

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Πληροφορικής



Εργασία Μαθήματος «Προγραμματισμός στο διαδίκτυο και στον
παγκόσμιο ιστό»

| | |
|------------------------|--|
| ΑΣΚΗΣΗ 02 | ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΤΟ ΔΙΑΔΙΚΤΥΟ ΚΑΙ ΣΤΟΝ ΠΑΓΚΟΣΜΙΟ ΙΣΤΟ |
| ΦΟΙΤΗΤΕΣ ΣΤΗΝ ΕΡΓΑΣΙΑ: | Π19204 – Γεώργιος Σεϊμένης |
| | Π19064 – Ευστράτιος Καρκάνης |
| | Π19032 – Νικόλαος Γεωργιάδης |
| Ημερομηνία παράδοσης | 14/06/2021 |



Εκφώνηση της άσκησης

Στόχοι άσκησης: Εγκατάσταση application server και database server και μεταξύ τους διασύνδεση, δημιουργία web project, δημιουργία Βάσης Δεδομένων της εφαρμογής, υλοποίηση ορισμένων λειτουργιών.

Σε αυτή την άσκηση θα δημιουργήσετε ένα dynamic web project σε Java το οποίο θα αποτελέσει τον κορμό για την τελική εργασία. Σε αυτό το web project, θα δημιουργήσετε κλάσεις-servlet για να υλοποιήσετε τις λειτουργίες της εφαρμογής σας. Επιπλέον, μπορείτε να χρησιμοποιήσετε/αξιοποιήσετε κλάσεις που δημιουργήσατε στην προηγούμενη εργασία, με τις κατάλληλες προσθήκες και τροποποιήσεις.

Αναλυτικά Βήματα:

1 Εγκατάσταση και παραμετροποίηση application server και database server.

1.1 Εγκαταστήστε και παραμετροποιήστε τον Tomcat application server (εάν επιθυμείτε μπορείτε να χρησιμοποιήσετε Glashfish server ή άλλον αντίστοιχο) και το Σύστημα Διαχείρισης Βάσης Δεδομένων (mysql ή postgres). Η εγκατάσταση του application server να συνδεθεί με το περιβάλλον IDE που χρησιμοποιείτε (π.χ. Eclipse).

1.2 Δημιουργήστε την σύνδεση του application server με τον database server, χρησιμοποιώντας τον αντίστοιχο jdbc database connector για το σύστημα βάσης της επιλογής σας. Χρησιμοποιήστε τη σύνδεση του μοντέλου 3-tier.

2 Δημιουργία Βάσης Δεδομένων

2.1 Δημιουργήστε το Μοντέλο Οντοτήτων-Σχέσεων, το οποίο περιγράφει τη Βάση Δεδομένων που θα χρησιμοποιήσετε για την εφαρμογή σας. Ενδεικτικά θα περιλαμβάνει πίνακες όπως, Ασθενείς, Γιατροί, Διαχειριστές, Ραντεβού. Να περιλάβετε στο μοντέλο σας τις σχέσεις μεταξύ των πινάκων.



2.2 Με τη βοήθεια του Μοντέλου Οντοτήτων-Σχέσεων, να δημιουργήσετε και να εκκινήσετε τη βάση στον database server. Μπορείτε να χρησιμοποιήσετε οποιοδήποτε βοηθητικό εργαλείο για την εξαγωγή της βάσης από το μοντέλο (π.χ. mysql Workbench για mysql).

2.3 Εισάγετε εικονικά δεδομένα σε όλους τους πίνακες, λαμβάνοντας υπόψη τα εξωτερικά κλειδιά που πιθανώς έχουν οι πίνακες.

3 Δημιουργία web project και υλοποίηση λειτουργιών

3.1 Δημιουργήστε ένα Dynamic Web Project.

3.2 Δημιουργήστε ένα ή περισσότερα πακέτα κλάσεων, τα οποία θα περιλαμβάνουν τις βασικές κλάσεις που έχετε υλοποιήσει στην προηγούμενη άσκηση.

3.3 Δημιουργείστε ένα νέο πακέτο κλάσεων το οποίο θα περιλαμβάνει όλα τα servlet που θα χρησιμοποιήσετε στην εργασία (ενδεικτικά PatientServlet, DoctorServlet, AdminServlet). Στην συγκεκριμένη άσκηση θα υλοποιήσετε μόνο ένα μέρος από ένα από αυτά όπως αναφέρεται στο επόμενο βήμα.

3.4 Για το servlet το οποίο θα υλοποιεί τις λειτουργίες του Ασθενή (π.χ. PatientServlet) να υλοποιήσετε τις παρακάτω λειτουργίες:

3.4.1 Λειτουργία σύνδεσης (login) για τον Ασθενή.

3.4.2 Προβολή στοιχείων του ασθενή (μόνο τα στοιχεία του ασθενούς που έχει συνδεθεί).

3.4.3 Προβολή του ιστορικού των ραντεβού του συγκεκριμένου ασθενή.

3.5 Για την προβολή του αποτελέσματος κάθε μίας από τις παραπάνω ενέργειες, θα δημιουργείται μία δυναμική html σελίδα μέσω του servlet (ή συνδυασμό servlet και JSP). Δημιουργήστε επίσης τις απαραίτητες στατικές html σελίδες που απαιτούνται.



Οδηγίες:

- Ισχύουν οι ίδιες ομάδες και οι ίδιες οδηγίες με την προηγούμενη εργασία.
- Το συνολικό παραδοτέο θα περιλαμβάνει σε ένα συμπιεσμένο αρχείο: (α) το project, (β) τη βάση δεδομένων (.sql ή .mwb αρχείο εάν χρησιμοποιείτε το Workbench) και (γ) την τεκμηρίωση αντίστοιχα.



ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

| | | |
|-------|---|----|
| 1 | Γενική περιγραφή της εργασίας..... | 6 |
| 2 | Επαναχρησιμοποίηση του κώδικα | 6 |
| 3 | Κώδικας Προγράμματος | 7 |
| 3.1 | Back End..... | 7 |
| 3.1.1 | Κατασκευή των servlets | 7 |
| 3.2 | Front End..... | 8 |
| 3.2.1 | Αρχική σελίδα (index.jsp) | 8 |
| 3.2.2 | Σελίδα σύνδεσης χρήστη (login.jsp)..... | 17 |
| 3.2.3 | Η σελίδα σφάλματος (fail.jsp) | 20 |
| 3.2.4 | Βασικό μενού του ασθενή (patient-main-environment.jsp) .. | 22 |
| 3.2.5 | Ιστορικό των ραντεβού του ασθενή (appointment history)... | 26 |
| 4 | Η Βάση δεδομένων..... | 27 |
| 4.1 | Μοντέλο Οντοτήτων-Σχέσεων..... | 27 |
| 4.2 | Τα δεδομένα στους πίνακες | 30 |
| 5 | Παραδείγματα υλοποίησης | 30 |
| 6 | Βιβλιογραφικές Πηγές | 39 |



1 Γενική περιγραφή της εργασίας

Η 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 **Apache Tomcat 8.5** και η βάση δεδομένων **MySQL**. Επιπρόσθετα, έχουν προστεθεί πολλά καινούρια αρχεία (.java, .jsp, .css, .png) 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.

2 Επαναχρησιμοποίηση του κώδικα

Αρκετά χρήσιμη φάνηκε η προηγούμενη εργασία, σε οτιδήποτε αφορά την επαναχρησιμοποίηση των ήδη υπάρχουσών κλάσεων. Όλα τα στοιχεία τους ήταν ήδη φτιαγμένα, κι αφού έγιναν οι κατάλληλες τροποποιήσεις, χρησιμοποιήθηκαν παρέα με τα servlets. Το μόνο που άλλαξε σε αυτές τις κλάσεις ήταν η λειτουργικότητα μερικών μεθόδων (επίσης προστέθηκαν ακόμα μερικές βοηθητικές μέθοδοι). Στην προηγούμενη εργασία, το μόνο που χρειάστηκε ήταν να εκτυπώνονται στοιχεία στο τερματικό. Τώρα, στην δεύτερη εργασία, ο κώδικας και η λειτουργικότητα των μεθόδων έχει τροποποιηθεί καταλλήλως, ώστε το όλο project να μετατραπεί ορθά σε ένα Web Project.



