ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

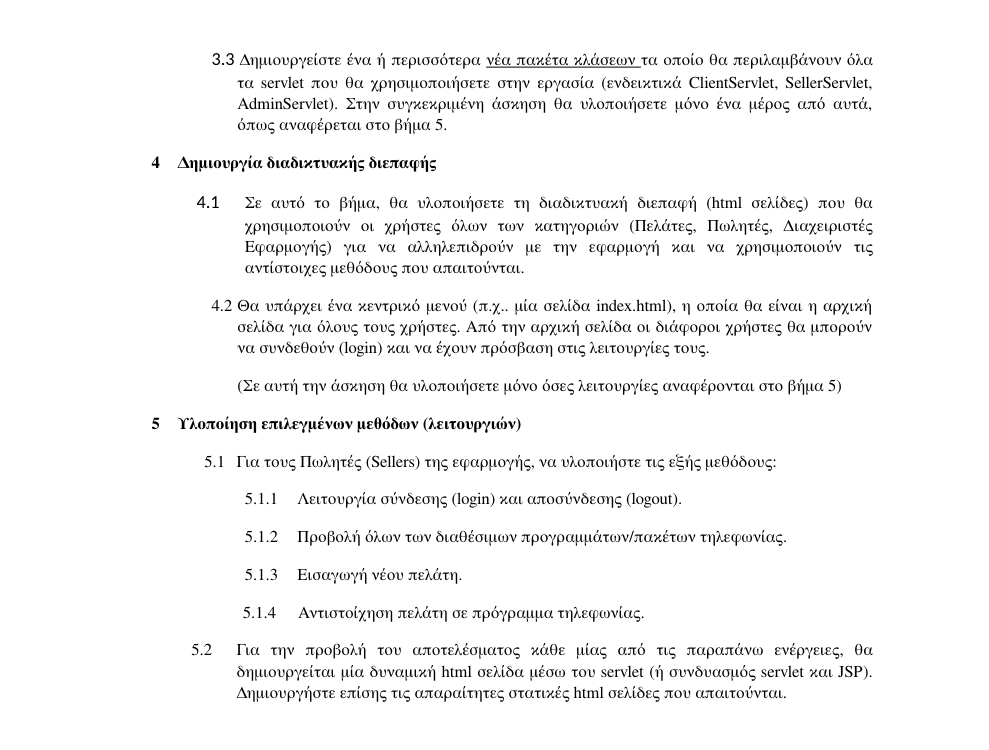
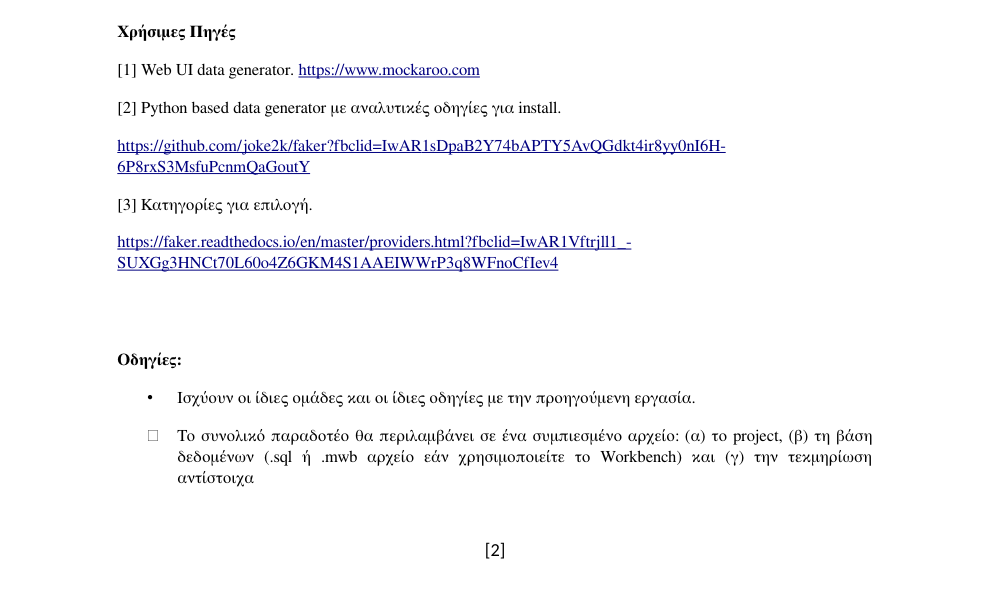
