ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής



Εργασία Μαθήματος «Προγραμματισμός στο διαδίκτυο και στον παγκόσμιο ιστό»

|  |  |
| --- | --- |
| **ΑΣΚΗΣΗ 03** | ***ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΤΟ ΔΙΑΔΙΚΤΥΟ ΚΑΙ ΣΤΟΝ ΠΑΓΚΟΣΜΙΟ***  ***ΙΣΤΟ*** |
| Φοιτητές στην εργασία: | Π19204 – Γεώργιος Σεϊμένης |
| Π19064 – Ευστράτιος Καρκάνης |
| Π19032 – Νικόλαος Γεωργιάδης |
| Ημερομηνία παράδοσης: | 14/07/2021 |

**Εκφώνηση της άσκησης**

Στόχοι εργασίας: Ολοκλήρωση λειτουργικότητας 3-tier εφαρμογής, ολοκλήρωση server-side τεχνολογιών(servlets και προαιρετικά jsp), επικοινωνία με βάση δεδομένων, ολοκλήρωση λειτουργιών.

Στην τελική εργασία του μαθήματος θα επεκτείνετε την 2η Άσκηση ώστε να ολοκληρώσετε την εφαρμογή τριών επιπέδων (3-tier), η οποία θα υλοποιεί όλες τις λειτουργίες (μεθόδους) που ορίσατε στην 1η Άσκηση (με τις πιθανές αλλαγές που έγιναν).

Αναλυτικά Βήματα:

1. Επέκταση web project προηγούμενης άσκησης

1.1. Στην τελική εργασία θα επεκτείνετε τη λειτουργικότητα του web project που δημιουργήσατε στην προηγούμενη άσκηση και θα υλοποιήσετε όλη την ζητούμενη λειτουργικότητα για κάθε κατηγορία χρηστών.

2. Δημιουργία διαδικτυακής διεπαφής

2.1. Για την είσοδο των χρηστών στο σύστημα θα υλοποιείστε μηχανισμό login με username και password. Το password θα αποθηκεύεται σε κρυπτογραφημένη (hashed+salted) μορφή. Από την αρχική σελίδα οι διάφοροι χρήστες θα μπορούν να έχουν πρόσβαση στις λειτουργίες τους.

2.2. Σε αυτό το βήμα, θα υλοποιήσετε τις διαδικτυακές διεπαφές (html ή jsp σελίδες) που θα χρησιμοποιούν οι χρήστες όλων των κατηγοριών (Ασθενείς, Ιατροί, Διαχειριστές) για να αλληλεπιδρούν με την εφαρμογή και να χρησιμοποιούν τις αντίστοιχες μεθόδους που απαιτούνται.

**2.2.1.** Θα υπάρχει ένα κεντρικό μενού σε μία index.html (ή index.jsp) σελίδα, η οποία θα είναι η αρχική σελίδα για όλους τους χρήστες. Μετά τo login θα προβάλλεται το μενού λειτουργιών κάθε χρήστη ανάλογα με την κατηγορία στην οποία ανήκει.

**2.2.2.** Λειτουργίες Ασθενών (Patient): Οι Ασθενείς θα μπορούν να εκτελούν κατ’ ελάχιστο τι λειτουργίες: προβολή ιστορικού προηγούμενων ραντεβού, προβολή διαθέσιμων κενών για κλείσιμο ραντεβού με έναν γιατρό κάποιας ειδικότητας, κλείσιμο ραντεβού ακύρωση ραντεβού (σε περίπτωση που το ραντεβού είναι προγραμματισμένο τουλάχιστον 3 ημέρες μετά).

**2.2.3.** Λειτουργίες Ιατρών (Doctor): Οι Ιατροί θα μπορούν να εκτελούν κατ’ ελάχιστο τις λειτουργίες: καταχώρηση διαθεσιμότητας για ραντεβού (ανά μήνα), προβολή πίνακα ραντεβού, ακύρωση ραντεβού (σε περίπτωση που είναι τουλάχιστον 3 ημέρες μετά).

**2.2.4.** Λειτουργίες Διαχειριστή (Administrator). Οι Διαχειριστές θα μπορούν να εκτελούν κατ’ ελάχιστο τις λειτουργίες: εισαγωγή νέου Ιατρού και χρήστη, διαγραφή Ιατρού.

2.3. Η εφαρμογή θα υποστηρίζει διαχείριση συνόδου (session management) από τη στιγμή που ο χρήστης συνδέεται, μέχρι την αποσύνδεσή του από την εφαρμογή. Κατά την αποσύνδεση του χρήστη θα πρέπει να διαγράφεται το session.

3. Υλοποίηση επιπέδου Δεδομένων και σύνδεση εφαρμογής με τη βάση

3.1. Όλα τα δεδομένα θα αποθηκεύονται σε βάση δεδομένων, την οποία έχετε σχεδιάσει από την 2η Άσκηση (π.χ. μέσω mysql + mysql workbench ή postgres ή άλλης αντίστοιχης τεχνολογίας). Μπορείτε να προβείτε σε όποιες τροποποιήσεις θεωρείτε απαραίτητες. Προσθέσετε δοκιμαστικά δεδομένα στη βάση.

3.2. Διαμορφώστε κατάλληλα το project σας ώστε να συνδέσετε τη Βάση Δεδομένων που έχετε δημιουργήσει με τον application server σας, ως μία 3-tier εφαρμογή (σύνδεση του application server με τη Βάση Δεδομένων και της εφαρμογής σας μέσω του application server – μπορείτε να βρείτε αντίστοιχο παράδειγμα στα παραδείγματα κώδικα που περιλαμβάνονται στη σελίδα του μαθήματος).

4. Υλοποίηση επιπέδου επεξεργασίας (servlet)

4.1. Διαμορφώστε κατάλληλα το project σας ώστε να επικοινωνεί με τον application server της επιλογής σας (στα java παραδείγματα έχουμε χρησιμοποιήσει apache tomcat).

4.2. Υλοποιήσετε όλες τις λειτουργίες που προσφέρει η εφαρμογή σας χρησιμοποιώντας τεχνολογία servlet. Δημιουργήστε ένα ή περισσότερα servlet τα οποία θα δέχονται είσοδο από το επίπεδο διεπαφής (html ή jsp σελίδες και φόρμες), θα αναζητούν στη βάση δεδομένων τα στοιχεία που απαιτούνται ότι απαιτείται και θα επιστρέφουν το αποτέλεσμα στον εκάστοτε χρήστη ως δυναμική html σελίδα.

4.3. Προαιρετικά, μπορείτε να χρησιμοποιήσετε τεχνολογία jsp για τη δημιουργία και την διαμόρφωση των ιστοσελίδων.

**ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ**

[1 Γενική περιγραφή της εργασίας 6](#_Toc76997714)

[2 Επαναχρησιμοποίηση του κώδικα 6](#_Toc76997715)

[3 Κώδικας Προγράμματος 6](#_Toc76997716)

[3.1 Back End 6](#_Toc76997717)

[3.1.1 Ο Διαχειριστής (Admin) 6](#_Toc76997718)

[3.1.2 Ο Ασθενής (Patient) 7](#_Toc76997719)

[3.1.3 Ο ιατρός (Doctor) 7](#_Toc76997720)

[3.2 Front End 7](#_Toc76997721)

[3.2.1 Το μενού της εφαρμογής 7](#_Toc76997722)

[3.2.2 H σελίδα εγγραφής (register) 7](#_Toc76997723)

[3.2.3 H σελίδα σύνδεσης (login) 8](#_Toc76997724)

[3.2.4 Τα κύρια μενού των χρηστών 9](#_Toc76997725)

[4 Η βάση δεδομένων 25](#_Toc76997726)

[5 Βιβλιογραφικές πηγές 25](#_Toc76997727)

# Γενική περιγραφή της εργασίας

H παρούσα εργασία πρόκειται για ένα ολοκληρωμένο Web Project. Στην ουσία, η σύνθεση του περιλαμβάνει τόσο τα προγραμματιστικά στοιχεία της προηγούμενης εργασίας (κλάσεις και κώδικας), όσο και την εγκατάσταση ενός Web Server και μίας βάσης δεδομένων. Πιο συγκεκριμένα, στο συγκεκριμένο Project έχουν χρησιμοποιηθεί ο Web Server **Αpache Tomcat 8.5** και η βάση δεδομένων **MySQL**. Επιπρόσθετα, έχουν προστεθεί πολλά καινούρια αρχεία (.java, .jsp, .css, .png, .html) τα οποία είναι απαραίτητα για την ορθή εκτέλεση του project.

Αξίζει να σημειωθεί ότι ιδιαίτερη έμφαση έχει δοθεί τόσο στο GUI (Graphical User Interface) της εφαρμογής (χρησιμοποιώντας αρχεία JSP και CSS) όσο και στην υλοποίηση των λειτουργιών που εκτελούνται από την πλευρά του εξυπηρετητή. Οι λεπτομέρειες αυτές περιγράφονται αναλυτικότερα παρακάτω.