3 Κώδικας Προγράμματος

3.1 Back End

3.1.1 Κατασκευή των servlets

Θα εκμεταλλευτούμε την δυνατότητα ανάκτησης στοιχείων από τα Servlet, ώστε να μεταδίδουμε τα στοιχεία προς τις κλάσεις που έχουμε φτιάξει. Μέσα στα Servlet, θα βλέπουμε τι έχει επιλέξει να κάνει ο χρήστης. Ανάλογα με την επιλογή του χρήστη θα μπορούμε να καλούμε την κατάλληλη μέθοδο. Ορισμένα στοιχεία τα οποία φάνηκαν χρήσιμα για την ανάπτυξη των servlets και των μεθόδων, έχουν αντληθεί από διάφορες ιστοσελίδες [1], [2] και βίντεο στο YouTube [3]. Επίσης, εξαιρετικά βοηθητικά φάνηκαν και τα εργαστηριακά παραδείγματα JAVA του μαθήματος [4].

- **Patient Servlet**

Όπως αναφέραμε και πριν, εκμεταλλευόμαστε τα servlet, διότι μπορούμε να ανακτήσουμε από την ιστοσελίδα στοιχεία και να τα περάσουμε ως ορίσματα σε συγκεκριμένες μεθόδους.

Στο Servlet του ασθενούς, έχουμε φροντίσει να αποθηκεύουμε σε μία μεταβλητή την «επιθυμία» του χρήστη. Αυτό επιτυγχάνεται, έχοντας κρυμμένες input ετικέτες σε HTML κώδικα που μέσω της γλώσσας JavaScript παίρνουν μία τιμή ανάλογη της «επιθυμίας» του χρήστη. Η τιμή αυτή περνάει στη συνέχεια στην προαναφερθείσα μεταβλητή. Η «επιθυμία» του χρήστη είναι, μάλιστα, ο ακρογωνιαίος λίθος για την λειτουργία, καθώς με αυτήν επιλέγεται η κατάλληλη μέθοδος από τις κλάσεις.

Οπότε, αν ο χρήστης έχει επιλέξει να κάνει είσοδο στο σύστημα, σημαίνει ότι έχει αποθηκευτεί στη μεταβλητή, κρυφά, η τιμή 6. Αν ο χρήστης έχει επιλέξει να κάνει εγγραφή, τότε αποθηκεύεται η τιμή 5, κοκ. Θα χρησιμοποιηθεί αυτό το μοτίβο, ώστε να υπάρχει συνοχή με τις ενέργειες, πράγμα που θα βοηθήσει και στο να γίνει πιο ευανάγνωστος ο κώδικας.

Ο κώδικας που έχει γραφεί, προς το παρόν, αναλαμβάνει την **εγγραφή**, την **είσοδο**, την **προβολή του ιστορικού ραντεβού για τον ασθενή** και την **προβολή των στοιχείων του ασθενή**. Αξίζει να σημειωθεί, ότι στην αρχή του Servlet, φτιάχνεται το Data Source της βάσης, αλλά δεν χρησιμοποιείται. Αντιθέτως, περνιέται ως όρισμα στις μεθόδους που καλεί το servlet, όπου χρειάζεται αλληλεπίδραση με τη βάση δεδομένων.



- **Doctor & Admin Servlet**

Αυτά τα δύο Servlets δημιουργήθηκαν ενδεικτικά, και δεν έχουν (προς το παρόν) κάποια λειτουργικότητα. Δεν έχει γραφεί καθόλου κώδικας στις μεθόδους των servlet, καθώς κάτι τέτοιο δεν έχει ζητηθεί στην 2η Εργασία. Ωστόσο, η δημιουργία τους συμβάλλει στο πρότυπο της Γενικής Εργασίας, όπου εκεί θα υλοποιηθούν πλήρως.

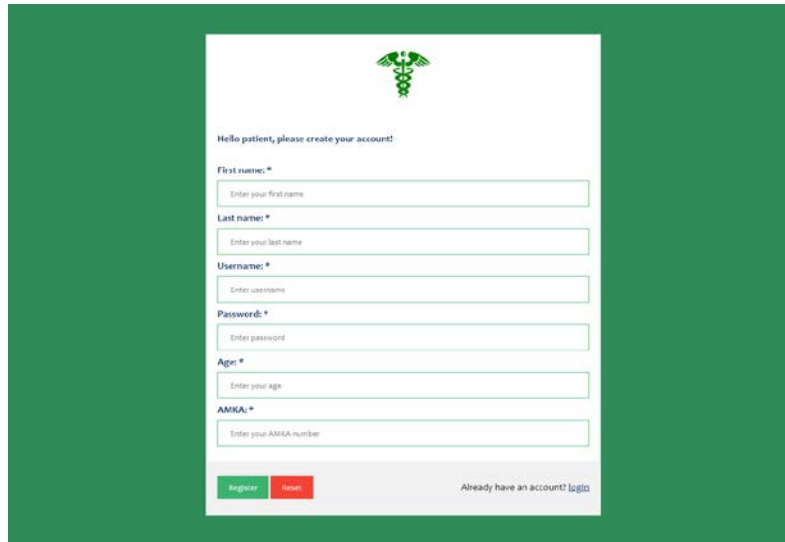
3.2 Front End

Ιδιαίτερη έμφαση έχει δοθεί στο σχεδιαστικό κομμάτι του Project, αν και αυτό δεν αποτελούσε κύριο στόχο της εργασίας. Για αυτόν τον σκοπό έχει γραφεί αρκετός κώδικας CSS και JavaScript μέσα σε αρχεία JSP όπως και σε ξεχωριστά αρχεία. Τέλος, είναι σημαντικό να δηλωθεί ότι όλες οι σελίδες είναι ομοιόμορφες σχεδιαστικά, δηλαδή έχει χρησιμοποιηθεί κοινός κώδικας CSS για κάθε σελίδα της εφαρμογής. Αυτό όχι μόνο προσδίδει μία αρμονία στο αποτέλεσμα, αλλά κάνει και ευκολότερη την σχεδίαση του όλου project.

Παρουσιάζονται τώρα οι κυριότερες σελίδες που συνθέτουν το γραφικό περιβάλλον της εφαρμογής:

3.2.1 Αρχική σελίδα (index.jsp)

Ξεκινώντας, η αρχική σελίδα που φορτώνεται όταν εκτελείται η εφαρμογή περιγράφεται στο αρχείο index.jsp. Ουσιαστικά πρόκειται για μία φόρμα εγγραφής, όπου ένας **ασθενής** μπορεί να δημιουργήσει έναν νέο λογαριασμό στο σύστημα διαχείρισης ιατρικών ραντεβού. Τα πεδία που ζητούνται να συμπληρώσει είναι τα ακόλουθα: First Name, Last Name, Username, Password, Age και ο μοναδικός αριθμός ΑΜΚΑ.



Αρχική σελίδα - φόρμα εγγραφής

Σε περίπτωση που ο **ασθενής** διαθέτει ήδη κάποιο λογαριασμό στο σύστημα, μπορεί να πατήσει τον σύνδεσμο “Login”, όπου και θα μεταβεί σε μία νέα σελίδα, για να συνδεθεί, την σελίδα login.jsp που περιγράφεται παρακάτω.