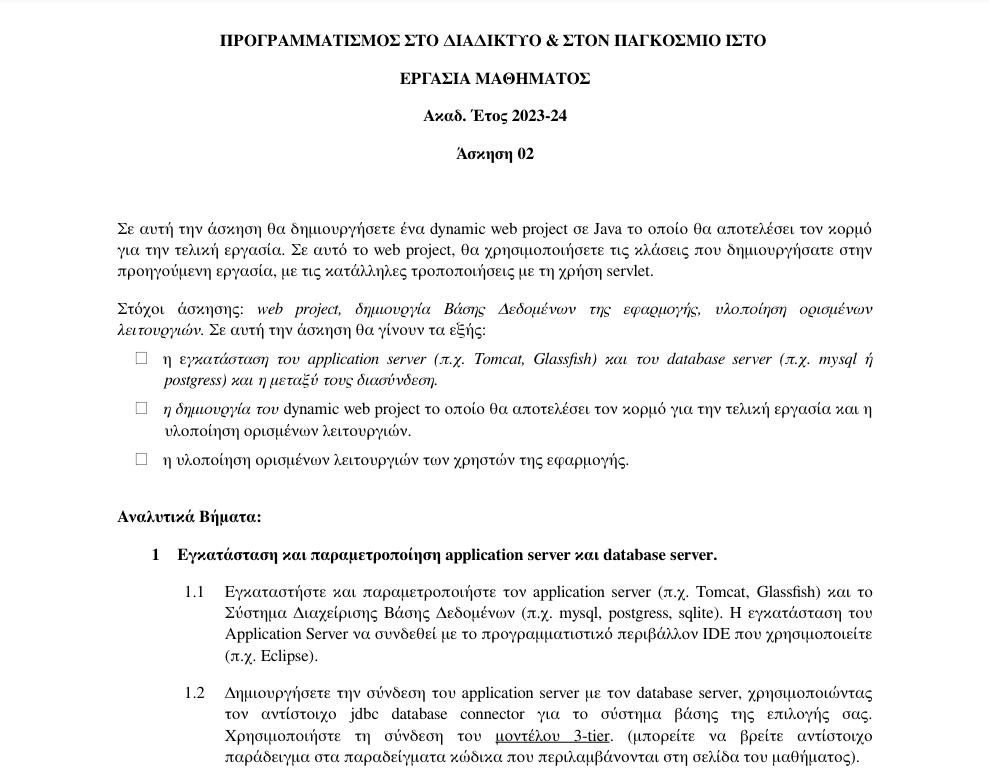
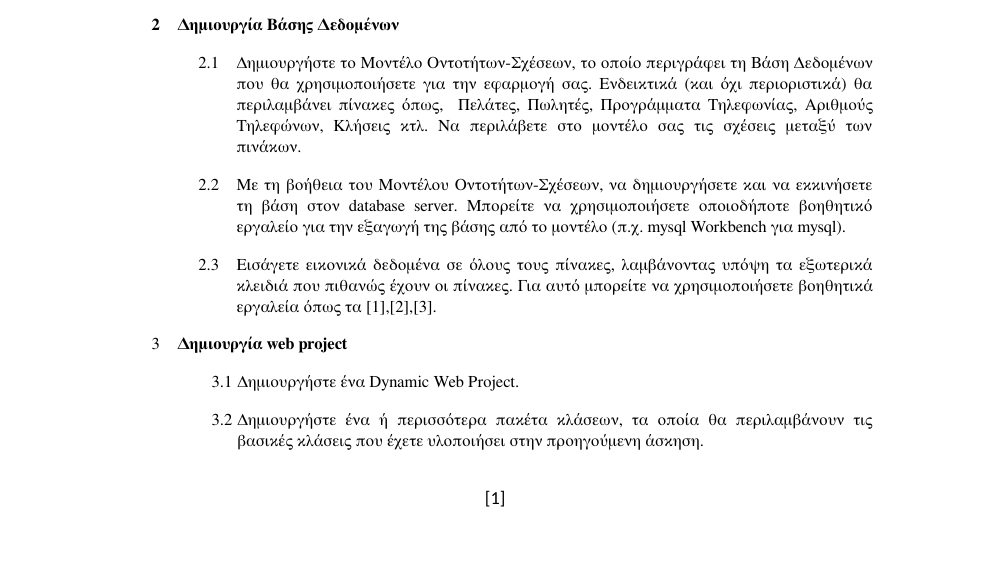
Τμήμα Πληροφορικής