# Επαναχρησιμοποίηση του κώδικα

Αρκετά χρήσιμες φάνηκαν οι δύο προηγούμενες εργασίες σε οτιδήποτε αφορά την επαναχρησιμοποίηση των ήδη υπαρχουσών κλάσεων και κώδικα. Όλα τα στοιχεία τους ήταν ήδη φτιαγμένα, κι αφού έγιναν οι κατάλληλες τροποποιήσεις, χρησιμοποιήθηκαν παρέα με τα servlets και το γραφικό περιβάλλον της εφαρμογής.

# Κώδικας Προγράμματος

## Back End

### **3.1.1 Ο Διαχειριστής (Admin)**

Ο Διαχειριστής είναι ο «κυρίαρχος» της ιστοσελίδας, καθώς αυτός έχει όλες τις δυνατότητες συγκριτικά με οποιονδήποτε άλλο χρήστη. Η κυρίαρχη δυνατότητά του είναι η πρόσθεση και η αφαίρεση άλλων χρηστών στη βάση δεδομένων. Ενώ ο Ασθενής μπορεί να κάνει μόνος του την εγγραφή του, οι άλλοι χρήστες θα χρειαστούν τον Διαχειριστή για την δημιουργία του λογαριασμού τους.

Το Servlet του διαχειριστή, είναι αρκετά απλό, καθώς οι λειτουργίες του είναι αρκετά ίδιες σε κώδικα η μία με την άλλη.

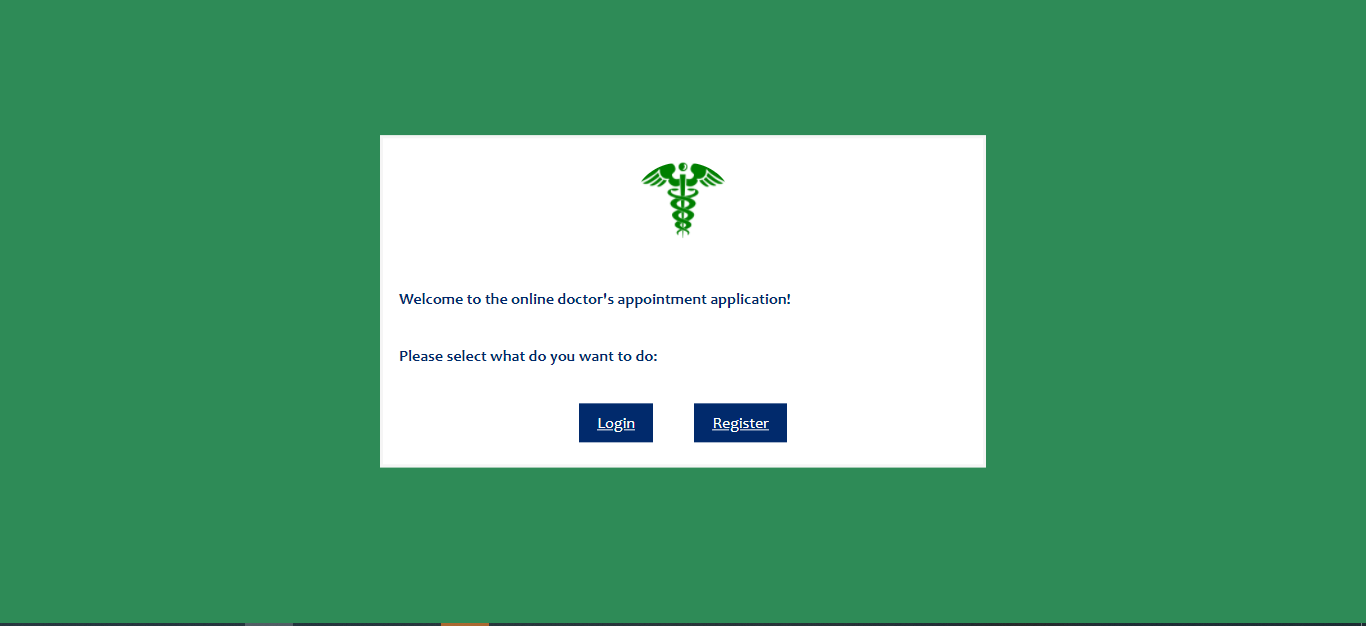
### **3.1.2 Ο Ασθενής (Patient)**

### **3.1.3 Ο ιατρός (Doctor)**

## Front End

### **3.2.1 Το μενού της εφαρμογής**

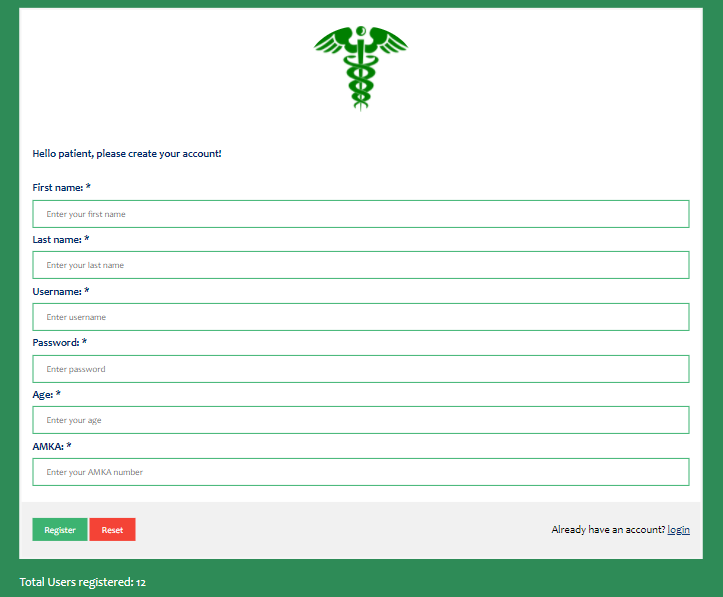
Το πρόγραμμα όταν φορτώσει εμφανίζει σε κάθε χρήστη (ιατρό, διαχειριστή ή ασθενή) την σελίδα *index.html*. Πρόκειται για το κεντρικό μενού της εφαρμογής. Ο χρήστης μπορεί να επιλέξει τι θέλει να κάνει (εγγραφή ή σύνδεση) και να συνεχίσει αναλόγως. Ακολουθεί σχετική εικόνα της αρχικής σελίδας:



Η σελίδα index.html

### **3.2.2 H σελίδα εγγραφής (register)**

Η σελίδα της εγγραφής (*register.jsp*) επιτρέπει σε έναν ασθενή να δημιουργήσει έναν νέο λογαριασμό στο σύστημα. Πρόκειται για μία φόρμα εισαγωγής στοιχείων (**First name, Last name, Username, Password, Age, AMKA**) του ασθενή. Τα στοιχεία αυτά αποθηκεύονται στη συνέχεια σε μία βάση δεδομένων, όταν ο ασθενής πατήσει το κουμπί “Register” της φόρμας. Ακολουθεί σχετική εικόνα της σελίδας:

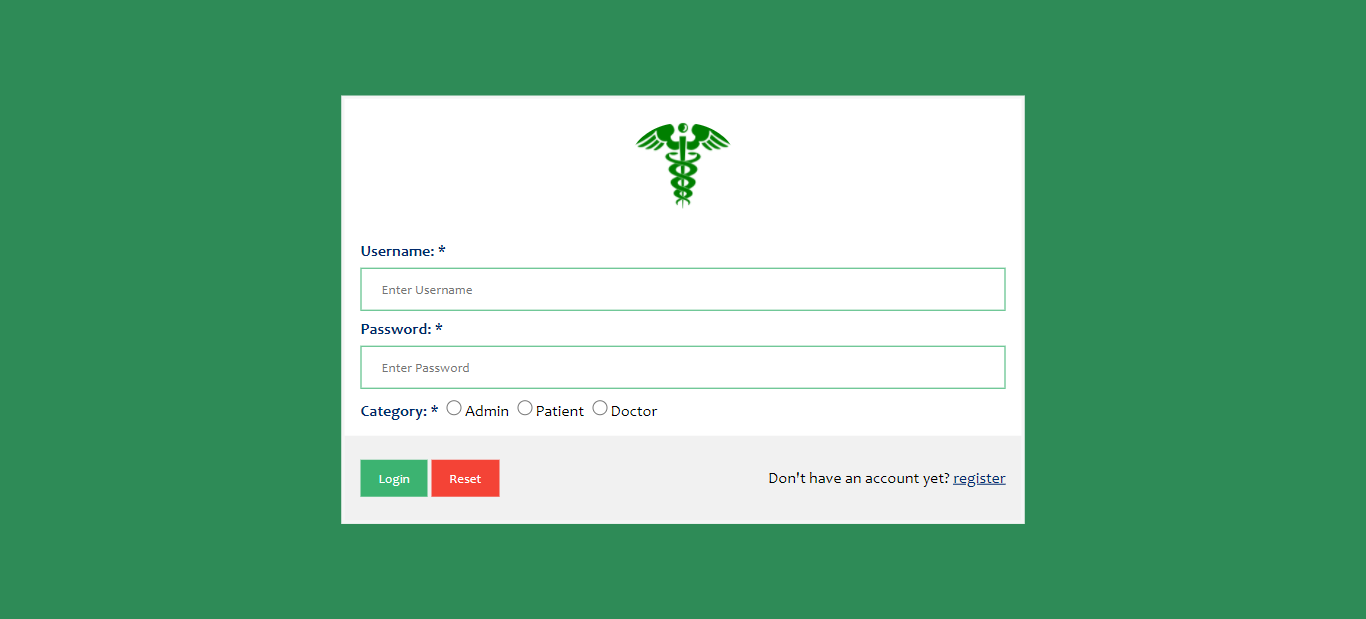


Η σελίδα register.jsp

Σημειώνεται ότι γίνεται έλεγχος εγκυρότητας των δεδομένων που εισάγονται τόσο από την πλευρά του πελάτη (χρήση JavaScript) όσο και από την πλευρά του server. Οι λεπτομέρειες αυτές έχουν αναλυθεί περεταίρω στο παραδοτέο της **δεύτερης** εργασίας.