Όσον αφορά το σχεδιαστικό κομμάτι της αρχικής σελίδας, έχει υλοποιηθεί κώδικας CSS που ρυθμίζει το φόντο, την γραμματοσειρά, τα χρώματα των κουμπιών, τα μεγέθη και χρώματα των πεδίων της φόρμας, το μέγεθος της εικόνας και πολλά άλλα [5]. Για την εύκολη ανάγνωση του κώδικα, υπάρχουν σχετικά σχόλια που περιγράφουν τι ακριβώς μορφοποιεί ο κώδικας CSS. Ο κώδικας αυτός βρίσκεται εντός των ετικετών <style>....</style> μέσα στο αρχείο index.jsp και είναι ο ακόλουθος:

```
<style>

    /* set border and background color to the form */
    form
    {
        border: 3px solid whitesmoke;
        background-color: white ;
    }

    /* the whole page has the same font */
    *
    {
        font-family:candara;
    }

```



```
/* style rules for the body of the page */
body
{
    background-color: seagreen;
    margin: 0;
    position: absolute;
    top: 50%;
    left: 50%;
    -ms-transform: translate(-50%, -50%);
    transform: translate(-50%, -50%);
}

/* style rules for the inputs boxes in the form */
input[type=text],
input[type=password]
{
    width: 100%;
    padding: 12px 20px;
    margin: 8px 0;
    display: inline-block;
    box-sizing: border-box;
    border: 1px solid mediumseagreen;
}

/* set a style for the buttons */
button
{
    background-color: mediumseagreen;
    color: white;
    padding: 10px 18px;
    margin: 8px 0;
    border: none;
    cursor: pointer;
    width: auto;
}

/* set a hover effect for the button */
button:hover
{
    opacity: 0.8;
}

/* set extra style for the cancel button */
.cancelbtn
{
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}

/* center the display image inside the container */
.imgcontainer
{
    text-align: center;
    margin: 24px 0 12px 0;
}
```



```
/* set image properties */
img.avatar
{
    width: 16%;
    border-radius: 70%;
}

/* set padding to the container class */
.container
{
    padding: 16px;
}

/* set the forgot password text */
span.psw
{
    float: right;
    padding-top: 16px;
}

/* set styles for span and cancel button on small screens */
@media screen and (max-width: 300px)
{
    span.psw
    {
        display: block;
        float: none;
    }

    .cancelbtn
    {
        width: 100%;
    }
}

/* style rules when hyperlinks are pressed */
a:visited
{
    color: #012A6C;
}

/* style the hyperlinks in the nav section */
a
{
    font-size: 16px;
    color: #012A6C;
}

/* style rules for the error message box */
#error_message
{
    background: #fe8b8e;
    text-align: center;
    font-size: 16px;
    transition: all 0.5s ease;
    width: 97%;
}
```



```
margin: 8px 0;  
display: inline-block;  
box-sizing: border-box;  
color: white;  
font-weight: bold;  
padding: 0px;  
}  
</style>
```

Αξίζει να σημειωθεί ότι εκτελείται και έλεγχος για την **ορθή** συμπλήρωση των στοιχείων από το χρήστη κάθε φορά που γίνεται κλικ στο κουμπί «Register» της φόρμας. Αυτό γίνεται από την πλευρά του πελάτη (client side), χρησιμοποιώντας κώδικα JavaScript [6], ο οποίος βρίσκεται εντός του αρχείου index.jsp και εντός των ετικετών <script>...</script> . Εάν η φόρμα περιέχει λάθη, τότε η υποβολή των στοιχείων δεν γίνεται αποδεκτή, ενώ επίσης εμφανίζεται σχετικό μήνυμα στον χρήστη. Σε αντίθετη περίπτωση, η φόρμα υποβάλλεται κανονικά και επιπλέον εμφανίζεται νέα σελίδα JSP (register-success.jsp) με το μήνυμα της επιβεβαίωσης εγγραφής.

Ο κώδικας JavaScript μαζί με ελέγχους στη βάση στο επίπεδο server, ορίζουν τους ακόλουθους περιορισμούς για κάθε πεδίο της φόρμας εγγραφής:

- Το πεδίο **First Name** ξεκινάει από κεφαλαίο λατινικό γράμμα και περιλαμβάνει τουλάχιστον ένα πεζό λατινικό γράμμα αμέσως μετά το κεφαλαίο γράμμα.
- Το πεδίο **Last Name** ξεκινάει από κεφαλαίο λατινικό γράμμα και περιλαμβάνει τουλάχιστον ένα πεζό λατινικό γράμμα αμέσως μετά το κεφαλαίο γράμμα.
- Το πεδίο **Username** δεν μπορεί να περιλαμβάνει μόνο κενούς χαρακτήρες (spaces). Είναι επίσης **μοναδικό** για την κατηγορία των ασθενών.
- Το πεδίο **password** περιλαμβάνει τουλάχιστον τέσσερις (4) χαρακτήρες.
- Το πεδίο **Age** περιλαμβάνει μόνο φυσικούς αριθμούς στο διάστημα [1,119].
- Το πεδίο **AMKA** περιλαμβάνει ακριβώς 11 νούμερα. Είναι επίσης **μοναδικό για όλους τους χρήστες**.



Ο κώδικας JavaScript είναι ο ακόλουθος που ελέγχει τους παραπάνω περιορισμούς δίδεται παρακάτω:

```
<script>

// this function is used for form's validation on client's side
function validation()
{
    // get the values of inputs that user gave in the form
    var first_name = document.getElementById("fn").value;
    var last_name = document.getElementById("ln").value;
    var username = document.getElementById("username").value;
    var password = document.getElementById("password").value;
    var age = document.getElementById("age").value;
    var AMKA = document.getElementById("AMKA").value;
    var error_message = document.getElementById("error_message");

    var text; // set a new variable

// use a new variable for storing the errors during validation
    error_message.style.padding = "10px";

    // validation check begins

    // check first name
    if (!/^[A-Z][a-z]+$/.test(first_name))
    {
        text = "First name includes at least two letters and should
begin with a capital letter!";
        error_message.innerHTML = text;
        return false;
    }

    // check last name
    if (!/^[A-Z][a-z]+$/.test(last_name))
    {
        text = "Last name includes at least two letters and should
begin with a capital letter!";
        error_message.innerHTML = text;
        return false;
    }

    // check username
    if (!/^\S$/.test(username))
    {
        text = "Username should not consist of only space
characters!";
        error_message.innerHTML = text;
        return false;
    }
}
```



```
// check password
if (password.length < 4)
{
    text = "Password consists of at least 4 characters!";
    error_message.innerHTML = text;
    return false;
}

// check age
if (!/^[0-9]+$/.test(age) || age > 119 || age <= 0)
{
    text = "Age consist of positive integer numbers and is lower
than or equal to 119!";
    error_message.innerHTML = text;
    return false;
}

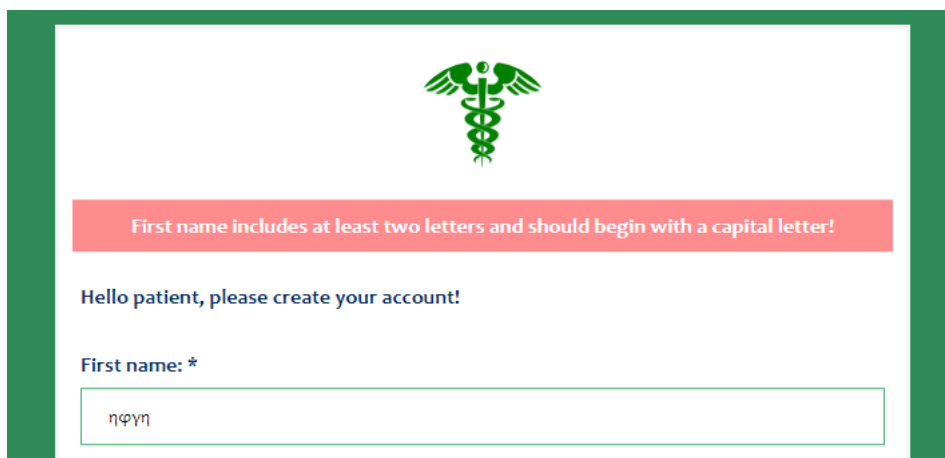
// check AMKA
if (!/^\d{11}$/.test(AMKA))
{
    text = "AMKA should be a 11-digit number!";
    error_message.innerHTML = text;
    return false;
}

return true; // everything is fine, no errors occurred
}

</script>
```

Επιπλέον, **δεν επιτρέπεται** κανένα πεδίο στη φόρμα να είναι κενό. Αυτό γίνεται με το χαρακτηριστικό «required» που υπάρχει γραμμένο στον ορισμό HTML καθενός πεδίου της φόρμας.

