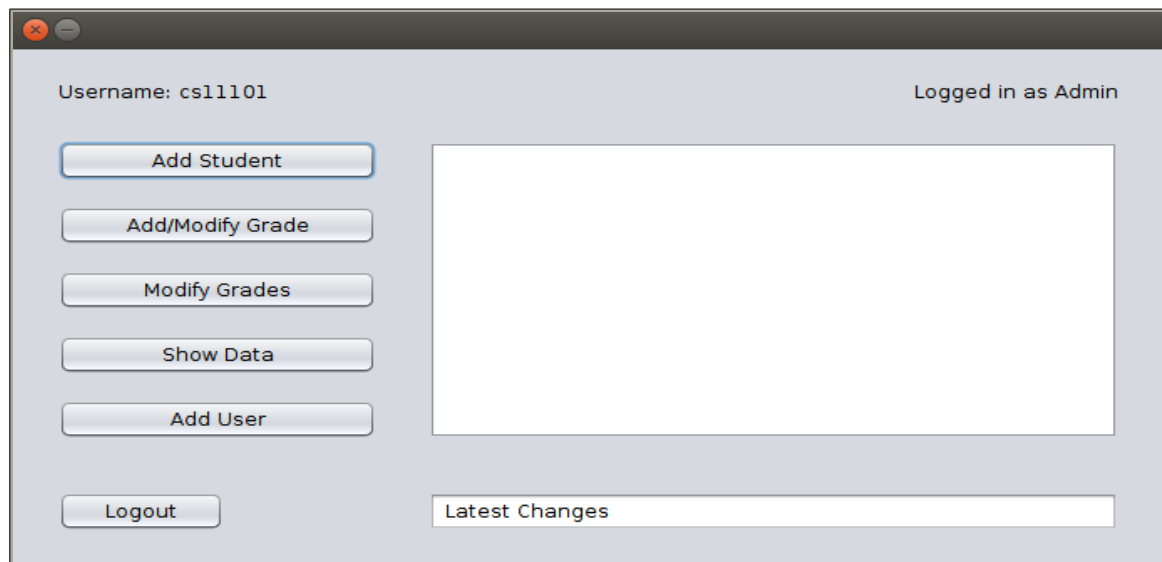


Στη συνέχεια καλούνται τα web services που είναι υπεύθυνα για το ρόλο αλλά και τις επιτρεπόμενες υπηρεσίες του κάθε χρήστη. Η κλάση είναι υπεύθυνη να ενημερώσει τα στοιχεία στην User μέσω των αντίστοιχων setters . Στην περίπτωση που τα στοιχεία χρήστη είναι σωστά η φόρμα εισόδου κλείνει, και δημιουργείται ένα νέο αντικείμενο της adminMain.

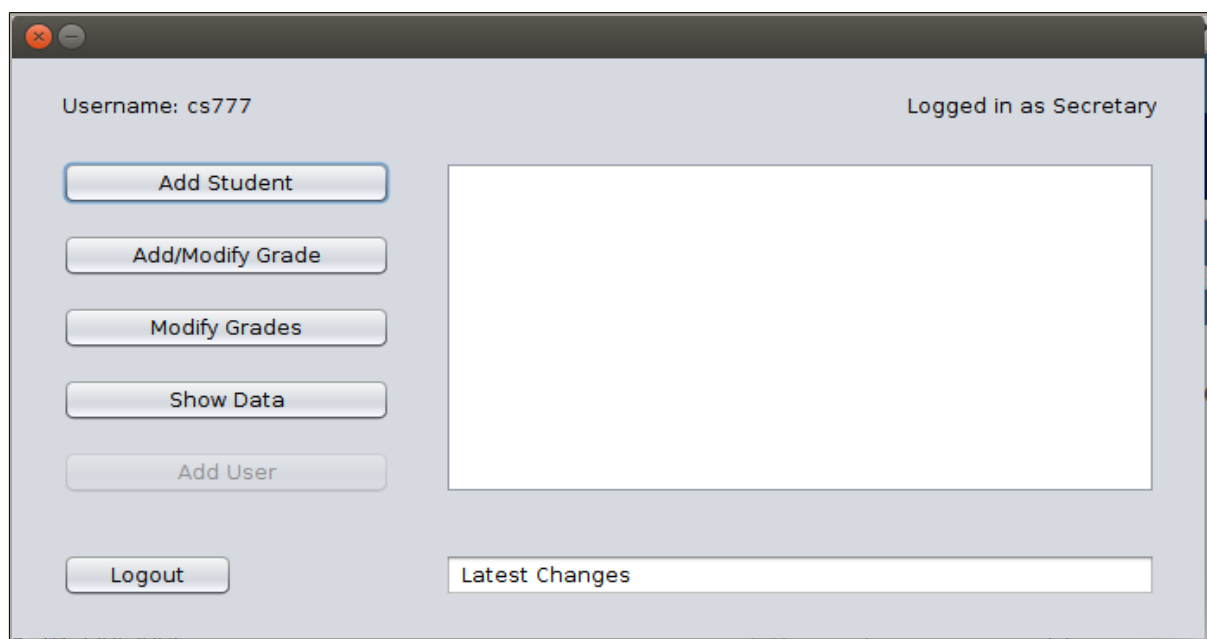
AdminMain.java :

