ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής



Εργασία Μαθήματος «Προγραμματισμός στο διαδίκτυο και στον παγκόσμιο ιστό»

|  |  |
| --- | --- |
|  | Άσκηση 2 |
| Όνομα φοιτητή – Αρ. Μητρώου  (όλων σε περίπτωση ομαδικής εργασίας) | ΕΛΕΥΘΕΡΙΟΣ ΚΟΝΤΟΥΡΗΣ - Π19077 |
| ΚΩΝΣΤΑΝΤΙΝΟΣ ΚΑΛΟΓΕΡΟΠΟΥΛΟΣ – Π19057 |
| ΜΙΧΑΗ ΣΤΥΛΙΑΝΙΔΗΣ – Π19165 |
|  |
| Ημερομηνία παράδοσης |  |

**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 postgress). Η εγκατάσταση του application server να συνδεθεί με το περιβάλλον IDE που χρησιμοποιείτε (π.χ. Eclipse).

1.2 Δημιουργήσετε 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.

* **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.

* **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.

[1]

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

Πίνακας Περιεχομένων

[1 Application Server και Database Server 5](#_Toc74251952)

[2 Βιβλιογραφικές Πηγές 6](#_Toc74251953)

1. Εισαγωγή

Για την δημιουργία της εφαρμογής χρησιμοποιήθηκε το IntelliJ IDEA 2021.1 (Ultimate Edition). Σε κάθε ενότητα δίνονται ακριβής οδηγίες για το πως να ρυθμιστούν σωστά τα εργαλεία που χρησιμοποιούνται.

1. Servers
   1. Application Server

Για Application Server χρησιμοποιήθηκε ο Tomcat 9. Αρχικά εγκαταστάθηκε και έπειτα με τη χρήση του IntelliJ έγινε παραμετροποίηση του. Η σύνδεση με την εφαρμογή έγινε μέσο του παραθύρου ‘Run/Debug Configurations’, προσθέτοντας στη λίστα με τα configurations το Tomcat Local Server .

Όμως, για να τρέξει τελικά η εφαρμογή μέσα από το IDEA πρέπει να δημιουργηθεί ένα artifact. Στην προκείμενη περίπτωση δημιουργήθηκε το ‘DoctorPro:war exploded’ artifact μέσω του πεδίου Before Launch Tasks στο ίδιο παράθυρο.

Πρέπει να αναφερθεί, επίσης, ότι στην αρχή έγινε χρήση της έκδοσης 10 του Tomcat αλλά καθιστούσε την εφαρμογή μη λειτουργική και εν τέλη αποφασίστηκε να χρησιμοποιηθεί η 9.

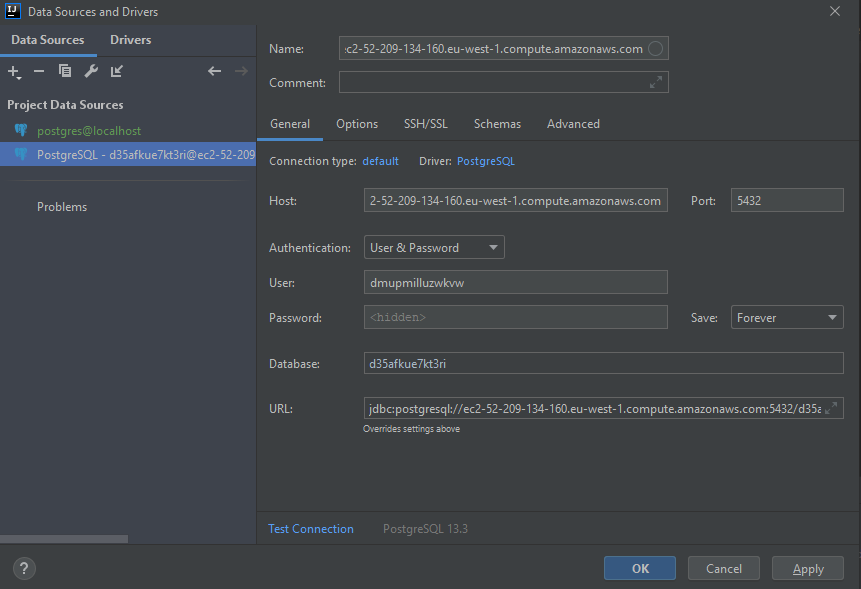
* 1. Database Server

Για Database Server έγινε χρήση του PostgresSQL. Η σύνδεση του με τον Tomcat 9 έγινε με τη χρήση του JDBC driver, μετακινώντας το αρχείο ‘postgresql-42.2.20.jar’, που κατεβάσαμε από την ιστοσελίδα ‘jdbc.postgresql.org’, στον κατάλογο ./apache-tomcat-9.0.46/lib.