Ακολουθούν ενδεικτικά screenshots που δείχνουν τι γίνεται σε περίπτωση που ο χρήστης συμπληρώσει λανθασμένα την φόρμα (εικόνα 1) και τι γίνεται όταν η εγγραφή ολοκληρώνεται με επιτυχία (εικόνα 2).



The image shows a registration form with a green border. At the top center is a green caduceus icon. Below it is a red rectangular box containing the text: "First name includes at least two letters and should begin with a capital letter!". Underneath this is the text "Hello patient, please create your account!". Then, there is a label "First name: *" followed by a text input field. The input field contains the Greek text "ηφγγη".

Εικόνα 1: Μήνυμα λάθους που εμφανίζεται στο χρήστη, έπειτα από το πάτημα του κουμπιού "Register"



Εικόνα 2: Σελίδα που εμφανίζεται στο χρήστη, έπειτα από μία επιτυχημένη εγγραφή



Όσον αφορά τη σελίδα που φαίνεται στην παραπάνω εικόνα, δηλώνει στο χρήστη ότι η εγγραφή του έχει γίνει σωστά. Ο κώδικας CSS που μορφοποιεί τη σελίδα αυτή βρίσκεται στο αρχείο `register-success.jsp` εντός των ετικετών `<style>...</style>` και δίδεται παρακάτω:

```
<style>

    /* the whole page has the same font */
    *
    {
        font-family: candara;
    }

    /* style rules for the body of the page */
    body
    {
        background-color: seagreen;
        margin: 0;
        position: absolute;
        top: 50%;
        left: 50%;
        -ms-transform: translate(-50%, -50%);
        transform: translate(-50%, -50%);
    }

    /* style rules for the image */
    img
    {
        width: 400px;
        height: 400px;
    }

    /* style rules for the menu at the bottom of the page */
    .navbar
    {
        font-size: 16px;
        bottom: 0;
        text-align: center;
        background-color: #f1f1f1;
        width: 100%;
        height: 20px;
    }

</style>
```




Αξίζει να σημειωθεί ότι η σελίδα αυτή (register-success.jsp) κάνει ανακατεύθυνση προς τη σελίδα login.jsp μετά από πέντε (5) δευτερόλεπτα (από τη στιγμή που έχει φορτωθεί η σελίδα register-success.jsp). Η σελίδα login.jsp περιγράφεται παρακάτω.

3.2.2 Σελίδα σύνδεσης χρήστη (login.jsp)

Στο ίδιο πνεύμα με την σχεδίαση της αρχικής σελίδας που προαναφέρθηκε, κυμαίνεται και η σχεδίαση της σελίδας login.jsp. Όπως εύκολα γίνεται αντιληπτό, η σελίδα αυτή περιλαμβάνει μία φόρμα με δύο πεδία (Username και Password) τα οποία **υποχρεωτικά** συμπληρώνει ο χρήστης για να **συνδεθεί** στην εφαρμογή. Επιπρόσθετα, σύμφωνα με την εκφώνηση της εργασίας, δικαίωμα σύνδεσης έχουν και οι τρεις κατηγορίες χρηστών, δηλαδή οι Ιατροί, οι Ασθενείς και οι Διαχειριστές. Για αυτό τον λόγο στην φόρμα υπάρχει ειδικό πεδίο (radio buttons) που επιλέγει ο χρήστης ώστε να δηλωθεί σε ποια από τις τρεις προαναφερθείσες κατηγορίες ανήκει.

Η σελίδα login.jsp



Ο χρήστης συνδέεται στην εφαρμογή εφόσον έχει κάνει **ήδη** την εγγραφή του σε προηγούμενη χρονική στιγμή. Σε περίπτωση που δεν διαθέτει ήδη λογαριασμό στο σύστημα, μπορεί να πατήσει τον σύνδεσμο «Register» για να μεταβεί στην σελίδα index.jsp και να εγγραφεί. Προσοχή, δικαίωμα εγγραφής έχουν **μόνο** οι χρήστες που είναι ασθενείς.

Όσον αφορά το σχεδιαστικό κομμάτι της σελίδας login.jsp, ο κώδικας CSS είναι ο ακόλουθος [5]:

```
<style>

/* set border and background color to the form */
form
{
    border: 3px solid whitesmoke;
    background-color: white ;
}

/* the whole page has the same font */
*
{
    font-family:candara;
}

/* style rules for the body of the page */
body
{
    background-color: seagreen;
    margin: 0;
    position: absolute;
    top: 50%;
    left: 50%;
    -ms-transform: translate(-50%, -50%);
    transform: translate(-50%, -50%);
}

/* set styles for the inputs of the login form */
input[type=text],
input[type=password]
{
    width: 100%;
    padding: 12px 20px;
    margin: 8px 0;
    display: inline-block;
    box-sizing: border-box;
    border:1px solid mediumseagreen;
}
```



```
/* set a style for the buttons */
button
{
    background-color: mediumseagreen;
    color: white;
    padding: 10px 18px;
    margin: 8px 0;
    border: none;
    cursor: pointer;
    width: auto;
}

/* set a hover effect for the button */
button:hover
{
    opacity: 0.8;
}

/* set extra style for the cancel button */
.cancelbtn
{
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}

/* center the display image inside the container */
.imgcontainer
{
    text-align: center;
    margin: 24px 0 12px 0;
}

/* set image properties */
img.avatar
{
    width: 16%;
    border-radius: 70%;
}

/* set padding to the container */
.container
{
    padding: 16px;
}

/* set the forgot password text */
span.psw
{
    float: right;
    padding-top: 16px;
}
```



```
/* style rules for the radio buttons */
input[type=radio] {
    height: 16px;
    width: 15px;
}

/* set styles for span and cancel button on small screens */
@media screen and (max-width: 300px)
{
    span.psw
    {
        display: block;
        float: none;
    }

    .cancelbtn
    {
        width: 100%;
    }
}

/* style rules when hyperlinks are pressed */
a:visited
{
    color: #012A6C;
}

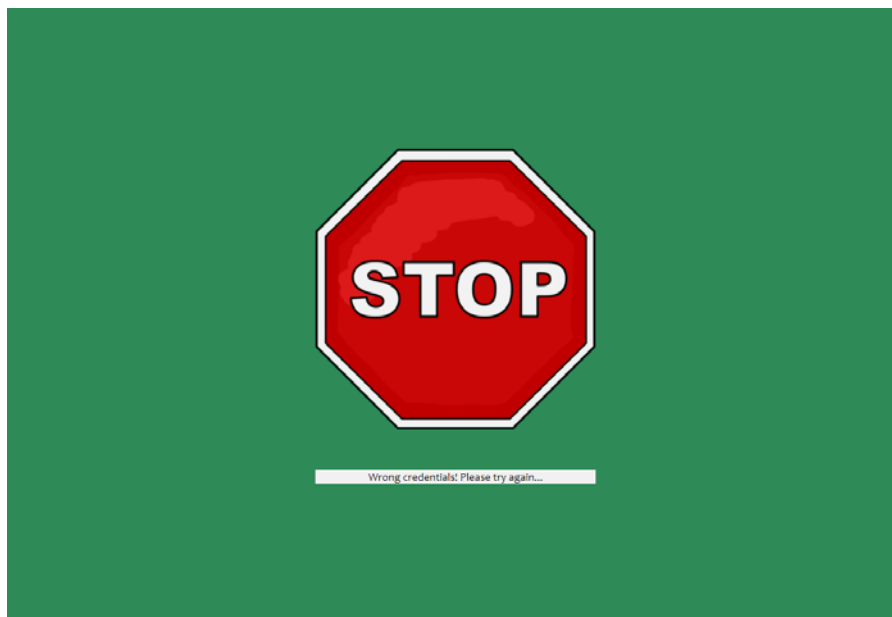
/* style the hyperlinks in the nav section */
a
{
    font-size: 16px;
    color: #012A6C;
}

</style>
```