Παρά την ονομασία της παραπάνω κλάσης που παραπέμπει σε μια οθόνη σχετική μονάχα με τον admin , η adminMain είναι επι της ουσίας η γενική οθόνη όλων των χρηστών που χρησιμοποιούν την εφαρμογή. Για να συνάδει με τις απαιτήσεις του συστήματος και τους ρόλους που έχουν δοθεί ενεργοποιεί κάθε φορά τα buttons με τα οποία επιτρέπεται να αλληλεπιδράσει ο χρήστης. Αυτός ο έλεγχος έχει γίνει προηγουμένως απο το στάδιο login. Εδώ απλά ελέγχεται αν οι υπηρεσίες που έχει ο χρήστης είναι και στα αντίστοιχα buttons. Η διαδικασία αυτή γίνεται μέσω της μεθόδου checkServices() και καλείται κατα την δημιουργία των components του GUI.

Παρακάτω , ακολουθούν δύο screenshots με login διαφορετικών χρηστών.



Login ως administrator (επιτρέπονται όλες οι ενέργειες)



Login ως γραμματέας (δεν επιτρέπεται το add User)

Όπως έχει αναφερθεί και προηγουμένως η ενεργοποίηση των κουμπιών προέρχεται απο μια δυναμικά ορισμένη διαδικασία και δεν έχει ουδεμία σχέση με τον κώδικα που έχει υλοποιηθεί για την ανάπτυξη της desktop εφαρμογής. Με κάθε αλλαγή στη βάση δεδομένων , επι παραδείγματι αν δοθεί μέσω της βάσης η δυνατότητα να κάνει add user ο γραμματέας , οι αλλαγές θα αποτυποθούν αυτομάτως στην εφαρμογή. Ομοίως και με απαγορεύσεις στις ενέργειες.

AddStudent.java :

Η κλάση addStudent όπως άλλωστε φανερώνει και το όνομα της , είναι υπεύθυνη για την δημιουργία του γραφικού περιβάλλοντος , τη συλλογή των προς αποθήκευση στοιχείων αλλά και την κατάλληλη μορφοποίηση τους για μεταφορά στα web services, όσον αφορά την προσθήκη νέου φοιτητή στη βάση δεδομένων του συστήματος. Η εισαγωγή των στοιχείων γίνεται σε ειδικά διαμορφωμένα πεδία , και όταν πατηθεί το κουμπί υποβολής εμφανίζεται κατάλληλο μήνυμα ενημέρωσης του χρήστη (πχ user added successfully) και τα διαγράφονται οι ήδη τυπομένες στα πεδία τιμές. Η μορφοποίηση των δεδομένων αφορά τον τύπο json και γίνεται μέσω του αντίστοιχου κώδικα:

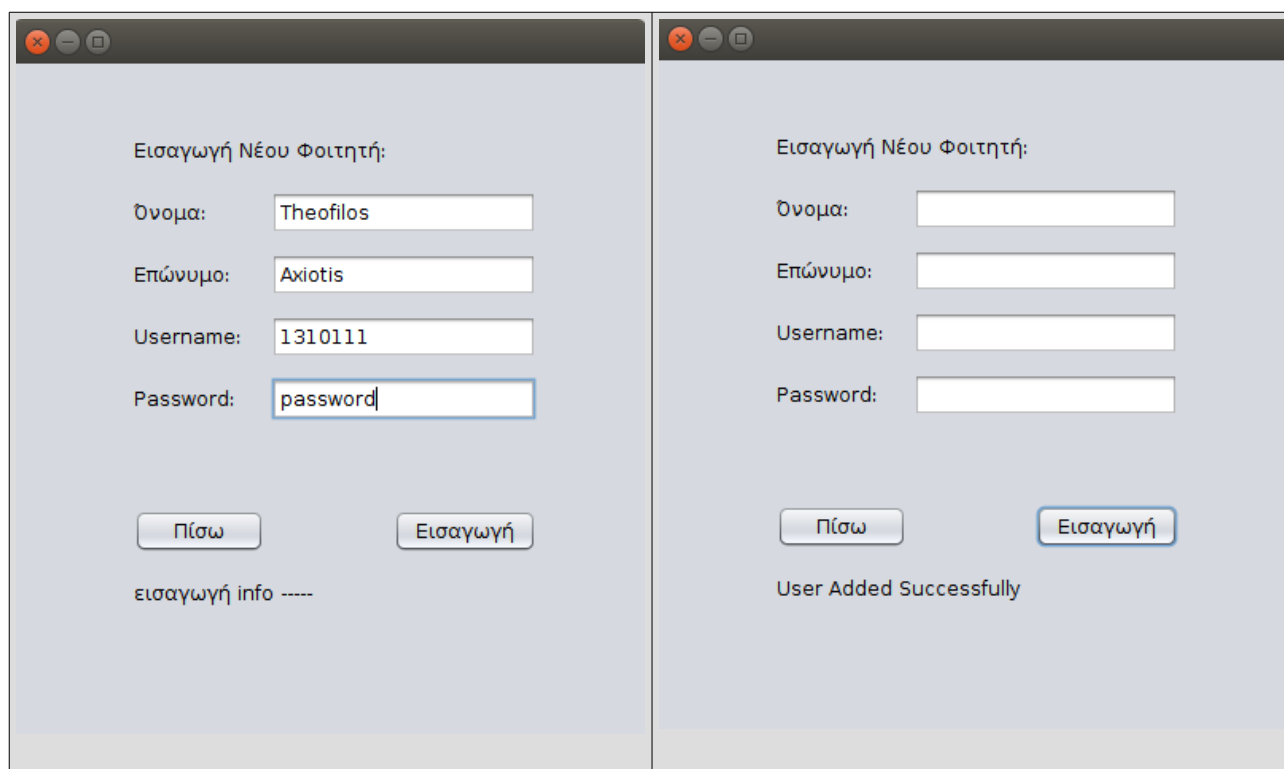
```
//δημιουργία του json που θα σταλεί στο webService, και περιέχει τα στοιχεία
//του φοιτητή.
JsonBuilderFactory factory=Json.createBuilderFactory(null);
JsonObjectBuilder json = factory.createObjectBuilder();

JsonObject myJson;
json= json.add("student", factory.createObjectBuilder()
.add("name", name)
.add("last_name", last_name)
.add("phone", phone)
.add("username", username)
.add("password", password));

myJson = json.build();
```

Στη συνέχεια ακολουθεί κατάλληλο encoding ώστε να γίνεται επιτυχώς η μεταφορά των δεδομένων μέσω του URI. Τέλος , καλείται η βοηθητική κλάση addStudentClient όπου και γίνεται η τελική κλήση του web service για την προσθήκη νέου φοιτητή στη βάση. Ο ρόλος δίνεται (=3) κατευθείαν στο αντίστοιχο web service.

Παρακάτω ακολουθούν δύο screenshots επιτυχημένης λειτουργίας της εισαγωγής φοιτητή:



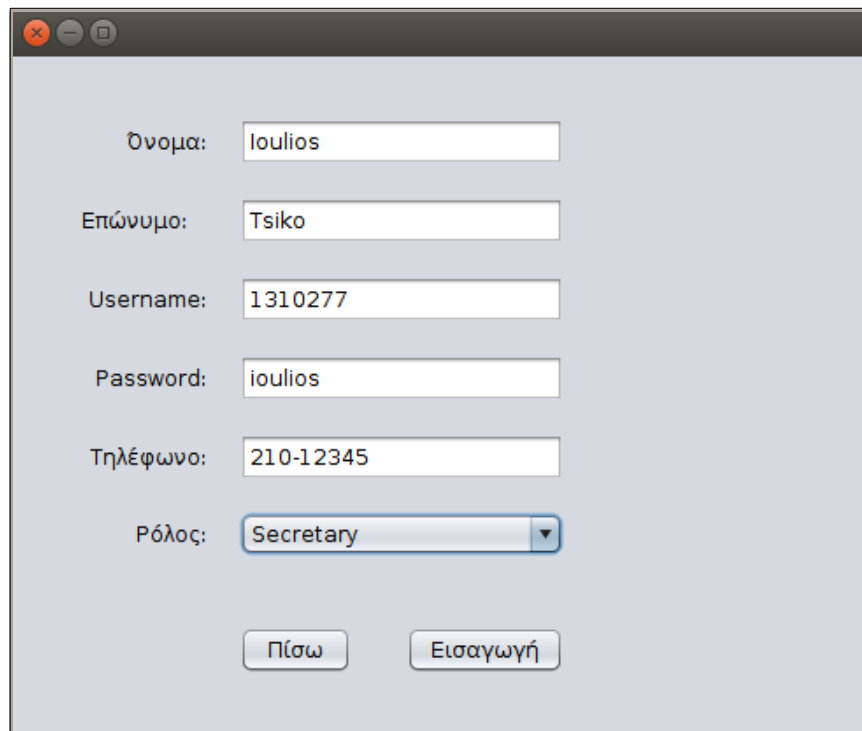
The image displays two side-by-side screenshots of a web application interface for adding a new student. Both windows have a title bar with standard OS controls (close, maximize, minimize).

Left Screenshot: The window is titled "Εισαγωγή Νέου Φοιτητή:". It contains four input fields: "Όνομα:" (Name) with the value "Theofilos", "Επώνυμο:" (Surname) with the value "Axiotis", "Username:" with the value "1310111", and "Password:" with the value "password". Below the fields are two buttons: "Πίσω" (Back) and "Εισαγωγή" (Add). At the bottom, there is a label "εισαγωγή info -----".

Right Screenshot: The window is titled "Εισαγωγή Νέου Φοιτητή:". The input fields are now empty. Below the fields are the same two buttons: "Πίσω" (Back) and "Εισαγωγή" (Add). At the bottom, there is a message "User Added Successfully".

AddUser.java :

Όμοια λειτουργία με την προαναφερθείσα addStudent. Η μόνη ειδοποιώς διαφορά είναι ότι εδώ ο χρήστης έχει τη δυνατότητα να επιλέξει μεταξύ άλλων τον ρόλο που θα δώσει στον προς εισαγωγή χρήστη. Μέσω ενός comboBox που έχει ως πεδία τις τιμές "Admin" και "Secretary" μπορεί να επιλέξει το ρόλο που θέλει να αποδώσει. Η επιλογή του μετουσιώνεται σε έναν κωδικό ρόλου, όπως αυτός είναι διαμορφωμένος στη βάση δεδομένων και μαζί με τα υπόλοιπα στοιχεία μορφοποιούνται κατάλληλα και αποστέλονται μέσω της βοηθητικής κλάσης addUserClient στο αντίστοιχο web service για εισαγωγή στη βάση δεδομένων.



Όνομα:

Επώνυμο:

Username:

Password:

Τηλέφωνο:

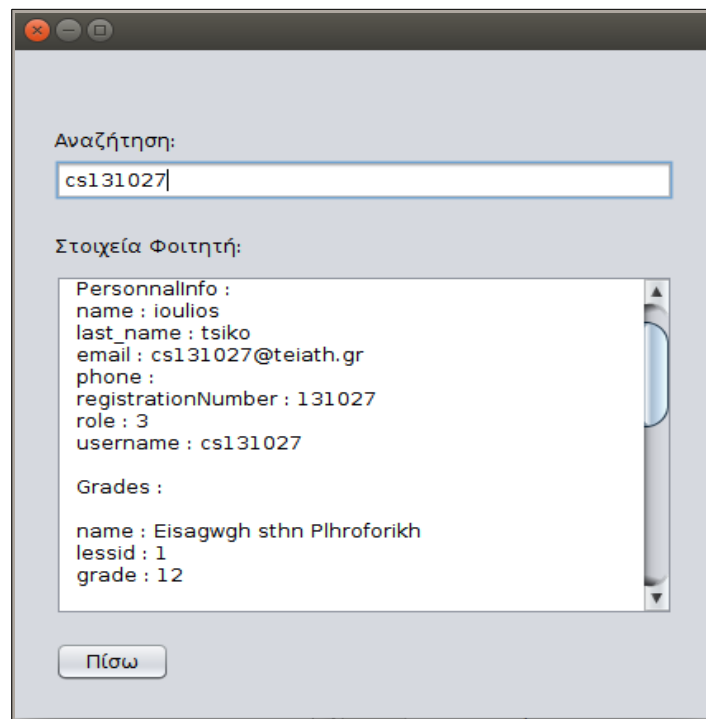
Ρόλος:

ModifyFromFile.java :

Η κλάση modifyFromFile δέχεται ένα αρχείο τύπου .xl με βαθμούς φοιτητών και ανανεώνει αυτομάτως τους βαθμούς των χρηστών που περιέχονται σε αυτό. Για να γίνει αυτό έχουν χρησιμοποιηθεί οι βιβλιοθήκες gmeleSheets , και οι παρεχόμενες σε αυτές μέθοδοι. Όταν το αρχείο διαβαστεί επιτυχώς καλείται το αντίστοιχο web service περνώντας τα μορφοποιημένα σε json δεδομένα προς αποθήκευση.

ShowStud.java :

Η κλάση showStudent είναι υπεύθυνη για την εμφάνιση των επιστρεφόμενων αποτελεσμάτων απο το web service "getStudInfo". Όπως φαίνεται και απο τον τίτλο , χρησιμοποιείται για την αναζήτηση ενός φοιτητή και την εμφάνιση των προσωπικών του στοιχείων όπως είναι όνομα , ΑΜ αλλά και βαθμών. Σε περίπτωση λάθους στο username προς αναζήτηση εμφανίζεται ένα καταλληλα διαμορφωμένο μήνυμα.



Aναζήτηση:

cs131027

Στοιχεία Φοιτητή:

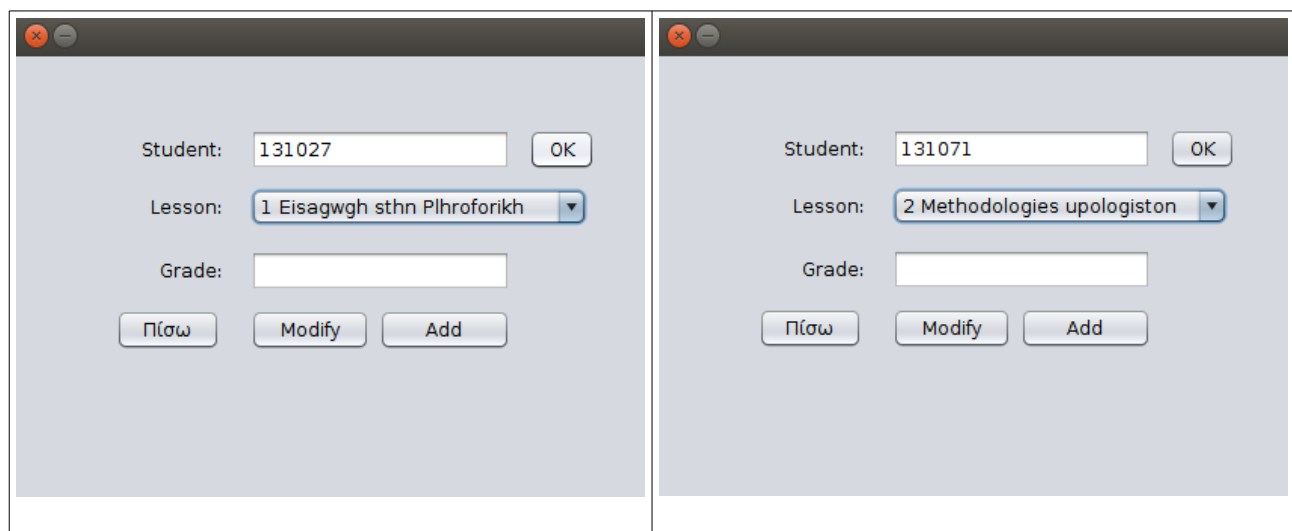
PersonnalInfo :
name : ioulios
last_name : tsiko
email : cs131027@teiath.gr
phone :
registrationNumber : 131027
role : 3
username : cs131027

Grades :
name : Eisagwgh sthn Plhroforikh
lessid : 1
grade : 12

Πίσω

ModifyGrade.java :