Εργασία Μαθήματος «Προγραμματισμός στο διαδίκτυο και στον παγκόσμιο ιστό»

|  |  |
| --- | --- |
| 2Η ΑΣΚΗΣΗ | Δημιουργία βασικών κλάσεων με βασική λειτουργικότητα της εφαρμογής |
| Όνομα φοιτητή – Αρ. Μητρώου | ΔΗΜΗΤΡΙΟΣ ΛΑΖΑΝΑΣ Π22082 |
| YURIY OSIPYAN ΓΟΥΡΙΪ ΟΣΙΠΙΑΝ Π22125 |
| ΑΝΤΩΝΙΟΣ ΤΣΑΛΜΠΟΥΡΗΣ Π22272 |
| ΝΑΠΑΤ ΡΟΤΖΑΝΑΝΟΝΤ Π22150 |
| Ημερομηνία παράδοσης | 17 Ιουλίου 2024 |

**Εκφώνηση της άσκησης**



ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

[1. Τεχνικές προδιαγραφές 5](#_Toc169557052)

[1.1 Περιβάλλον ανάπτυξης 5](#_Toc169557053)

[1.2 Application Server 5](#_Toc169557054)

[1.3 Βάση δεδομένων και O.R.M. (Object Relational Mapping) 5](#_Toc169557055)

[1.4 Maven 5](#_Toc169557056)

[1.5 Διάγραμμα κλάσεων 6](#_Toc169557057)

[1.6 Δομή του Project 6](#_Toc169557058)

[1.7 Οδηγίες εκτέλεσης 8](#_Toc169557059)

[2. Διαδικτυακή διεπαφή (για τον Seller) 8](#_Toc169557060)

[2.1 Αρχική σελίδα (*index.html*) 8](#_Toc169557061)

[2.2 Log in Page (*loginPage.jsp*) 9](#_Toc169557062)

[2.3 Μενού του Seller (*menu.jsp*) 9](#_Toc169557063)

[2.4 Προσθήκη Client (*AddClient.jsp*) 10](#_Toc169557064)

[2.5 Ανάθεση προγράμματος τηλεφωνίας σε Client (*ClientDetails.jsp*) 10](#_Toc169557065)

[2.6 Προβολή διαθέσιμων προγραμμάτων κινητής (*ProgramsList.jsp*) 11](#_Toc169557066)

[3. Κώδικας - Λειτουργικότητες (Servlets) 11](#_Toc169557067)

[3.1 Log in (LoginController.java) 11](#_Toc169557068)

[3.2 Log out (LogoutController.java) 12](#_Toc169557069)

[3.3 Register (RegisterController.java) 13](#_Toc169557070)

[3.4 Seller Servlet (SellerController.java) 14](#_Toc169557071)

[3.5 Client Servlet (ClientController.java) 15](#_Toc169557072)

[3.6 Phone Servlet (PhoneControler.java) 16](#_Toc169557073)

[3.7 Program Servlet (ProgramController.java) 17](#_Toc169557074)

[4. Βιβλιογραφικές Πηγές 19](#_Toc169557075)



1. Τεχνικές προδιαγραφές
   1. Περιβάλλον ανάπτυξης

Για την ανάπτυξη και εκτέλεση του προγράμματος, χρησιμοποιήθηκε το ***Visual Studio Code*** ως κύριο περιβάλλον ανάπτυξης.