Τέλος, όσον αφορά την φόρμα σύνδεσης, δεν γίνεται κάποιος έλεγχος εγκυρότητας των δεδομένων με την βοήθεια της JavaScript. Τέτοιου είδους έλεγχοι γίνονται από την πλευρά του εξυπηρετητή (server).

3.2.3 Η σελίδα σφάλματος (fail.jsp)

Η σελίδα αυτή (fail.jsp) δίδεται ως απάντηση του server στον χρήστη, όταν τα στοιχεία που έχει υποβάλλει στην φόρμα σύνδεσης (Login Form) είναι λανθασμένα. Αφού φορτωθεί πλήρως η σελίδα αυτή, γίνεται ανακατεύθυνση προς την σελίδα login.jsp σε πέρας τεσσάρων (4) δευτερολέπτων. Η σελίδα αυτή είναι απλή και φαίνεται στην επόμενη φωτογραφία:



Η σελίδα fail.jsp

Τέλος, ο κώδικας CSS που μορφοποιεί την σελίδα αυτή βρίσκεται εντός του αρχείου fail.jsp και είναι ο ακόλουθος:

```
<style>

/* the whole page has the same font */
*
{
    font-family:candara;
}

/* style rules for the body of the page */
body
{
    background-color: seagreen;
    margin: 0;
    position: absolute;
    top: 50%;
    left: 50%;
    -ms-transform: translate(-50%, -50%);
    transform: translate(-50%, -50%);
}

/* style rules for the image */
img
{
    width:400px;
    height:400px;
}

/* style rules for the menu at the bottom of the page */
```

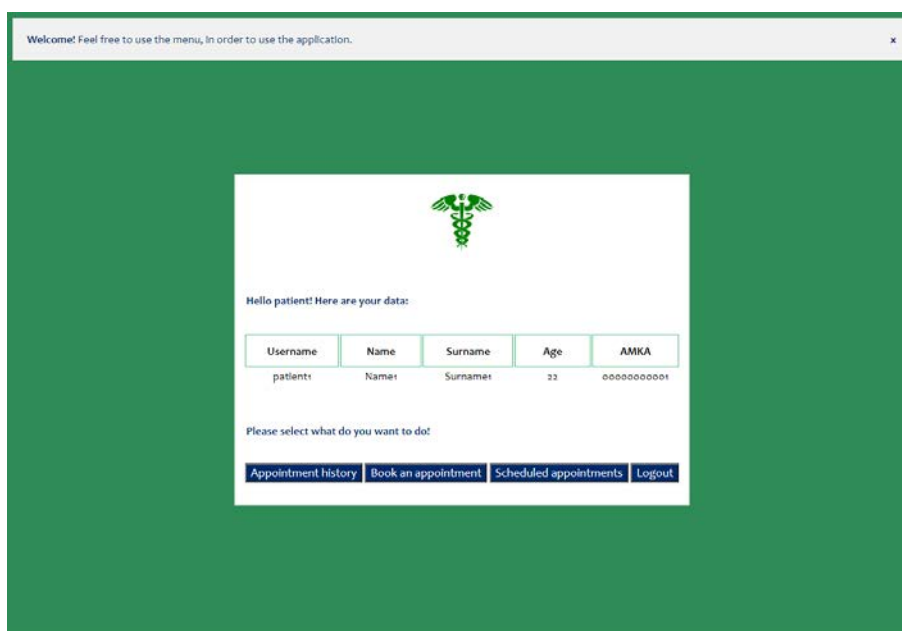


```
.navbar
{
    font-size: 16px;
    bottom: 0;
    text-align: center;
    background-color: #f1f1f1;
    width: 100%;
    height: 20px;
}

</style>
```

3.2.4 Βασικό μενού του ασθενή (patient-main-environment.jsp)

Όταν ο ασθενής κάνει επιτυχημένη την σύνδεσή του, τότε εμφανίζεται σε αυτόν το βασικό μενού (η σελίδα patient-main-environment.jsp) όπου μπορεί να εκτελέσει διάφορες λειτουργίες. Η σελίδα αυτή φαίνεται στην παρακάτω φωτογραφία:



Η σελίδα patient-main-environment.jsp



Πιο συγκεκριμένα, η σελίδα αυτή αποτελεί το βασικό περιβάλλον διαχείρισης της εφαρμογής για τον ασθενή. Όταν ο ασθενής συνδέεται, μπορεί να δει τα στοιχεία του (Username, Name, Surname, Age, AMKA) που εμφανίζονται σε έναν πίνακα. Τα στοιχεία αυτά υπάρχουν στην βάση δεδομένων που χρησιμοποιεί η εφαρμογή. Επιπλέον, μπορεί να πατήσει ένα από τα τέσσερα κουμπιά που εμφανίζονται στην σελίδα για να κάνει μία από τις ακόλουθες τέσσερις λειτουργίες:

- Να δει το ιστορικό με τα ραντεβού του (Appointment history)
- Να κλείσει ένα νέο ραντεβού (Book an appointment)
- Να δει τα προγραμματισμένα του ραντεβού (Scheduled appointments)
- Να κάνει αποσύνδεση (logout)

Από τις παραπάνω τέσσερις (4) λειτουργίες, έχει υλοποιηθεί μονάχα η πρώτη, δηλαδή το ιστορικό των ραντεβού του ασθενή που έχει συνδεθεί.

Επιπρόσθετα, ο κώδικας CSS που μορφοποιεί αυτό την σελίδα του project βρίσκεται στην σελίδα patient-main-environment.jsp εντός των ετικετών <style>...</style> και είναι ο ακόλουθος:

```
<style>

    /* style rules for the buttons */
    #buttons
    {
        font-size:17px;
        text-align:center;
        color:white;
        background-color: #012A6C;
    }

    /* style rules when hyperlinks are pressed */
    a:visited
    {
        color: #012A6C;
    }

    /* style the hyperlinks in the nav section */
    a
    {
        font-size:14px;
        color: #012A6C;
    }

    /* the whole page has the same font */
    *
    {
```



```
        font-family: candara;
    }

    /* style rules for the body of the page */
    body
    {
        background-color: seagreen;
    }

    /* style rules for the logo image that is appeared in the body of the
web page */
    .imgcontainer
    {
        text-align: center;
        margin: 24px 0 12px 0;
    }

    /* set image properties */
    img.avatar
    {
        width: 16%;
        border-radius: 70%;
    }

    /* set style rules for the container class */
    .container
    {
        padding: 10px;
    }

    /* style rules about the alert message box */
    .alert
    {
        padding: 20px;
        background-color: #f1f1f1;
        color: #012A6C;
        margin-bottom: 15px;
    }

    /* The close button of the message at the upper of the web page */
    .closebtn
    {
        margin-left: 15px;
        color: #012A6C;
        font-weight: bold;
        float: right;
        font-size: 22px;
        line-height: 20px;
        cursor: pointer;
        transition: 0.3s;
    }

    /* style rules for the article section of the web page */
    article
    {
        margin: 0;
```




```
        position: absolute;
        top: 50%;
        left: 50%;
        -ms-transform: translate(-50%, -50%);
        transform: translate(-50%, -50%);
    }

    /* set border and background color of form */
    form
    {
        border: 3px solid whitesmoke;
        background-color: white ;
    }

    /* when moving the mouse over the close button, make it black */
    .closebtn:hover
    {
        color: black;
    }

    /* style rules for the table that is showed to the user */
    table
    {
        align: center;
    }

    /* style rules for every table heading of the table */
    th
    {
        width:120px;
        height:20px;
        text-align:center;
        color: black;
        border:1px solid mediumseagreen;
        padding: 12px 20px;
    }

    /* style rules for every table row of the table */
    tr
    {
        text-align: center;
        padding: 16px;
    }
</style>
```