### **3.2.3 H σελίδα σύνδεσης (login)**

Η σελίδα της εγγραφής (*login.jsp*) επιτρέπει σε έναν χρήστη (ιατρό, διαχειριστή ή ασθενή) να συνδεθεί στο σύστημα. Πρόκειται για μία φόρμα εισαγωγής στοιχείων (**Username, Password**) του χρήστη. Επιπλέον, ο χρήστης καλείται να συμπληρώσει και την κατηγορία στην οποία ανήκει, δηλαδή αν είναι ιατρός, ασθενής ή διαχειριστής. Αυτό είναι εφικτό με τα radio buttons που υπάρχουν στην φόρμα login. Τα κουμπάκια αυτά (radio buttons) βοηθούν την διαδικασία της σύνδεσης, έτσι ώστε να ενεργοποιείται κάθε φορά το αντίστοιχο servlet της εφαρμογής. Μόλις ο χρήστης πατήσει το κουμπί “*Login*” εμφανίζεται γίνεται ή όχι επιτυχής η σύνδεσή του στο σύστημα. Ακολουθεί σχετική εικόνα της σελίδας:



Η σελίδα login.jsp

Σε περίπτωση λανθασμένης εισαγωγής στοιχείων, εμφανίζεται στην οθόνη του χρήστη η σελίδα *fail.html*. Ακολουθεί σχετική εικόνα της σελίδας:



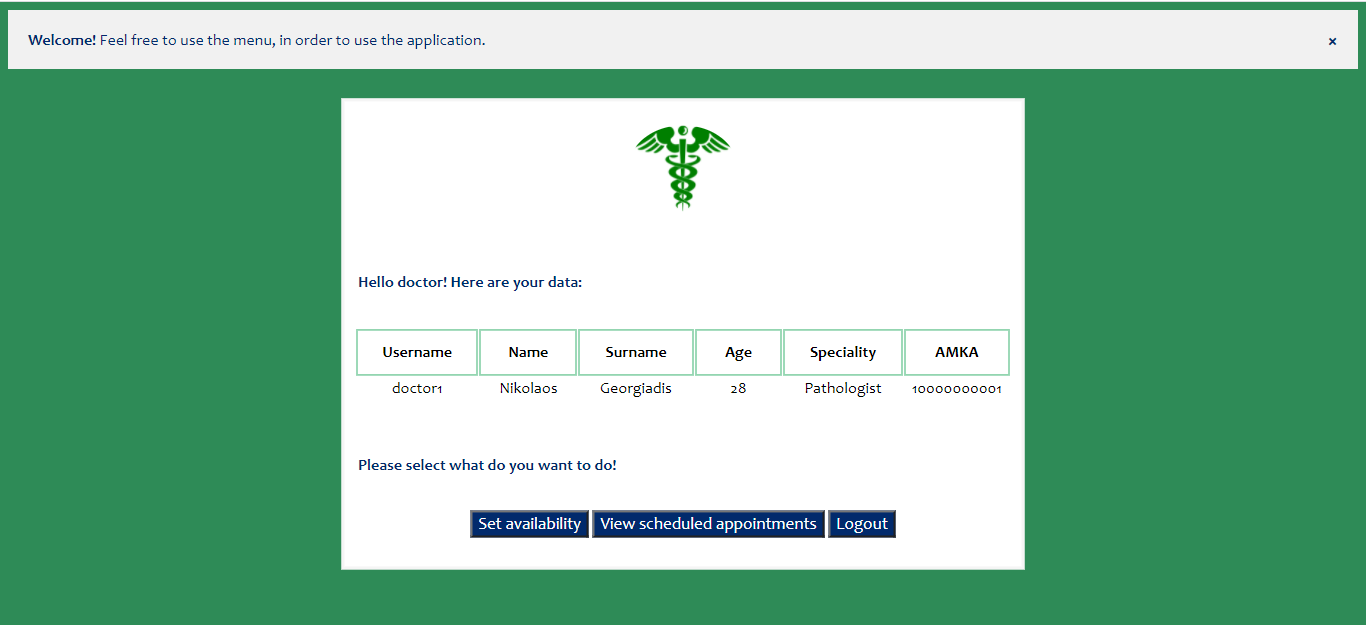
Η σελίδα fail.html

Σε περίπτωση επιτυχούς σύνδεσης εμφανίζεται το προσωπικό μενού κάθε χρήστη (ανάλογα την κατηγορία του χρήστη, εμφανίζεται διαφορετικό μενού με διαφορετικές λειτουργίες).

### **3.2.4 Τα κύρια μενού των χρηστών**

#### **3.2.4.1 Ιατρός**

Στην παρακάτω φωτογραφία βλέπουμε το κύριο μενού ενός χρήστη κατηγορίας ιατρού (doctor):



H σελίδα doctor\_main\_environment.jsp

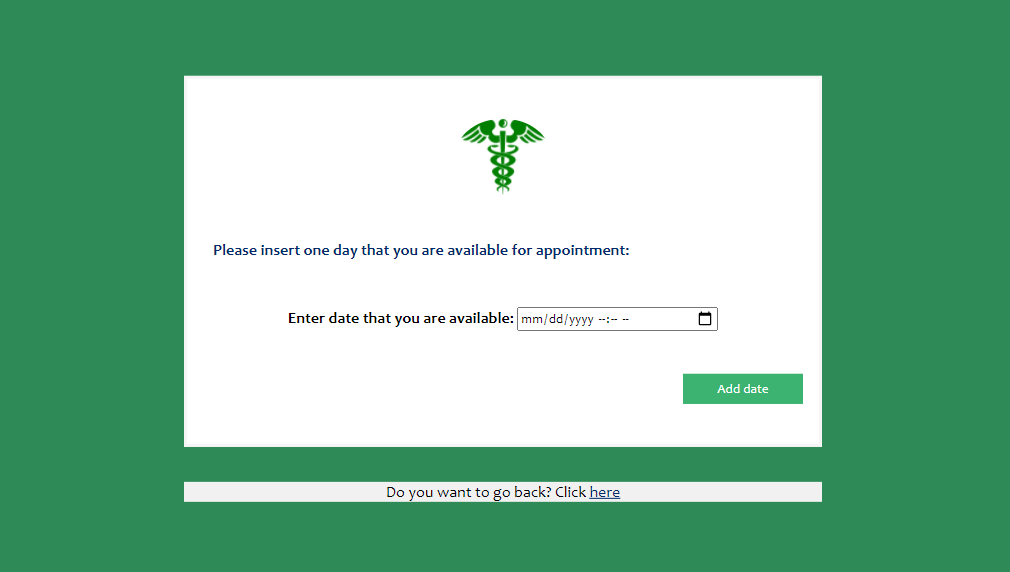
Η συγκεκριμένη σελίδα (*doctor\_main\_environment.jsp*) εμφανίζεται στην οθόνη μετά από μία επιτυχή σύνδεση (login) του ιατρού στο σύστημα. Η σελίδα περιλαμβάνει τα ακόλουθα στοιχεία για κάθε ιατρό που συνδέεται:

1. **τα στοιχεία του ιατρού** (όπως αυτά βρίσκονται αποθηκευμένα στη βάση δεδομένων). Συγκεκριμένα εμφανίζονται τα ακόλουθα στοιχεία σε μορφή πίνακα:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Username** | **Name** | **Surname** | **Age** | **Speciality** | **AMKA** |

1. **τρία κουμπιά ενεργειών** που εκτελούν έκαστο διαφορετική λειτουργία. Το πρώτο κουμπί (*Set availability*) επιτρέπει σε έναν ιατρό να ορίσει το πότε είναι διαθέσιμος για ραντεβού. Ακολούθως, το δεύτερο κουμπί (*View scheduled appointments*) επιτρέπει στον χρήστη να δει όλα τα προγραμματισμένα ραντεβού που έχει (ανά μήνα ή ανά εβδομάδα) και το τελευταίο κουμπί (*Logout*) κάνει αποσύνδεση του ιατρού από το σύστημα.