Η κλάση modifyGrade αφορά την λειτουργία της εισαγωγής ή μορφοποίησης ενός βαθμού φοιτητή στο σύστημα. Αρχικά δεν εμφανίζονται στοιχεία μαθημάτων ή λίστα φοιτητών. Ο χρήστης πρέπει να πληκτρολογήσει τον αριθμό μητρώου του φοιτητή του οποίου θέλει να μορφοποιήσει το βαθμό και στη συνέχεια να πατήσει το κουμπί ok. Αυτομάτως θα κληθεί κατάλληλα διαμορφωμένο web service που θα επιστρέψει τα μαθήματα του φοιτητή με το δεδομένο username. Αυτό το αποτέλεσμα θα αποτελέσει τα πεδία του ComboBox. Έτσι επιτυγχάνεται η ελαχιστοποίηση περιπτώσεων λάθους απο την πλευρά του χρήστη εάν εμφανίζονταν όλα τα μαθήματα.



Student: 131027 OK

Lesson: 1 Eisagwgh sthn Plhroforikh

Grade:

Πίσω Modify Add

Student: 131071 OK

Lesson: 2 Methodologies upologiston

Grade:

Πίσω Modify Add

Παραπάνω φαίνεται, πώς σε διαφορετικούς χρήστες τα μαθήματα είναι διαφορετικά. Στο comboBox δεν έχει επιλεγεί κάποια τιμή. Η τιμή που εμφανίζεται είναι η πρώτη διαθέσιμη τιμή. Στην αριστερή φωτογραφία υπάρχει το μάθημα με κωδικό 1 ως πρώτη τιμή, ενώ στη δεξιά το μάθημα με κωδικό 2, αφού ο χρήστης 131071 δεν έχει το μάθημα 1.

Clients :

Όλες οι κλάσεις που η ονομασία τους τελειώνει με την λέξη client είναι βοηθητικές κλάσεις στις οποίες γίνεται απλώς η κλήση του web service με τις κατάλληλες παραμέτρους και τον τύπο, αλλά και η επιστροφή των τιμών από αυτά.

3. Web Services :

Τα web services αποτελούν τον πυρήνα λειτουργίας των εφαρμογών. Όλες οι ενέργειες περιέχουν τη χρήση/κλήση των αντίστοιχων υπηρεσιών. Όπως ζητείται και μέσω της εκφώνησης της εργασίας όλα συμπεριλαμβανομένης και της σύνδεσης με τη βάση δεδομένων γίνονται μέσω των web services.

Τα web services είναι δομημένα με τέτοιο τρόπο ώστε να διασφαλίζεται πως θα καλούνται μόνο από χρήστες που έχουν δικαίωμα να τα καλέσουν. Πέραν των όποιων περιορισμών έχουν ληφθεί από τις δύο εφαρμογές ώστε ο χρήστης να μην έχει πρόσβαση σε services που δεν δικαιούνται, πήραμε και μέτρα αποτροπής και ελέγχου των χρηστών και μέσα στον κώδικα υλοποίησης των web services. Πιο συγκεκριμένα, στο παρακάτω screenshot φαίνεται η βασική δομή των web services. Το συγκεκριμένο απόκομμα είναι από το web service εισαγωγής νέου χρήστη.

```
* @return ένα String που περιέχει ένα μήνυμα που αφορά την επιτυχία ή όχι του web service.
* @throws ClassNotFoundException
* @throws SQLException
*/
@GET
@Produces("text/plain")
public String putUser(@QueryParam ("usr") String username,@QueryParam ("key") String key,@QueryParam ("info") String info) {
    DBManager dbm=DBManager.getInstance(); // Πέρνει το μοναδικό αντικείμενο της κλάσης DBManager.
    User u=dbm.checkKey(username,key); // Ελέγχει αν υπάρχει χρήστης και επιστρέφει ένα αντικείμενο User
    if(u.getUserExist()) // Αν ο χρήστης υπάρχει τότε επιστρέφει true αλλιώς false.
    {
        if(u.hasService(serviceNum))// Αν ο χρήστης έχει δικαίωμα να καλεί το συγκεκριμένο web service τότε
            return dbm.addStudent(info); // Μετατρέπει το string σε json και προσθέτει το ν φοιτητή στην βάση
        else
            return "An error has occurred: Permission Denied";
    }
    else
        return "An error has occurred: You have to login";
}
```

Πιο αναλυτικά, μέσω της εντολής DBManager.getInstance() καταφέρνουμε να πάρουμε το μοναδικό* instance της κλάσης DBManager. Στη συνέχεια μέσω της κλάσης User αποθηκεύουμε το αποτέλεσμα της μεθόδου checkKey η οποία επιστρέφει ένα αντικείμενο τύπου User (User (false) αν δεν υπάρχει ο χρήστης), και ελέγχει αν το username συνάδει με το μοναδικό κλειδί ταυτοποίησης που δόθηκε κατά τη διαδικασία εισαγωγής. Στη συνέχεια γίνεται έλεγχος αν ο χρήστης υπάρχει στη βάση, και τέλος αν ο συγκεκριμένος χρήστης έχει δικαίωμα να χρησιμοποιήσει το παρόν web service.