3.2.5 Ιστορικό των ραντεβού του ασθενή (appointment history)

Η σελίδα αυτή χρησιμοποιείται για να προβάλλει στον ασθενή όλα τα ραντεβού, τα οποία αυτός είχε κάνει στο παρελθόν. Φυσικά, η πληροφορία αυτή πηγάζει από την βάση δεδομένων που χρησιμοποιεί η εφαρμογή. Επιπλέον, στην σελίδα αυτή υπάρχει και σύνδεσμος προς τη σελίδα `patient_main_environment.jsp`.

Σχεδιαστικά, η σελίδα είναι φτιαγμένη με ανάλογο κώδικα CSS με τις προηγούμενες προαναφερθείσες σελίδες της διαδικτυακής εφαρμογής. Τα προηγούμενα ραντεβού εμφανίζονται το ένα κάτω από το άλλο σε έναν πίνακα. Για κάθε ραντεβού παρέχονται οι εξής πληροφορίες: Date, Start time, End time, Patient AMKA και Doctor AMKA.

Ένα παράδειγμα της σελίδας αυτής φαίνεται και στην επόμενη εικόνα:

| Date | Start time | End time | Patient AMKA | Doctor AMKA |
|------------|------------|----------|--------------|-------------|
| 08-09-2017 | 12:00:00 | 12:30:00 | 00000000001 | 10000000001 |
| 13-08-2018 | 18:00:00 | 18:30:00 | 00000000001 | 10000000003 |
| 07-06-2021 | 17:00:00 | 17:30:00 | 00000000001 | 10000000003 |
| 01-06-2021 | 19:00:00 | 19:30:00 | 00000000001 | 20000000001 |
| 07-06-2021 | 10:00:00 | 10:30:00 | 00000000001 | 20000000001 |

Choose a category to search appointments by:

Insert the doctor's AMKA/appointment date:

Do you want to go back? Click [here](#)

Το ιστορικό των ραντεβού ενός ασθενή



4 Η Βάση δεδομένων

4.1 Μοντέλο Οντοτήτων-Σχέσεων

Το μοντέλο οντοτήτων-σχέσεων της εργασίας αποτελεί το αρχείο `app_db.mwb(my sql)`. Αποτελείται από τους πίνακες patient, doctor, appointment και admin.

- Ο patient έχει τα εξής χαρακτηριστικά:

- patientAMKA VARCHAR(11)
- username VARCHAR(45)
- hashedpassword VARCHAR(45)
- name VARCHAR(45)
- surname VARCHAR(45)
- salt VARCHAR(45)
- age INT

Υποψήφια κλειδιά του patient είναι το **AMKA** και το **username** του, καθώς κανένα από αυτά δεν μπορεί να παρουσιάζεται με την ίδια τιμή σε δύο διαφορετικές εγγραφές του πίνακα. Το AMKA και το username ενός ασθενή, τον προσδιορίζουν μοναδικά.

- Ο admin έχει τα εξής χαρακτηριστικά:

- username VARCHAR(45)
- hashedpassword VARCHAR(45)
- salt VARCHAR(45)
- age INT
- name VARCHAR(45)
- surname VARCHAR(45)

Πρωτεύον κλειδί εδώ, είναι το **username** ενός admin.



- Ο doctor έχει τα εξής χαρακτηριστικά:

```
doctorAMKA VARCHAR(11)
username VARCHAR(45)
hashedpassword VARCHAR(45)
name VARCHAR(45)
surname VARCHAR(45)
specialty VARCHAR(45)
ADMIN_username VARCHAR(45)
salt VARCHAR(45)
age INT
```

Υποψήφια κλειδιά είναι το AMKA και το **username** του doctor. Επίσης το **username** του admin αποτελεί ξένο κλειδί που αναφέρεται στον πίνακα admin.

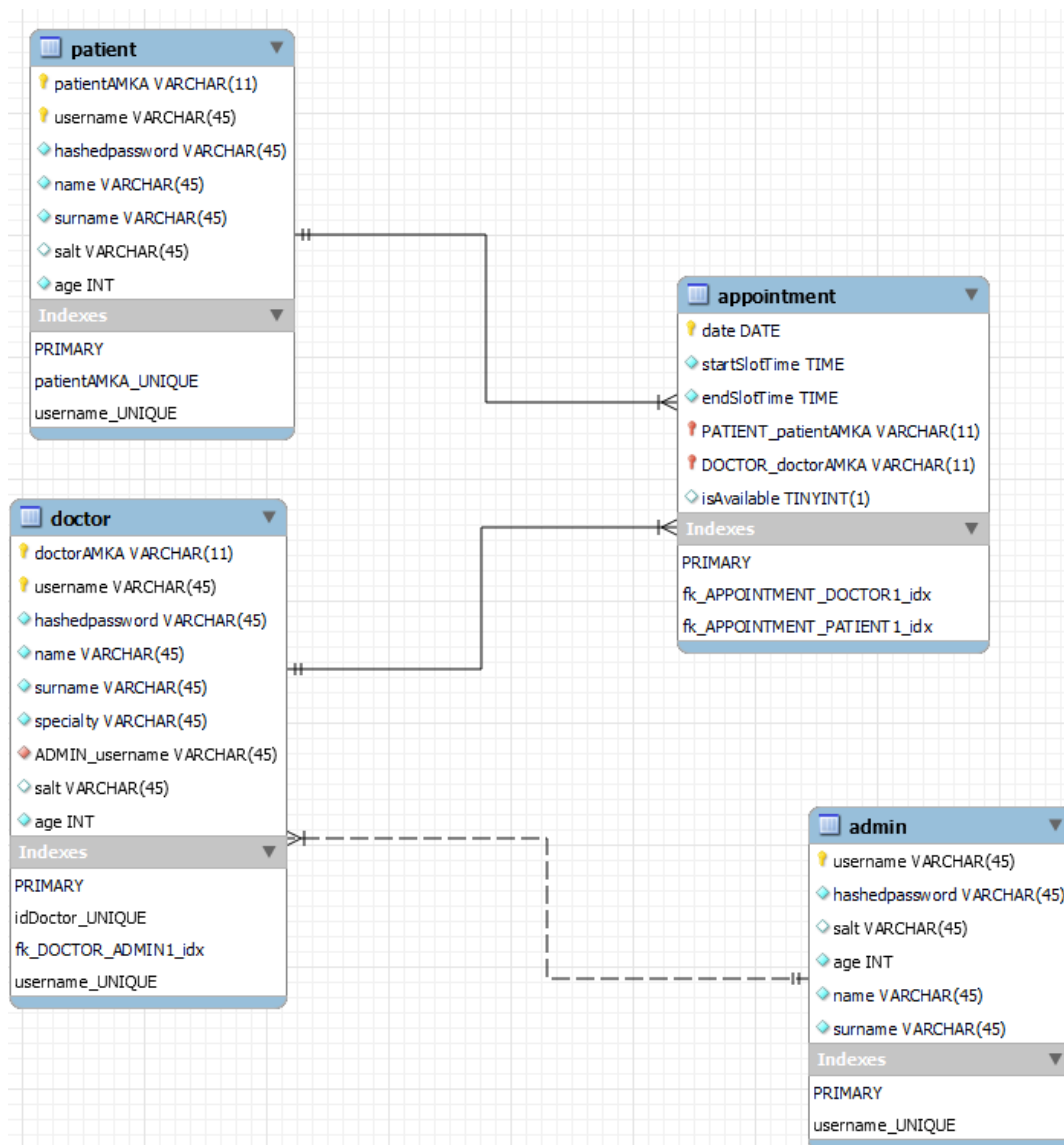
- Ο appointment αποτελείται από τα εξής χαρακτηριστικά:

```
date DATE
startSlotTime TIME
endSlotTime TIME
PATIENT_patientAMKA VARCHAR(11)
DOCTOR_doctorAMKA VARCHAR(11)
isAvailable TINYINT(1)
```