Οι αναλυτικές οδηγίες για την εγκατάσταση των απαραίτητων εργαλείων και την προετοιμασία του περιβάλλοντος ανάπτυξης βρίσκονται στο αρχείο ***README*** που συνοδεύει τον κώδικα του προγράμματος.

* 1. Application Server

Ο Application Server που επιλέχθηκε ως servlet container για την εκτέλεση της εφαρμογής είναι ο *Apache Tomcat 9.*

* 1. Βάση δεδομένων και O.R.M. (Object Relational Mapping)

Το Σύστημα Διαχείρισης Βάσης Δεδομένων που επιλέχθηκε ήταν η ***SQLite*** λόγω της απλότητας, της μηδενικής διαμόρφωσης και της ενσωματωμένης φύσης της. Παράλληλα, επιλέχθηκε και το ***Hibernate ORM*** προκειμένου να απλοποιηθεί περαιτέρω η αλληλεπίδραση με τη βάση, μέσω της αντιστοίχισης αντικειμένων του προγράμματος σε πίνακες της βάσης και αντιστρόφως.

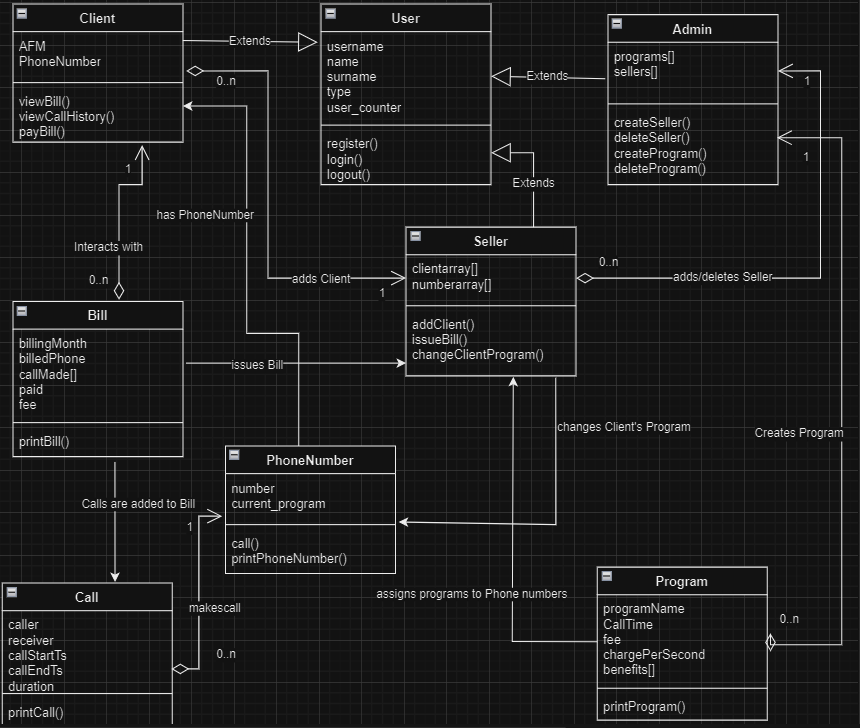
Η βάση δεδομένων που δημιουργείται από το πρόγραμμα αποθηκεύεται στον φάκελο “*bin*” του Application Server “*Apache Tomcat”*.

Το αρχείο “vietnam.db”, το οποίο έχει αποσταλεί μαζί με τον κώδικα του προγράμματος, περιέχει ενδεικτικά δεδομένα για την εκτέλεση και δοκιμή της εφαρμογής.

* 1. Maven

Το ***Maven*** επιλέχθηκε κατά την ανάπτυξη της διαδικτυακής εφαρμογής στο *VS Code,* για να γίνει εγκατάσταση και σωστή διαχείριση των εξαρτήσεων και να αυτοματοποιηθεί η διαδικασία κατασκευής και ανάπτυξης, διασφαλίζοντας μια συνεπή και οργανωμένη δομή έργου.