Αυτή η διαδικασία σε γενικές γραμμές ακολουθείται σε κάθε web service, κάθε φορά με μικρές αλλαγές ανάλογα με τις εκάστωτε ανάγκες.

DBManager :

Η κλάση DBManager είναι μια βοηθητική κλάση για τα web services που κάνει αυθεντικοποίηση του χρήστη και περιέχει όλες τις μεθόδους που καλούνται από τα Web Services. Επιτρέπεται μόνο ένα αντικείμενο της κλάσης και οι εξωτερικές κλάσεις μπορούν να το χρησιμοποιούν καλώντας την static μέθοδο getInstance. Αυτή η λειτουργία είναι απαραίτητη για λόγους ασφάλειας και ορθής λειτουργίας του συστήματος. Με αυτό πετυχαίνουμε να διαχειριζόμαστε ένα μόνο αντικείμενο και να μην επηρεάζουν οι ενέργειές μας η μία την άλλη. Τέλος , η χρήση singletons μας βοηθά και σε θέμα απόδοσης του προγράμματος , καθώς η δημιουργία ενός αντικειμένου τύπου dbmanager κάθε φορά θα ήταν χρονοβόρα και θα απαιτούσε σπατάλη πόρων που θα μπορούσαμε αλλιώς να είχαμε αποφύγει.

Παρακάτω ακολουθεί μία πολύ σύντομη περιγραφή όλων των διαφορετικών web services που υλοποιήσαμε για την εύρυθμη λειτουργία του συστήματος. Δεν γίνεται εκτενής αναφορά στον κώδικα λειτουργίας των web services καθώς η βασική τους δομή έχει αναλυθεί παραπάνω και υπάρχει εμπεριστατωμένος σχολιασμός στον πηγαίο κώδικα που παραδίδουμε.

- AddStudentResource : Δέχεται ως όρισμα τα στοιχεία ενός φοιτητή σε μορφή json, και τον εισάγει στη βάση δεδομένων. Επιστρέφει καταλλήλα διαμορφωμένο μήνυμα επιτυχίας ή λάθους.
- AddUserResource : Δέχεται ως όρισμα τα στοιχεία ενός user (διαχειριστής ή γραμματέας) σε μορφή json, και τον εισάγει στη βάση δεδομένων. Επιστρέφει καταλλήλα διαμορφωμένο μήνυμα επιτυχίας ή λάθους.
- GetLessonsResource : Επιστρέφει όλη τη λίστα μαθημάτων ή μόνο τα μαθήματα ενός φοιτητή. Δέχεται ως όρισμα ένα username. Αν η τιμή αυτή είναι 0 , επιστρέφει τη λίστα όλων των διαθέσιμων μαθημάτων.
- GetLessonResource : Δέχεται ως όρισμα ένα κωδικό μαθήματος και επιστρέφει πληροφορίες σχετικές μόνο με το συγκεκριμένο μάθημα.
- GetServiceResource : Δέχεται ως ορίσματα ένα username (του χρήστη που έχει συνδεθεί) και τον ρόλο του ως string. Πχ cs11101 , admin . Εκτελεί ένα sqlQuery και επιστρέφει τις διαθέσιμες ενέργειες που μπορεί να εκτελέσει ο χρήστης σε μορφή string.
- GetStudentInfoResource : Πέρνει τα στοιχεία του φοιτητή, τα μαθήματα του και τους βαθμούς από την βάση, τα εισάγει σε ένα αντίκείμενο json, τα μετατρέπει σε String και τα επιστρέφει. Ως όρισμα δέχεται ένα όνομα χρήστη προς αναζήτηση. Επιστρέφει ένα String που περιέχει τα στοιχεία σε μορφή json array. Σε περίπτωση σφάλματος επιστρέφει ένα μήνυμα που αναφέρει τι πήγε λάθος.
- LoginResource : Δέχεται ως ορίσματα τον κωδικό και το username του χρήστη και εάν ο χρήστης δεν έχει ήδη συνδεθεί επιστρέφει ένα τυχαία δημιουργημένο μοναδικό key. Σε περίπτωση που ο χρήστης υπάρχει επιστρέφει την τιμή 0 , και δεν δημιουργεί νέο key.
- LogoutResource : Δέχεται ως ορίσματα τον κωδικό και το username του χρήστη και διαγράφει το δοθέν κλειδί κλείνοντας έτσι την συνεδρία του χρήστη.