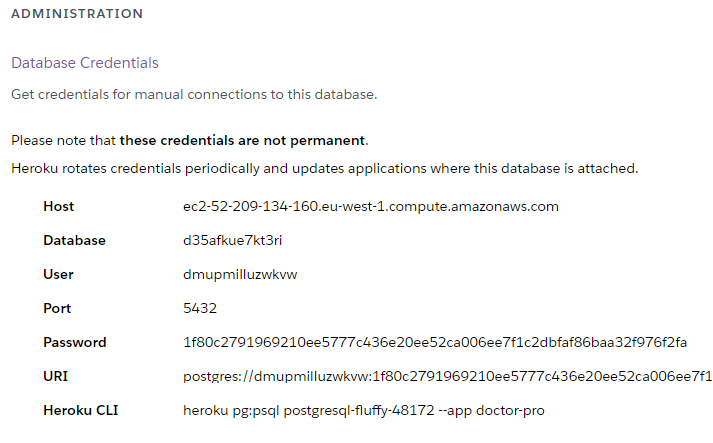
Εξαιτίας της ανάγκης να χρησιμοποιείται η ίδια βάση και λόγω ευκολίας στη διαχείριση, προχωρήσαμε σε hosting τη βάσης σε πραγματικό server. Για να την επίτευξη αυτής της ενέργειας χρησιμοποιήθηκε η υπηρεσία Heroku. Η πρόσβαση και απόκτηση των βασικών πακέτων έγινε μέσο του GitHub Student Developer Pack, όπου το Heroku αποτελεί ένα από τα προνόμια του.

Η σύνδεση με το IntelliJ έγινε μέσο του παραθύρου View/Tool-Windows/Databases, όπου προστέθηκε ως Data Source η PostgreSQL και έπειτα χρησιμοποιώντας τα πιστοποιητικά που εκδόθηκαν από το Heroku, ρυθμίστηκε η βάση. Μπορείτε να βρείτε τα στοιχεία αυτά στο αρχείο ‘database-creds.txt’.

Παρακάτω παρουσιάζεται το παράθυρο όπου ρυθμίζεται η Βάση στο IntelliJ.