Πρωτεύον κλειδί είναι η τριάδα **date**, **patientAMKA** και **DoctorAMKA**. Έχουμε επιλέξει, ένα ραντεβού να καθορίζεται μοναδικά από αυτήν την τριάδα, πράγμα το οποίο σημαίνει ότι δεν μπορούν να κλείνονται δύο ραντεβού με τον ίδιο ασθενή και τον ίδιο ιατρό, την ίδια μέρα. Επίσης, τα AMKA αποτελούν ξένα κλειδιά των patient και doctor.



Μία ολοκληρωμένη εικόνα του μοντέλου οντοτήτων-σχέσεων δίνεται παρακάτω:



- Ένας ιατρός μπορεί να έχει πολλά ραντεβού
- Ένας ασθενής μπορεί να έχει πολλά ραντεβού
- Ένας admin μπορεί να έχει βάλει στο σύστημα πολλούς ιατρούς



4.2 Τα δεδομένα στους πίνακες

Όλες οι εγγραφές των πινάκων βρίσκονται στον φάκελο `sql statements` ο οποίος μπορεί να γίνει `import` στο `mysql workbench`. Επίσης, σχετικά με την χρήση των δεδομένων σε αυτή την εργασία, έχουμε παραλείψει το `salt` σε όλους τους πίνακες που περιέχουν `password`, καθώς δεν πραγματοποιείται το `hashing` ακόμα (το `password` αποθηκεύεται μη κρυπτογραφημένο στη βάση). Τέλος, έχουμε αφήσει τη στήλη `isAvailable` του πίνακα `appointment` σε αυτή την εργασία (πιθανόν να αφαιρεθεί στην επόμενη εργασία) και έχουμε θέσει σε όλες τις γραμμές της 0, καθώς έχουμε προσθέσει μόνο κλεισμένα ραντεβού.

5 Παραδείγματα υλοποίησης

Σελίδα εκκίνησης (`index.jsp`)

Ο ασθενής συμπληρώνει τα στοιχεία του για την εγγραφή:

Hello patient, please create your account!

First name: *

Nikos

Last name: *

Georgiadis

Username: *

nick geo

Password: *

••••

Age: *

20

AMKA: *

12345678910

Παράδειγμα συμπλήρωσης φόρμας(σωστά στοιχεία)



Στη συνέχεια, αφού πατήσει το κουμπί της εγγραφής,

Register

εάν τα στοιχεία του είναι σωστά, γίνεται ανακατεύθυνση στην σελίδα register-success.jsp και μετά από μερικά δευτερόλεπτα γίνεται ξανα, αυτόματα, μια ανακατεύθυνση στη σελίδα εισόδου χρήστη (login.jsp). Σε αυτή τη φάση, έχουν αποθηκευτεί κανονικά τα στοιχεία του χρήστη στη βάση.



Επιτυχία εγγραφής

| | | | | | | |
|-------------|----------|------|-------|------------|------|----|
| 12345678900 | 2rf | 1234 | Rf | Ed | NULL | 42 |
| 12345678910 | nick geo | 1234 | Nikos | Georgiadis | NULL | 20 |

Η εγγραφή στη βάση(η στήλη που είναι null περιέχει το salt)



Username: *

Enter Username

Password: *

Enter Password

Category: * ☐ Admin ☐ Patient ☐ Doctor

Ανακατεύθυνση στη σελίδα εισόδου χρήστη

Ακολουθούν μερικά παραδείγματα λάθους εισαγωγής στοιχείων:

First name includes at least two letters and should begin with a capital letter!

Hello patient, please create your account!

First name: *

georgios

Last name: *

Seimenis

Username: *

george

Password: *

••••

Age: *

20

AMKA: *

11122233344

Περίπτωση λάθους στο First name(ξεκινάει με πεζό λατινικό γράμμα)



Username should not consist of only space characters!

Hello patient, please create your account!

First name: *

Georgios

Last name: *

Seimenis

Username: *

■

Password: *

••••

Age: *

20

AMKA: *

11122233344

Περίπτωση λάθους στο username(αποτελείται μόνο από κενούς χαρακτήρες)

Something went wrong!

This username/AMKA is already taken!

Περίπτωση λάθους στο username/AMKA(ο χρήστης έδωσε ένα υπάρχον AMKA ή username)



Username: *

Please fill out this field.

Category: * ☐ Admin ☐ Patient ☐ Doctor

Don't have an account yet? [register](#)

Ο χρήστης άφησε κενό κάποιο πεδίο

Στη σελίδα **index.jsp**, υπάρχει και ένας σύνδεσμος που μας μεταφέρει απευθείας στην **login.jsp** εάν ο χρήστης θέλει να κάνει απευθείας login:

Already have an account? [login](#)

Login σύνδεσμος

Σελίδα εισόδου χρήστη(login.jsp)

Εδώ, έχουμε τη δυνατότητα να επιλέξουμε την κατηγορία χρήστη που κάνει login μέσω των radio buttons(προς το παρόν δεν έχουν κάποια λειτουργικότητα διότι μόνο οι ασθενείς μπορούν να κάνουν login)

Username: *

Password: *

Category: * ☐ Admin ☒ Patient ☐ Doctor

Don't have an account yet? [register](#)

Login ενός patient

Όπως και στην index.jsp έτσι και εδώ, υπάρχει ένας αντίστοιχος σύνδεσμος για την εγγραφή(register):



Don't have an account yet? [register](#)

Register σύνδεσμος

Στη περίπτωση λάθους εισαγωγής στοιχείων(ο χρήστης δε βρέθηκε στη βάση), μεταφερόμαστε στην **fail.jsp** και μετά από λίγο, πίσω στην **login.jsp**:



Fail.jsp

Σελίδα περιβάλλοντος ασθενή(patient_main_environment.jsp)

Ο ασθενής που έκανε προηγουμένως login, έχει πρόσβαση στα στοιχεία του και σε μερικές ενέργειες:

Hello patient! Here are your data:

| Username | Name | Surname | Age | AMKA |
|----------|-------|------------|-----|-------------|
| nick geo | Nikos | Georgiadis | 20 | 12345678910 |