Όταν ένας ιατρός που έχει συνδεθεί στο σύστημα, πατήσει το πρώτο κουμπί (*Set availability*) για να ορίσει ο ίδιος την διαθεσιμότητά του για ραντεβού, εμφανίζεται η σελίδα *doctor\_set\_availability.jsp*. Η σελίδα αυτή εμφανίζεται στην επόμενη φωτογραφία:

****

Η σελίδα doctor\_set\_availability.jsp

Εδώ ο ιατρός απλώς εισάγει μία ημερομηνία και πατάει το κουμπί “*Add date*” για να δηλωθεί η διαθέσιμη αυτή ημερομηνία στο σύστημα. Αν η δήλωση της νέας διαθέσιμης ημερομηνίας του ιατρού αποθηκευτεί στη βάση σωστά, εμφανίζεται η ακόλουθη σελίδα:



Επιτυχής εισαγωγή διαθέσιμης ημερομηνίας

Όταν ο ιατρός πατήσει το δεύτερο κουμπί (*View scheduled appointments*) εμφανίζεται στην οθόνη η σελίδα *doctor\_view\_appointments.jsp,* όπως φαίνεται και στην επόμενη φωτογραφία:



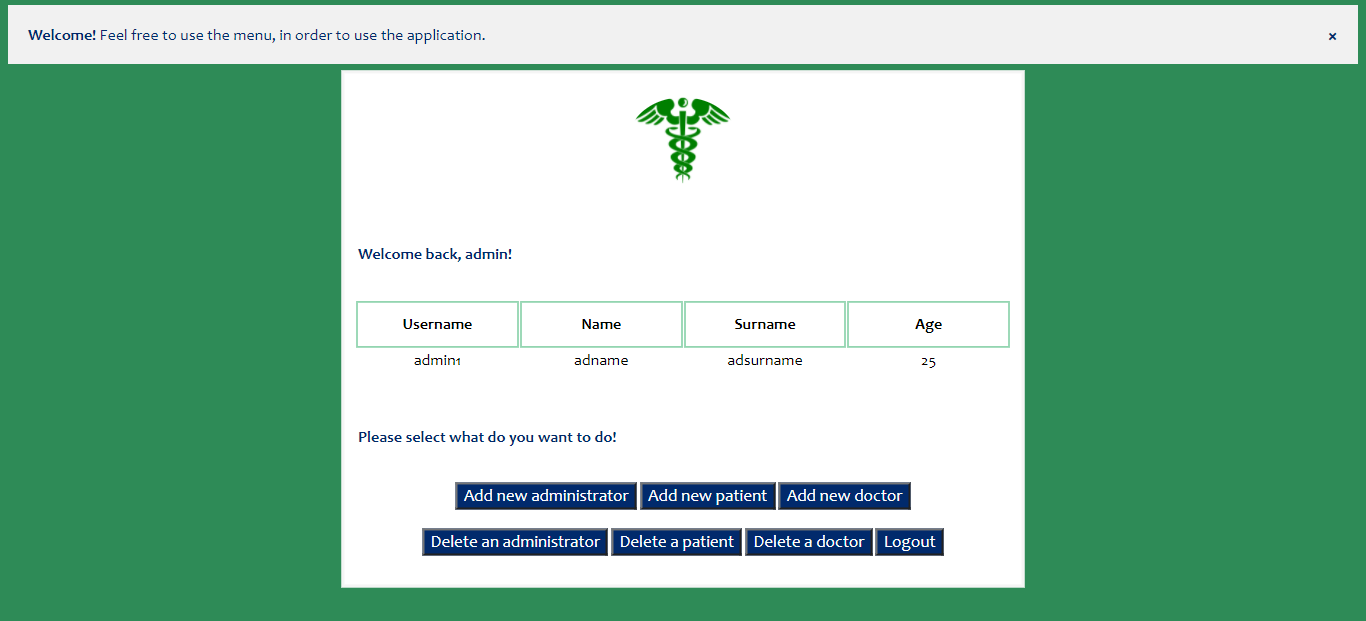
Η σελίδα doctor\_view\_appointments.jsp

Ο ιατρός λοιπόν, μπορεί να αναζητήσει όλα τα ραντεβού που έχει είτε ανά μήνα, είτε ανά εβδομάδα. Αυτό γίνεται με την βοήθεια της drop down λίστας.

Τέλος, το κουμπί “*Logout*” ΄κλείνει την σύνδεση (session) και μεταφέρεται ο έλεγχος του προγράμματος στην σελίδα *login.jsp.*

#### **3.2.4.2 Διαχειριστής**

Στην παρακάτω φωτογραφία βλέπουμε το κύριο μενού ενός χρήστη κατηγορίας διαχειριστή (admin):



Η σελίδα admin\_main\_environment.jsp

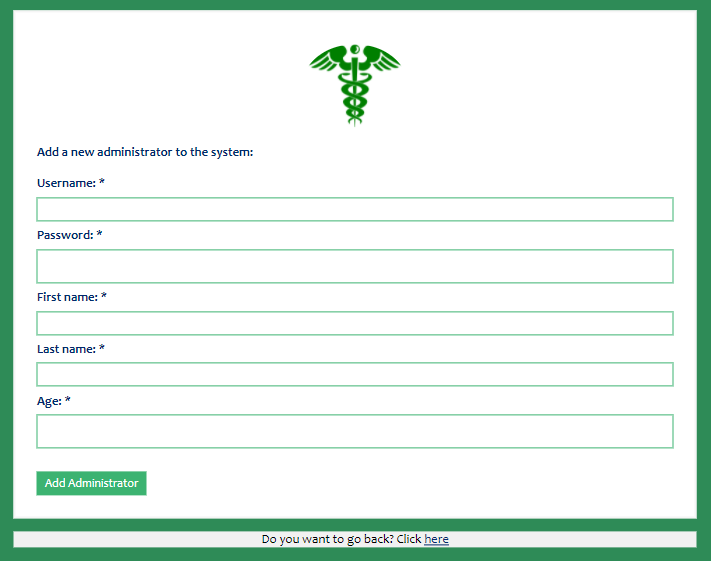
Η συγκεκριμένη σελίδα (*admin\_main\_environment.jsp*) εμφανίζεται στην οθόνη μετά από μία επιτυχή σύνδεση (login) του διαχειριστή στο σύστημα. Η σελίδα περιλαμβάνει τα ακόλουθα στοιχεία για κάθε admin που συνδέεται:

1. **τα στοιχεία του διαχειριστή** (όπως αυτά βρίσκονται αποθηκευμένα στη βάση δεδομένων). Συγκεκριμένα εμφανίζονται τα ακόλουθα στοιχεία σε μορφή πίνακα:

|  |  |  |  |
| --- | --- | --- | --- |
| **Username** | **Name** | **Surname** | **Age** |

1. **επτά κουμπιά ενεργειών** που εκτελούν έκαστο διαφορετική λειτουργία. Οι λειτουργίες κάθε κουμπιού αναλύονται τώρα:

* **κουμπί Add new administrator:** Όταν ένας διαχειριστής πατήσει το συγκεκριμένο κουμπί, στην ουσία μεταφέρεται στην σελίδα *add\_new\_admin.jsp*, για να εισάγει στο σύστημα έναν νέο διαχειριστή (admin). Η σελίδα αυτή φαίνεται στο επόμενο screenshot:

****

Εισαγωγή νέου διαχειριστή

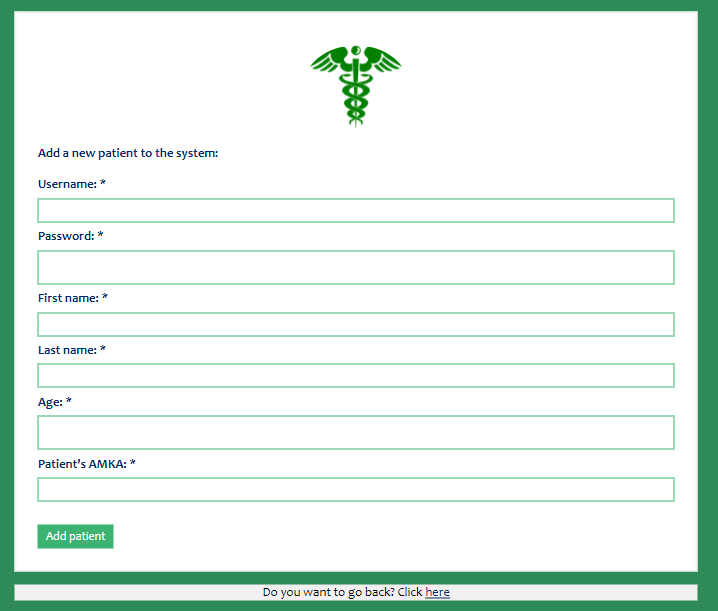
Ο διαχειριστής έπειτα εισάγει τα στοιχεία (του νέου διαχειριστή που θα προστεθεί στην εφαρμογή) στη φόρμα και πατάει το κουμπί “*Add Administrator”* για την εκτέλεση αυτής της ενέργειας. Έπειτα από την επιτυχή εισαγωγή, εμφανίζεται η ακόλουθη σελίδα:



Επιτυχής εισαγωγή διαχειριστή

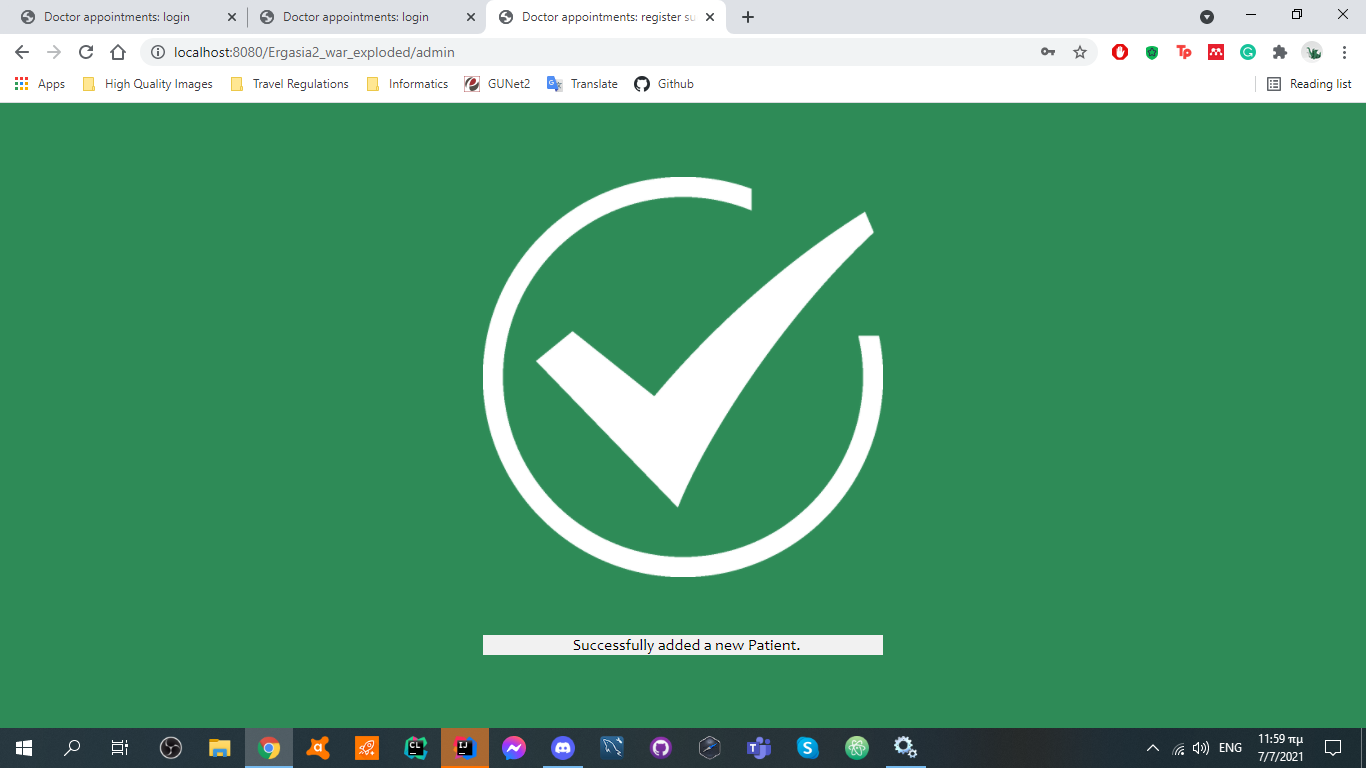
Σημειώνεται ότι κατά την υποβολή των στοιχείων, γίνεται έλεγχος εγκυρότητας των δεδομένων της φόρμας από την πλευρά του εξυπηρετητή. Οι έλεγχοι αυτοί έχουν αναλυθεί σε προηγούμενη εργασία.

* **κουμπί Add new patient:** Όταν ένας διαχειριστής πατήσει το συγκεκριμένο κουμπί, στην ουσία μεταφέρεται στην σελίδα *add\_new\_patient.jsp*, για να εισάγει στο σύστημα έναν νέο ασθενή (patient). Η σελίδα αυτή φαίνεται στο επόμενο screenshot:

****

Εισαγωγή νέου ασθενή

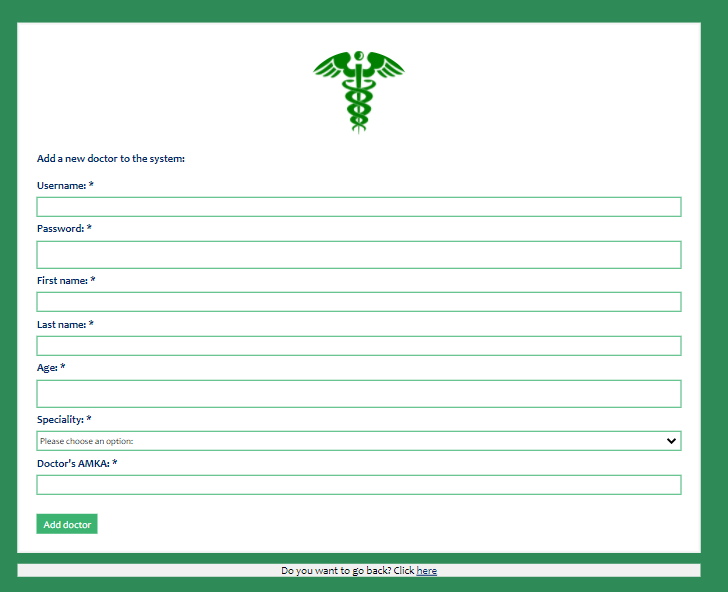
Ο διαχειριστής έπειτα εισάγει τα στοιχεία (του νέου ασθενή που θα προστεθεί στην εφαρμογή) στη φόρμα και πατάει το κουμπί “*Add patient”* για την εκτέλεση αυτής της ενέργειας. Έπειτα από την επιτυχή εισαγωγή, εμφανίζεται η ακόλουθη σελίδα:



Επιτυχής εισαγωγή ασθενή

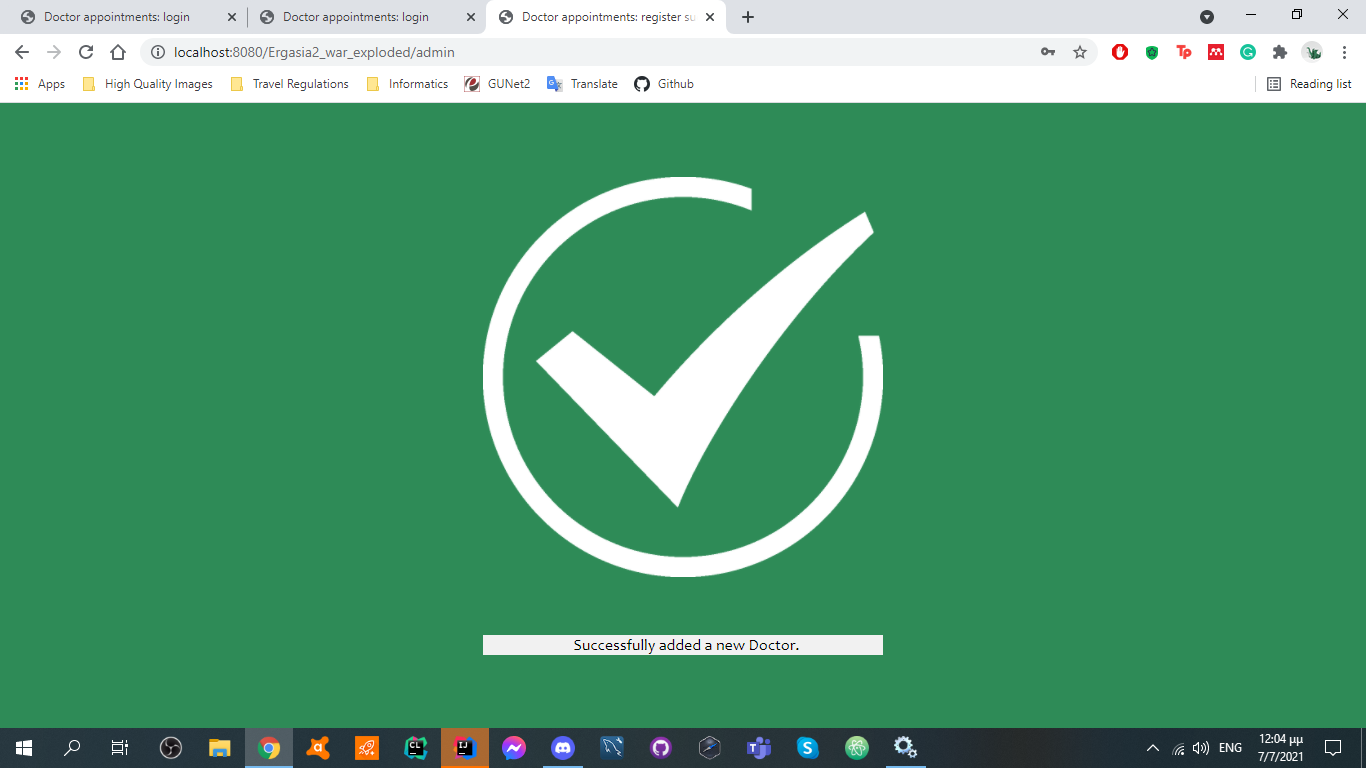
Σημειώνεται ότι κατά την υποβολή των στοιχείων, γίνεται έλεγχος εγκυρότητας των δεδομένων της φόρμας από την πλευρά του εξυπηρετητή. Οι έλεγχοι αυτοί έχουν αναλυθεί σε προηγούμενη εργασία.