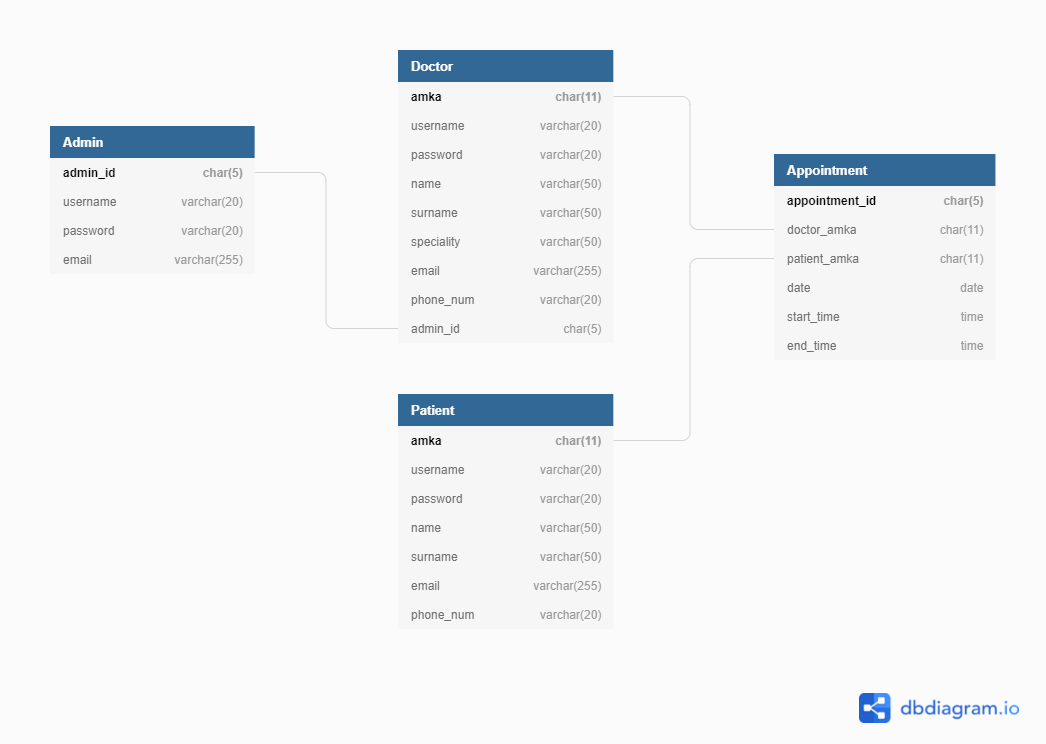


Επίσης, παρακάτω φαίνονται τα πιστοποιητικά που δίνει το Heroku ώστε μια εφαρμογή να συνδεθεί στη Βάση.



Το .sql αρχείο μπορείτε να το βρείτε στον κατάλογο της εργασίας.

1. Βάση Δεδομένων – Μοντέλο Οντοτήτων-Σχέσεων



Το Μοντέλο Οντοτήτων-Σχέσεων περιέχει τους βασικούς πίνακες της εφαρμογής. Πιο συγκεκριμένα:

* **Doctor:** Ο πίνακας των γιατρών περιέχει τον ΑΜΚΑ του γιατρού, ο οποίος αποτελεί πρωτεύον κλειδί του πίνακα και είναι ένα σύνολο από 11 χαρακτήρες σταθερού μήκους. Τα υπόλοιπα πεδία, εκτός του admin\_id, είναι πεδία χαρακτήρων μεταβλητού μήκους, το καθένα με διαφορετικό ανώτατο όριο. Τέλος, έχουμε το πεδίο admin\_id, το οποίο σχετίζεται με τον πίνακα Admin και δηλώνει το ID του διαχειριστή που πρόσθεσε τον εκάστοτε γιατρό.
* **Patient:** Ο πίνακας των ασθενών φέρει πολλές ομοιότητες με τον πίνακα των Γιατρών. Η μόνη διαφορά έγκειται στην μη ύπαρξη των πεδίων της ειδικότητας (speciality) και του admin\_id, καθώς ο κάθε ασθενής είναι ο ίδιος υπεύθυνος για την εγγραφή του στο σύστημα.
* **Appointment:** Ο πίνακας αυτός αποθηκεύει ραντεβού. Κάθε ραντεβού έχει έναν μοναδικό 5ψήφιο αριθμό για αναγνωριστικό, τους ΑΜΚΑ του γιατρού και του ασθενή, την ημέρα (χωρίς ώρα) που θα διεξαχθεί το ραντεβού καθώς και την ώρα (χωρίς ζώνη ) που θα ξεκινήσει και θα τελειώσει. Σχετίζεται με τους πίνακες των Γιατρών και των Ασθενών καθώς από εκεί παίρνει τους ΑΜΚΑ.
* **Admin:** Ο πίνακας των διαχειριστών είναι υπεύθυνος για την αποθήκευση των διαχειριστών. Για κάθε διαχειριστή αποθηκεύει έναν μοναδικό 5ψήφιο αριθμό που λειτουργεί ως πρωτεύον κλειδί, ένα username, έναν κωδικό και το email του. Σχετίζεται με τον πίνακα των Γιατρών καθώς οι διαχειριστές είναι υπεύθυνοι για την προσθήκη, διαγραφή και επεξεργασία των γιατρών.

Η Βάση δημιουργήθηκε από την αρχή στον database server και χρησιμοποιώντας το Mockaroo, το οποίο αποτελεί μια δωρεάν εφαρμογή που τρέχει στον φυλλομετρητή και δημιουργεί εικονικά δεδομένα για την βάση. Στον πίνακα των ραντεβού χρησιμοποιήθηκαν στοιχεία που υπήρχαν ήδη από τους υπόλοιπους πίνακες. Για λόγους ευκολίας, στο αρχείο ‘test\_users.txt’ θα βρείτε ασθενείς που έχουν περασμένα ραντεβού.