Τα στοιχεία του ασθενή

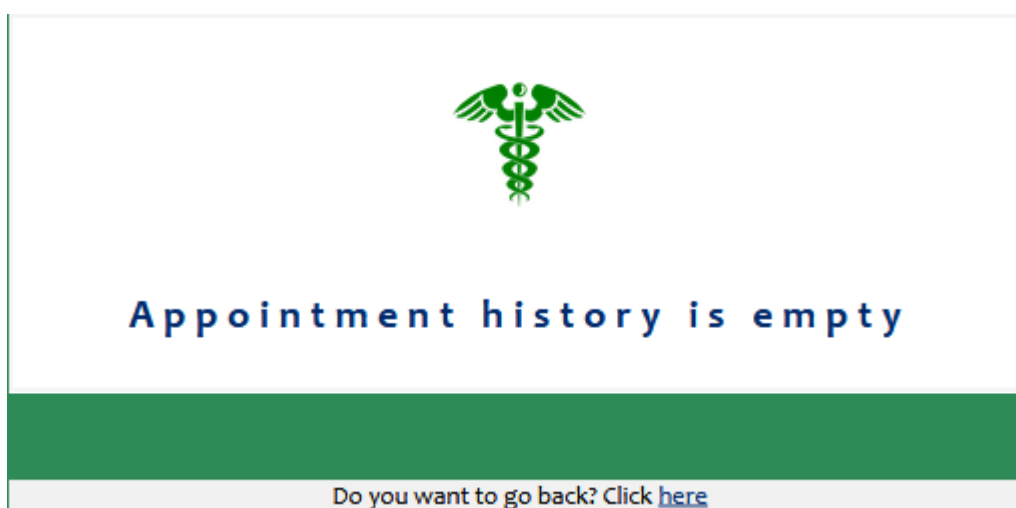


Please select what do you want to do!

[Appointment history](#) [Book an appointment](#) [Scheduled appointments](#) [Logout](#)

Οι ενέργειες του ασθενή

Η μόνη λειτουργική για αυτή την εργασία, είναι η προβολή ραντεβού(Appointment history).Ο συγκεκριμένος ασθενής έχει άδειο ιστορικό:



Άδειο ιστορικό ασθενή

Ένα παράδειγμα ασθενή με γεμάτο ιστορικό, είναι αυτό του patient1.Κάνοντας εγγραφή ως patient1 έχουμε τα ακόλουθα ραντεβού:

| Date | Start time | End time | Patient AMKA | Doctor AMKA |
|------------|------------|----------|--------------|-------------|
| 08-09-2017 | 12:00:00 | 12:30:00 | 00000000001 | 10000000001 |
| 13-08-2018 | 18:00:00 | 18:30:00 | 00000000001 | 10000000003 |
| 07-06-2021 | 17:00:00 | 17:30:00 | 00000000001 | 10000000003 |
| 01-06-2021 | 19:00:00 | 19:30:00 | 00000000001 | 20000000001 |
| 07-06-2021 | 10:00:00 | 10:30:00 | 00000000001 | 20000000001 |

Choose a category to search appointments by:

Insert the doctor's AMKA/appointment date:

Ιστορικό ραντεβού patient1



Το ιστορικό ραντεβού ενός ασθενή αποτελείται μόνο από αυτά με ημερομηνία προγενέστερη της σημερινής ή αυτά με τη σημερινή ημερομηνία αλλά με το endSlotTime μικρότερο της στιγμιαίας ώρας:

| date | startSlotTime | endSlotTime | PATIENT_patientAMKA | DOCTOR_doctorAMKA | isAvailable |
|------------|---------------|-------------|---------------------|-------------------|-------------|
| 2021-07-01 | 18:00:00 | 18:30:00 | 00000000001 | 10000000003 | 0 |
| 2021-06-13 | 19:00:00 | 19:30:00 | 00000000001 | 10000000001 | 0 |
| 2021-06-13 | 11:00:00 | 11:30:00 | 00000000001 | 10000000002 | 0 |
| 2021-06-07 | 17:00:00 | 17:30:00 | 00000000001 | 10000000003 | 0 |
| 2021-06-07 | 10:00:00 | 10:30:00 | 00000000001 | 20000000001 | 0 |
| 2021-06-01 | 19:00:00 | 19:30:00 | 00000000001 | 20000000001 | 0 |
| 2018-08-13 | 18:00:00 | 18:30:00 | 00000000001 | 10000000003 | 0 |
| 2017-09-08 | 12:00:00 | 12:30:00 | 00000000001 | 10000000001 | 0 |

Όλα τα ραντεβού του patient1

| Date | Start time | End time | Patient AMKA | Doctor AMKA |
|------------|------------|----------|--------------|-------------|
| 08-09-2017 | 12:00:00 | 12:30:00 | 00000000001 | 10000000001 |
| 13-06-2021 | 11:00:00 | 11:30:00 | 00000000001 | 10000000002 |
| 13-08-2018 | 18:00:00 | 18:30:00 | 00000000001 | 10000000003 |
| 07-06-2021 | 17:00:00 | 17:30:00 | 00000000001 | 10000000003 |
| 01-06-2021 | 19:00:00 | 19:30:00 | 00000000001 | 20000000001 |
| 07-06-2021 | 10:00:00 | 10:30:00 | 00000000001 | 20000000001 |

Το ιστορικό ραντεβού του patient1 στις 13/6/2021 ώρα 16:12

Παρατηρούμε πως υπάρχουν δύο ραντεβού στις 13/6/2021, ένα που τελειώνει στις 19:30 και ένα στις 11:30. Στο ιστορικό, εμφανίζεται μόνο αυτό που τελειώνει στις 11:30 αφού η ώρα προβολής του ιστορικού είναι 16:12.

Ο ασθενής, έχει το δικαίωμα να κάνει προβολή μόνο συγκεκριμένων ραντεβού με φίλτρα(βάση Doctor AMKA και Date). Επιλέγουμε κάποιο φίλτρο από το option box και στην συνέχεια γράφουμε την τιμή του φίλτρου(πχ για doctor AMKA):

its by:

late:

Επιλογή φίλτρου "Doctor AMKA"



Choose a category to search appointments by: Doctor AMKA

Insert the doctor's AMKA/appointment date: 10000000003

Search

Αναζήτηση ιστορικού ραντεβού με AMKA ιατρού 10000000003

| Date | Start time | End time | Patient AMKA | Doctor AMKA |
|------------|------------|----------|--------------|-------------|
| 13-08-2018 | 18:00:00 | 18:30:00 | 00000000001 | 10000000003 |
| 07-06-2021 | 17:00:00 | 17:30:00 | 00000000001 | 10000000003 |

Αποτελέσματα

Τέλος, όπως και στα υπόλοιπα πεδία έτσι και εδώ, εμφανίζονται τα ανάλογα μηνύματα αν κάτι πάει στραβά:



No results found for Doctor AMKA 10000000004

Δεν βρέθηκαν αποτελέσματα για κάποιο AMKA



Invalid Date format

Ο ασθενής έβαλε λάθος μορφή ημερομηνίας



6 Βιβλιογραφικές Πηγές

- [1] “W3Schools Online Web Tutorials.” <https://www.w3schools.com/> (accessed Jun. 13, 2021).
- [2] “Stack Overflow - Where Developers Learn, Share, & Build Careers.” <https://stackoverflow.com/> (accessed Jun. 13, 2021).
- [3] “Servlet & JSP Tutorial | Full Course - YouTube.” <https://www.youtube.com/watch?v=OuBUUkQfBYM> (accessed Jun. 13, 2021).
- [4] “GUNet2 eClass - Τμήμα Πληροφορικής | ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΤΟ ΔΙΑΔΙΚΤΥΟ ΚΑΙ ΤΟΝ ΠΑΓΚΟΣΜΙΟ ΙΣΤΟ | Έγγραφα.” <https://gunet2.cs.unipi.gr/modules/document/document.php?course=TMB117&openDir=/201102161747563cexy8tj> (accessed Jun. 14, 2021).
- [5] “HTML | Responsive Modal Login Form - GeeksforGeeks.” <https://www.geeksforgeeks.org/html-responsive-modal-login-form/> (accessed Jun. 13, 2021).
- [6] “Contact Us Form Validation Using Javascript | Form Validation In Javascript - YouTube.” <https://www.youtube.com/watch?v=WY4rvU4ImgE> (accessed Jun. 13, 2021).