* **κουμπί Add new doctor:** Όταν ένας διαχειριστής πατήσει το συγκεκριμένο κουμπί, στην ουσία μεταφέρεται στην σελίδα *add\_new\_doctor.jsp*, για να εισάγει στο σύστημα έναν νέο ιατρό (doctor). Η σελίδα αυτή φαίνεται στο επόμενο screenshot:

****

Εισαγωγή νέου ιατρού

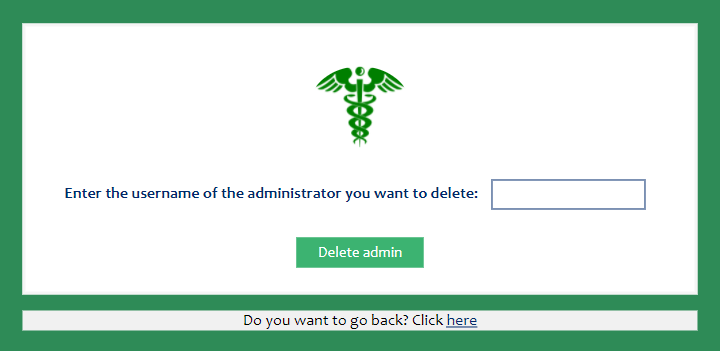
Ο διαχειριστής έπειτα εισάγει τα στοιχεία (του νέου ιατρού που θα προστεθεί στην εφαρμογή) στη φόρμα και πατάει το κουμπί “*Add patient”* για την εκτέλεση αυτής της ενέργειας. Έπειτα από την επιτυχή εισαγωγή, εμφανίζεται η ακόλουθη σελίδα:



Επιτυχής εισαγωγή ιατρού

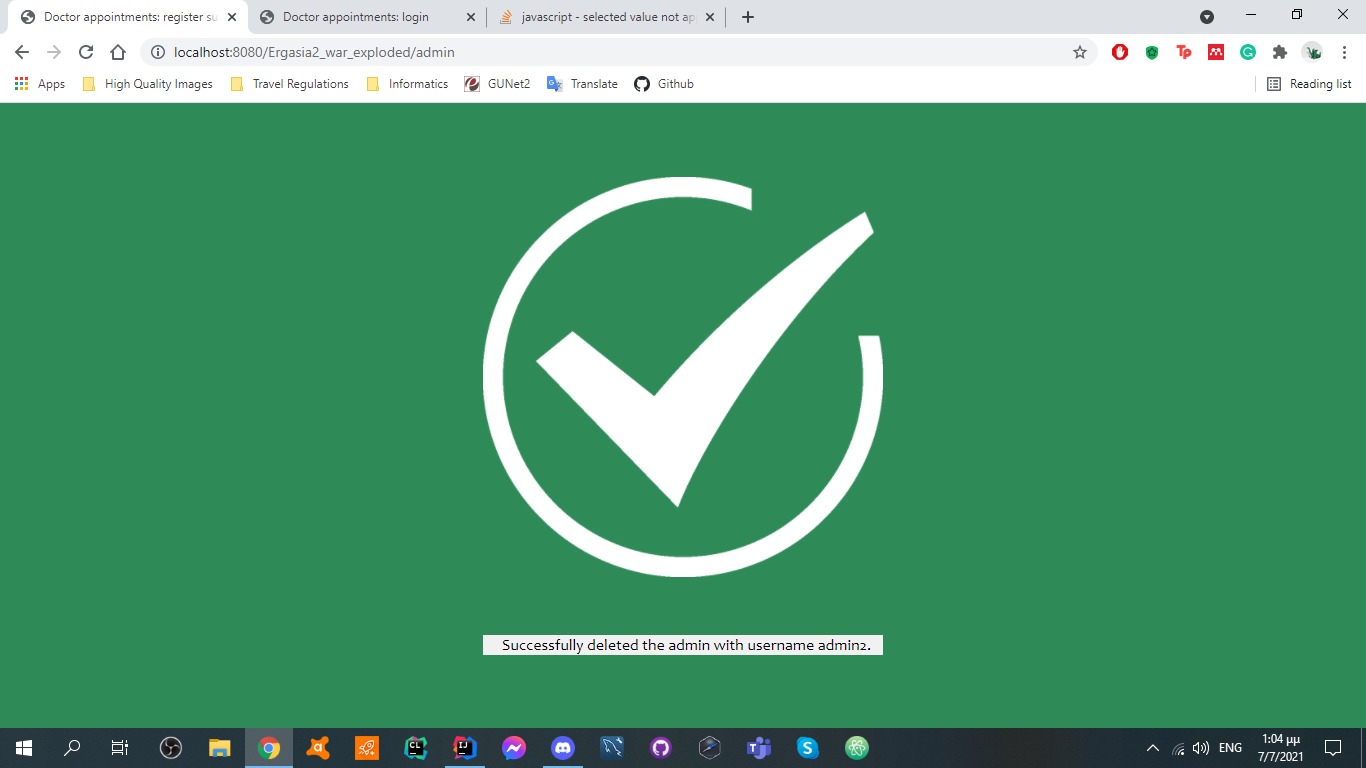
Σημειώνεται ότι κατά την υποβολή των στοιχείων, γίνεται έλεγχος εγκυρότητας των δεδομένων της φόρμας από την πλευρά του εξυπηρετητή. Οι έλεγχοι αυτοί έχουν αναλυθεί σε προηγούμενη εργασία.

* **κουμπί Delete an administrator:** Όταν ένας διαχειριστής πατήσει το συγκεκριμένο κουμπί, στην ουσία μεταφέρεται στην σελίδα *delete\_admin.jsp*, για να διαγράψει από το σύστημα έναν διαχειριστή (admin). Η σελίδα αυτή φαίνεται στο επόμενο screenshot:

****

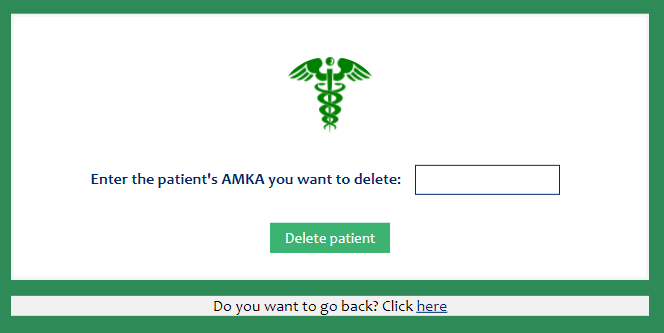
Διαγραφή ενός διαχειριστή

Ο διαχειριστής εισάγει το username του διαχειριστή που θέλει να διαγράψει. Η επιτυχής διαγραφή δηλώνεται με την εμφάνιση της ακόλουθης σελίδας:



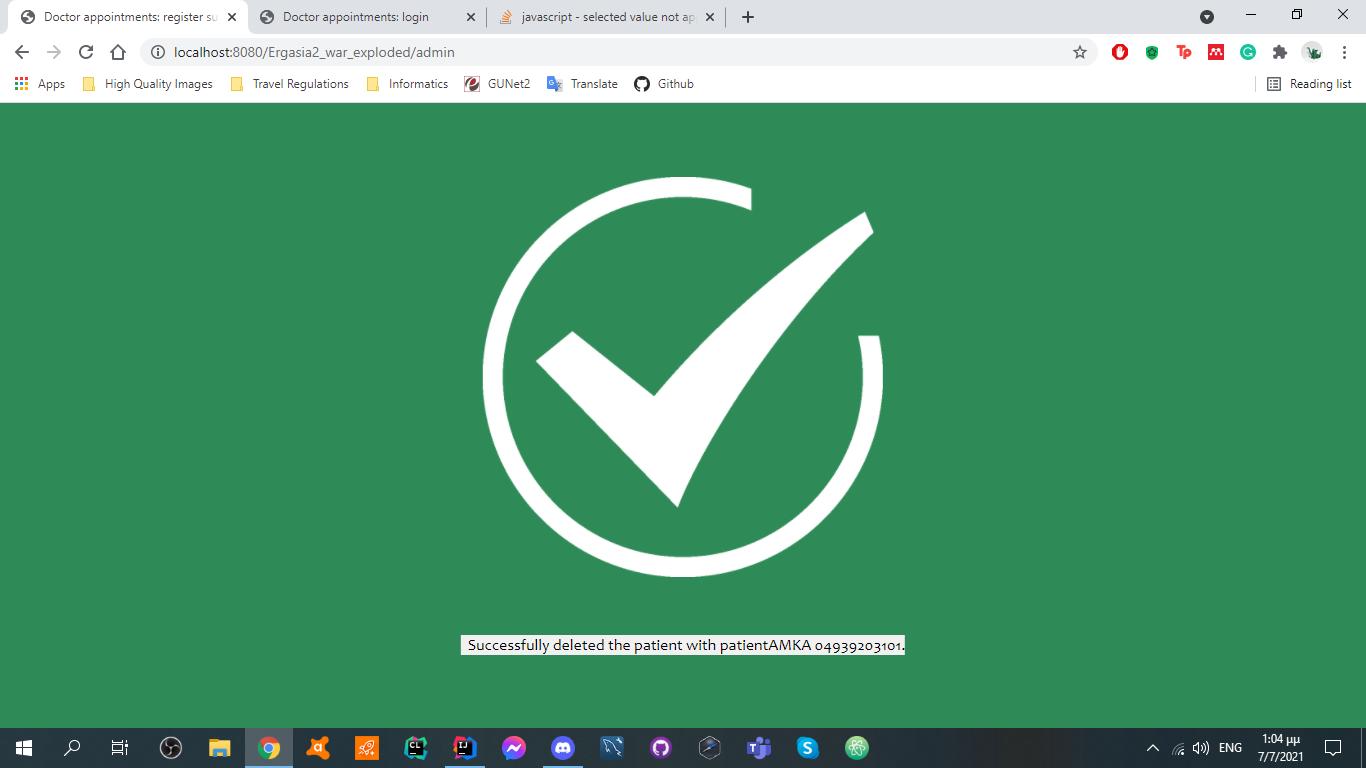
Επιτυχής διαγραφή διαχειριστή

* **κουμπί Delete a patient:** Όταν ένας διαχειριστής πατήσει το συγκεκριμένο κουμπί, στην ουσία μεταφέρεται στην σελίδα *delete\_patient.jsp*, για να διαγράψει από το σύστημα έναν ασθενή (patient). Η σελίδα αυτή φαίνεται στο επόμενο screenshot:

****

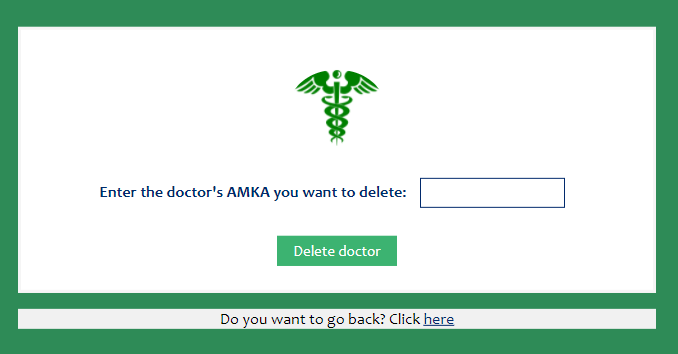
Διαγραφή ενός ασθενή

Ο διαχειριστής εισάγει το ΑΜΚΑ του ασθενή που θέλει να διαγράψει. Η επιτυχής διαγραφή δηλώνεται με την εμφάνιση της ακόλουθης σελίδας:



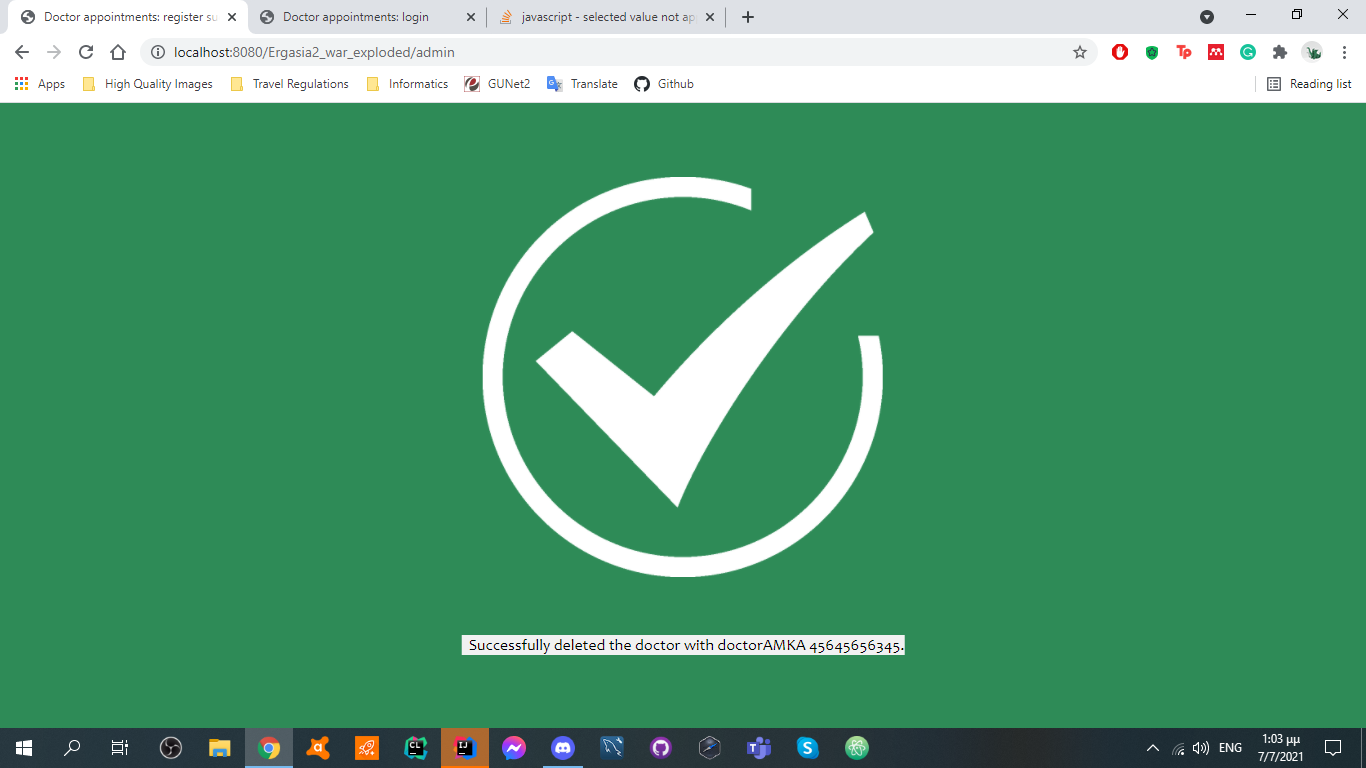
Επιτυχής διαγραφή ασθενή

* **κουμπί Delete a doctor:** Όταν ένας διαχειριστής πατήσει το συγκεκριμένο κουμπί, στην ουσία μεταφέρεται στην σελίδα *delete\_doctor.jsp*, για να διαγράψει από το σύστημα έναν ιατρό (doctor). Η σελίδα αυτή φαίνεται στο επόμενο screenshot:

****

Διαγραφή ενός ιατρού

Ο διαχειριστής εισάγει το ΑΜΚΑ του ιατρού που θέλει να διαγράψει. Η επιτυχής διαγραφή δηλώνεται με την εμφάνιση της ακόλουθης σελίδας:

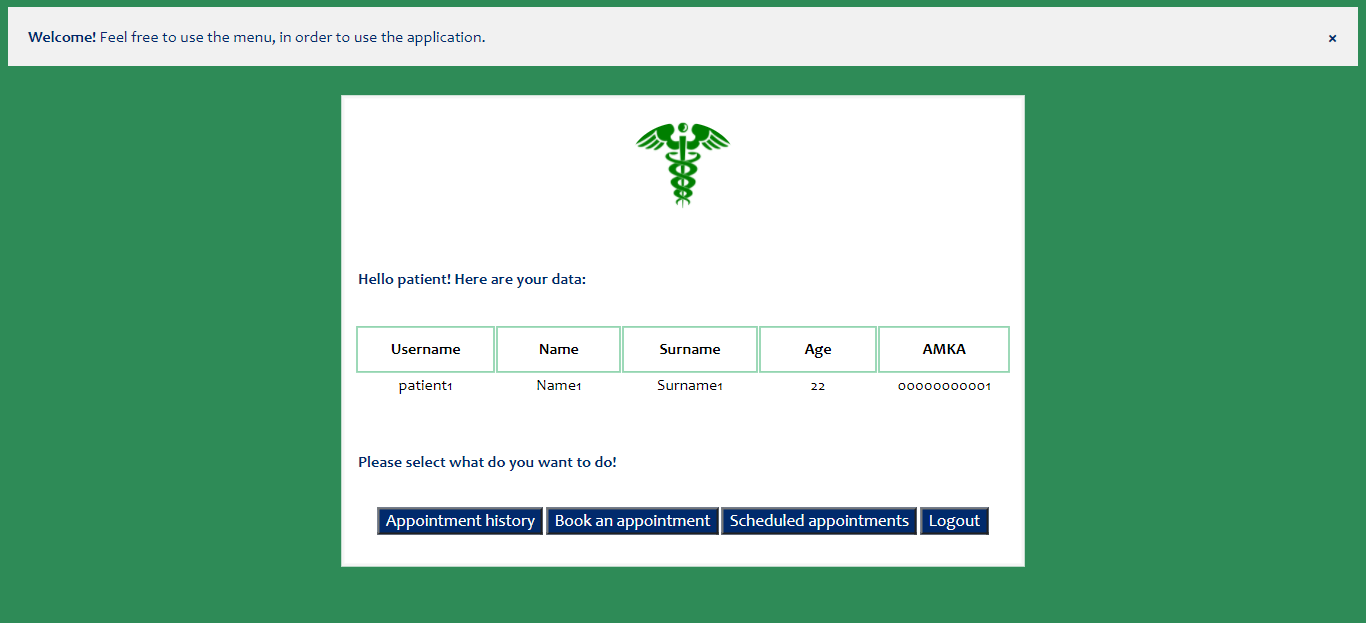


Επιτυχής διαγραφή ιατρού

* **κουμπί Logout:** όταν ο διαχειριστής πατήσει αυτό το κουμπί,κλείνει την σύνδεση (session) και μεταφέρεται ο έλεγχος του προγράμματος στην σελίδα *login.jsp.*