1. Εκκίνηση και Παραμετροποίηση του Web Project

Η δημιουργία του DoctorProSystemCare έγινε μέσα από το παράθυρο ‘New Project’, επιλέγοντας το ‘Java Enterprise’ από τη λίστα. Ως template επιλέχθηκε το ‘Web Application’ ώστε να γνωρίζει το IntelliJ το είδος της εφαρμογής και να δημιουργήσει αυτόματα τους κατάλληλους καταλόγους και αρχεία.

Από την Εργασία 1 έγινε εισαγωγή των Model τάξεων Patient, Doctor, Appointment, Admin και User, οι οποίες τοποθετήθηκαν στο πακέτο ‘com.WebFlexers.models’.

Στο πακέτο ‘com.WebFlexers.servlets’ δημιουργήθηκαν τα servlets που θα χρησιμοποιηθούν για την εργασία. Ενδεικτικά τα PatientServlet, DoctorServlet και AdminServlet.

1. Λειτουργίες

Στην συγκεκριμένη εργασία έγινε υλοποίηση κάποιων σημαντικών λειτουργιών του Ασθενή. Πιο συγκεκριμένα:

* **Login:** Πατώντας το κουμπί ‘Login’ στη φόρμα σύνδεσης της κεντρικής σελίδας, στέλνεται ένα μήνυμα post στο PatientServlet. Αυτό επιτυγχάνεται με τα χαρακτηριστικά method και action της ετικέτας form. Στο action ορίζουμε το πως θα σταλούν τα δεδομένα και στο method τον προορισμό.

Έπειτα, το PatientServlet λαμβάνει το μήνυμα της φόρμας και αναλαμβάνει αρχικά να πιστοποιήσει τον χρήστη και έπειτα να τον συνδέσει στην σελίδα.

Η πιστοποίηση γίνεται μέσω της στατικής τάξης validatePatient(), η οποία βρίσκεται στην τάξη του μοντέλου Patient. Αυτό επρόκειτο να αλλάξει στην τρίτη εργασία εξαιτίας του ότι περιπλέκει πολύ τον κώδικα και τις λειτουργίες.

* **Εμφάνιση στοιχείων:** Αφού εκτελεστεί το query στην βάση και πιστοποιηθεί ο χρήστης τότε μέσω της, επίσης στατικής, τάξης getPatientDetails() λαμβάνονται σε έναν πίνακα τα στοιχεία του χρήστη. Έπειτα δημιουργείται ένα καινούργιο session και στέλνονται στην σελίδα ‘profile.jsp’ μέσω της setAttribute() οι πληροφορίες που λήφθηκαν, ώστε να εμφανιστούν.
* **Προβολή ιστορικού ραντεβού:** Για την λήψη των ραντεβού του ασθενή που συνδέθηκε, χρησιμοποιήθηκε η, επίσης, στατική συνάρτηση getAppointmentsHistory() από το μοντέλο του Patient, όπου χρησιμοποιώντας τον ΑΜΚΑ γίνεται ένα query στον πίνακα των ραντεβού και αν βρεθούν ραντεβού εμφανίζονται σε σχετικό πεδίο στην σελίδα ‘profile.jsp’. Η προβολή γίνεται με τη χρήση μιας επανάληψης foreach στην λίστα τον ραντεβού που έστειλε η setAttribute() η οποία με τη σειρά της έλαβε τα δεδομένα από την προαναφερόμενη συνάρτηση.

1. Βιβλιογραφικές Πηγές

*Εδώ θα περιλάβετε τις πιθανές πηγές που έχετε χρησιμοποιήσει (βιβλία, διαδίκτυο, παραδείγματα κώδικα κτλ). Προσοχή! Εργασίες στις οποίες έχετε χρησιμοποιήσει κώδικα από πηγές τις οποίες δεν έχετε αναφέρει ρητά, θεωρούνται (και είναι) αντιγραφή!*

1. **Menezes, Vandstone.** *Handbook of Applied Cryptography.* New York : CRC PRess, 1996.