- **ModifyGrades** : Δέχεται ως όρισμα τα στοιχεία του φοιτητή σε μορφή json, και μετά απο σειρά ελέγχων αλλάζει τη βαθμολογία ενός φοιτητή στη βάση δεδομένων. Επιστρέφει κατάλληλα διαμορφωμένα μηνύματα. Χρησιμοποιείται και για την αλλαγή ενός μόνο βαθμού απο την desktop εφαρμογή αλλά και απο την αλλαγή βαθμών με τη χρήση αρχείου xl.
- **RolesResource** : Επιστρέφει τον ρόλο του χρήστη ως ένα απλό string. Το όνομα Χρηστη δίνετε σαν όρισμα.
- ***PutClassesResource** : Αυτό το web service δεν ήταν ένα απο τα ζητήματα της εκφώνησης. Πήραμε την πρωτοβουλία να το υλοποιήσουμε για λόγους πληρότητας. Η λειτουργία του αφορά τη δυνατότητα ενός φοιτητή να επιλέγει μόνος του απο λίστα μαθημάτων ποιά θέλει να παρακολουθήσει. Μέσω αυτού του service, που ως ορίσματα δέχεται το username, το μοναδικό key , και τη λίστα των μαθημάτων , ενημερώνεται στη βάση δεδομένων η λίστα των μαθημάτων που παρακολουθεί ο εκάστωτε φοιτητής.

4. Web Εφαρμογή :

Η web εφαρμογή αποτελείται απο servlets που διαχειρίζονται τα http requests και τις συνδέσεις. Η λειτουργία τους περιορίζεται σε βασικές ενέργειες ελέγχου των sessions ή και δημιουργία αυτών , αλλά και την δημιουργία του σκελετού html και της μορφοποίησης της εμφάνισης μέσω css. Η γενικότερη λειτουργία της εφαρμογής ξεκινά απο την σελίδα LoginPage. Εκεί ελέγχονται τα δεδομένα που δίνει ο χρήστης και εφόσον ανήκουν σε φοιτητή η σελίδα τον ανακατευθύνει στο HomePage. Σε περίπτωση που τα στοιχεία δεν ανήκουν σε φοιτητή αλλά σε admin κ.ο.κ , δεν γίνεται επιτρεπτή η είσοδος. Δεν δώσαμε τη δυνατότητα στους admins και γραμματείς να κάνουν login διαδικτυακά γιατί δεν υπάρχει νόημα. Δεν μπορούν να κάνουν κάποια λειτουργία πέραν του να δούν τα στοιχεία τους.

Στο HomePage εμφανίζεται μια λίστα μαθημάτων , αυτά που έχει επιλέξει ο φοιτητής , με τα στοιχεία τους αλλά και τη βαθμολογία. Στη συνέχεια υπάρχουν δύο buttons , εκ των οποίων το ένα κάνει το logout και το άλλο είναι ένας σύνδεσμος στη σελίδα των μαθημάτων. Στη σελίδα Courses δίνεται η δυνατότητα στον φοιτητή να δηλώσει ένα νέο μάθημα απο τη λίστα που του εμφανίζεται. Όταν επιλέξει μάθημα η λίστα των μαθημάτων που εμφανιζόταν στο HomePage ανανεώνεται και περιέχει πλέον και το νέο δηλωμένο μάθημα , με βαθμό , φυσικά, μια παύλα.

Η κλάση RestCaller είναι βοηθητική κλάση που περιέχει όλες τις υπομεθόδους για να καλούνται τα υπόλοιπα web services. Η λειτουργία της παρουσιάζει αρκετές ομοιότητες με τη DBManager που συναντήσαμε στα web services.

Τέλος, στο φάκελο υπάρχει το αρχείο style.css στο οποίο γίνεται μια υποτυπώδης μορφοποίηση των στοιχείων των σελιδών.