#### **3.2.4.3 Ασθενής**

Στην παρακάτω φωτογραφία βλέπουμε το κύριο μενού ενός χρήστη κατηγορίας ασθενή (patient):



Η σελίδα patient\_main\_environment.jsp

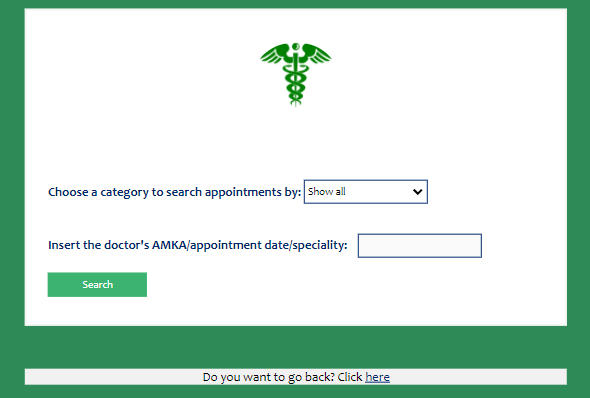
Η συγκεκριμένη σελίδα (*patient\_main\_environment.jsp*) εμφανίζεται στην οθόνη μετά από μία επιτυχή σύνδεση (login) του ασθενή στο σύστημα. Η σελίδα περιλαμβάνει τα ακόλουθα στοιχεία για κάθε ασθενή που συνδέεται:

1. **τα στοιχεία του ασθενή** (όπως αυτά βρίσκονται αποθηκευμένα στη βάση δεδομένων). Συγκεκριμένα εμφανίζονται τα ακόλουθα στοιχεία σε μορφή πίνακα:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Username** | **Name** | **Surname** | **Age** | **AMKA** |

1. **τέσσερα κουμπιά ενεργειών** που εκτελούν έκαστο διαφορετική λειτουργία. Το πρώτο κουμπί (*Appointment history*) επιτρέπει σε έναν ασθενή να δει το ιστορικό των ραντεβού που είχε. Το επόμενο κουμπί (*Book an appointment*) επιτρέπει στον ασθενή να κλείσει ένα νέο ραντεβού. Το τρίτο κουμπί (*Scheduled appointments*) επιτρέπει στον χρήστη να δει όλα τα προγραμματισμένα ραντεβού που έχει και, τέλος, το τελευταίο κουμπί (*Logout*) αποσυνδέει τον ασθενή από το σύστημα (κλείνοντας το session).

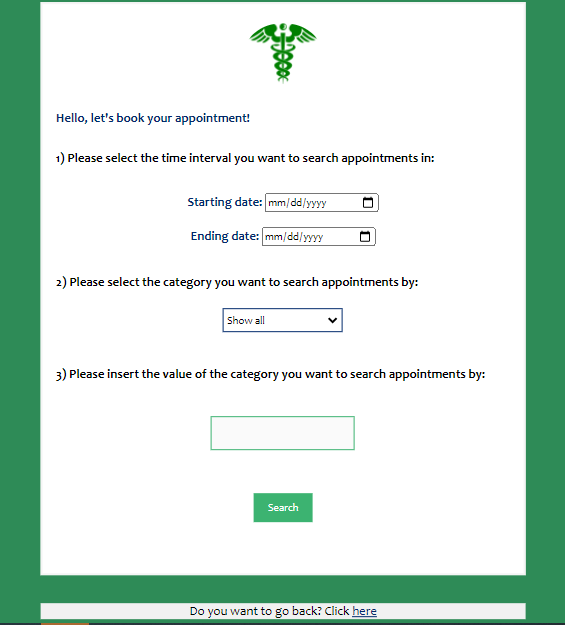
Όταν ένας ασθενής που έχει συνδεθεί στο σύστημα, πατήσει το πρώτο κουμπί (*Appointment history*) θα μεταφερθεί στην σελίδα *appointmenthistory.jsp* για να δει όλα τα ραντεβού που είχε στο παρελθόν. Η σελίδα φαίνεται στην επόμενη φωτογραφία:

****

Αναζήτηση παλαιών ραντεβού του ασθενή

Πατώντας το κουμπί “Search”, ο ασθενής μπορεί να δει **όλα** τα παλαιά ραντεβού του. Επιπλέον, με τη βοήθεια της drop down λίστας, ο ασθενής μπορεί να περιορίσει την αναζήτηση των ραντεβού του κατά **ημερομηνία ραντεβού**, **ειδικότητα του ιατρού** και κατά **ΑΜΚΑ του ιατρού**.

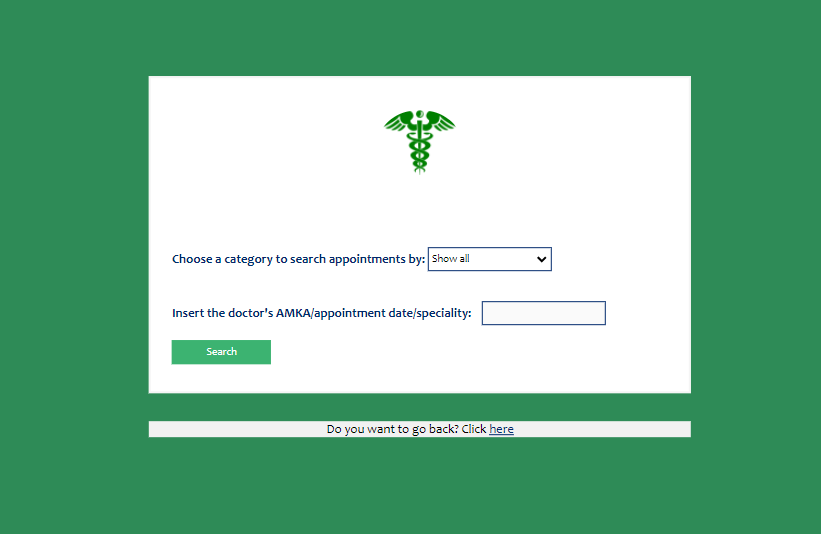
Όταν ο ασθενής πατήσει το δεύτερο κουμπί (*Book an appointment*) εμφανίζεται στην οθόνη η σελίδα *AvailableDoctorAppointments.jsp,* όπως φαίνεται και στην επόμενη φωτογραφία:



Προγραμματισμός νέου ραντεβού από τον ασθενή

Εδώ ο ασθενής επιλέγει τις ημερομηνίες **Starting\_date** και **Ending\_date**, για να προβληθούν στην οθόνη όλα τα διαθέσιμα ραντεβού που υπάρχουν στο χρονικό διάστημα **[Starting\_date , Ending\_date]**. Επιπλέον, ο ασθενής μπορεί να περιορίσει την αναζήτηση των ραντεβού που πρόκειται να εμφανιστούν στην οθόνη με την βοήθεια της drop down λίστας. Έτσι, ο ασθενής μπορεί να επιλέξει να εμφανίζονται ραντεβού με ιατρούς συγκεκριμένης ειδικότητας (*Speciality*), είτε με ιατρούς που φέρουν συγκεκριμένο ονοματεπώνυμο (*Full name*) και AMKA (*Doctor AMKA*). Όταν ο ασθενής πατήσει το κουμπί “Search”, εμφανίζονται τα διαθέσιμα ραντεβού με βάση τις ρυθμίσεις που δηλώθηκαν προηγουμένως.

Όταν ο ασθενής πατήσει το κουμπί “Scheduled appointments”, θα μεταφερθεί αυτομάτως στην σελίδα *ScheduledAppointments.jsp*, η οποία εμφανίζεται στην ακόλουθη φωτογραφία:



Προβολή μελλοντικών (προγραμματισμένων) ραντεβού ενός ασθενή

Έτσι, κάθε ασθενής που συνδέεται στο σύστημα, μπορεί να δει όλα τα προγραμματισμένα (μελλοντικά) ραντεβού που έχει κλείσει. Πατώντας το κουμπί “Search”, εμφανίζονται όλα τα ραντεβού του ασθενή που υπάρχουν δηλωμένα στη βάση. Φυσικά, όπως και προηγουμένως, η αναζήτηση αυτή μπορεί να περιοριστεί ως προς τα αποτελέσματά της με την βοήθεια της Drop down λίστας.

Τέλος, το κουμπί “*Logout*” κλείνει την σύνδεση (session) και μεταφέρεται ο έλεγχος του προγράμματος στην σελίδα *login.jsp.*

# Η βάση δεδομένων

# 5 Βιβλιογραφικές πηγές