* 1. Διάγραμμα κλάσεων



* 1. Δομή του Project
* **src:** Περιλαμβάνει τον φάκελο Java, ο οποίος περιέχει το πακέτο (com.webProgramming) στο οποίο βρίσκεται ο κώδικας του project.
* **controllers**: Περιέχει τα αρχεία των *controllers* (*Servlets*) που διαχειρίζονται τις αιτήσεις των χρηστών, επεξεργάζονται τα δεδομένα και αποστέλλουν τις κατάλληλες απαντήσεις. (Θα τα δούμε αναλυτικότερα παρακάτω)
* **models:** Περιέχει τις κύριες κλάσεις του προγράμματος, οι οποίες χρησιμοποιούνται από το Hibernate ως μοντέλα για τη δημιουργία της βάσης δεδομένων. Οι κλάσεις αυτές αντιπροσωπεύουν τις οντότητες (entities) της εφαρμογής. Επιπλέον, έχουμε τις ακόλουθες κλάσεις:
  + Ο **AppContextListener** διασφαλίζει ότι η σύνδεση με τη βάση δεδομένων αρχικοποιείται όταν ξεκινάει η εφαρμογή και κλείνει όταν τερματίζει.
  + Η κλάση **Util** παρέχει μια μέθοδο για τη δημιουργία και την απόκτηση του SessionFactory, το οποίο είναι απαραίτητο για την αλληλεπίδραση με τη βάση δεδομένων μέσω του Hibernate.
  + Το **UserType** είναι ένας τύπος δεδομένων που ορίζει τους πιθανούς τύπους χρηστών στην εφαρμογή.
* **daos:** Το αρχείο κώδικα UserDao.java παρέχει λειτουργίες πρόσβασης σε δεδομένα για έλεγχο ταυτότητας χρήστη, ανάκτηση ανά αναγνωριστικό και αποθήκευση χρηστών σε βάση δεδομένων χρησιμοποιώντας Hibernate. Υποστηρίζει τους τύπους χρηστών Διαχειριστής, Πωλητής και Πελάτης. Η μέθοδος login επαληθεύει τα διαπιστευτικά χρήστη, η findById ανακτά έναν χρήστη με βάση το αναγνωριστικό του και η saveUser αποθηκεύει έναν νέο χρήστη στη βάση δεδομένων. Παρόμοια λειτουργικότητα παρέχεται από τα ProgramDao.java και PhoneNumberDao.java, τα οποία διαχειρίζονται την πρόσβαση σε δεδομένα για προγράμματα και αριθμούς τηλεφώνου, αντίστοιχα.
  + PhoneNumberDao
  + ProgramDao
  + UserDao
* **webapp:**
  + admin**,** client**,** seller**:** Περιέχουν τα αρχεία JSP που σχετίζονται με τη διεπαφή χρήστη (UI) για τους διαχειριστές, τους πελάτες και τους πωλητές αντίστοιχα.
  + WEB-INF**:** Περιέχει το αρχείο web.xml, το οποίο είναι το αρχείο ρύθμισης της εφαρμογής web.
  + common.jsp**,** error.jsp**,** index.html**, κ.λπ.:** Περιέχουν κοινά αρχεία JSP, HTML και CSS που χρησιμοποιούνται σε όλη την εφαρμογή.
* **assignments:** Περιέχει τα PDF των εργασιών και το αρχείο τεκμηρίωσης (documentation).
* **Επιπλέον:** Περιέχεται το διάγραμμα του προγράμματος (Diagram.drawio), το αρχείο README (README.md) και το αρχείο ρυθμίσεων του Maven (pom.xml).
* Σημείωση: Για τις ανάγκες της εφαρμογής χρειάστηκε να υλοποιήσουμε μέρος των λειτουργιών του Admin προκειμένου να δημιουργούμε sellers και programs και να δοκιμάζουμε τις λειτουργίες αυτών.
  1. Οδηγίες εκτέλεσης

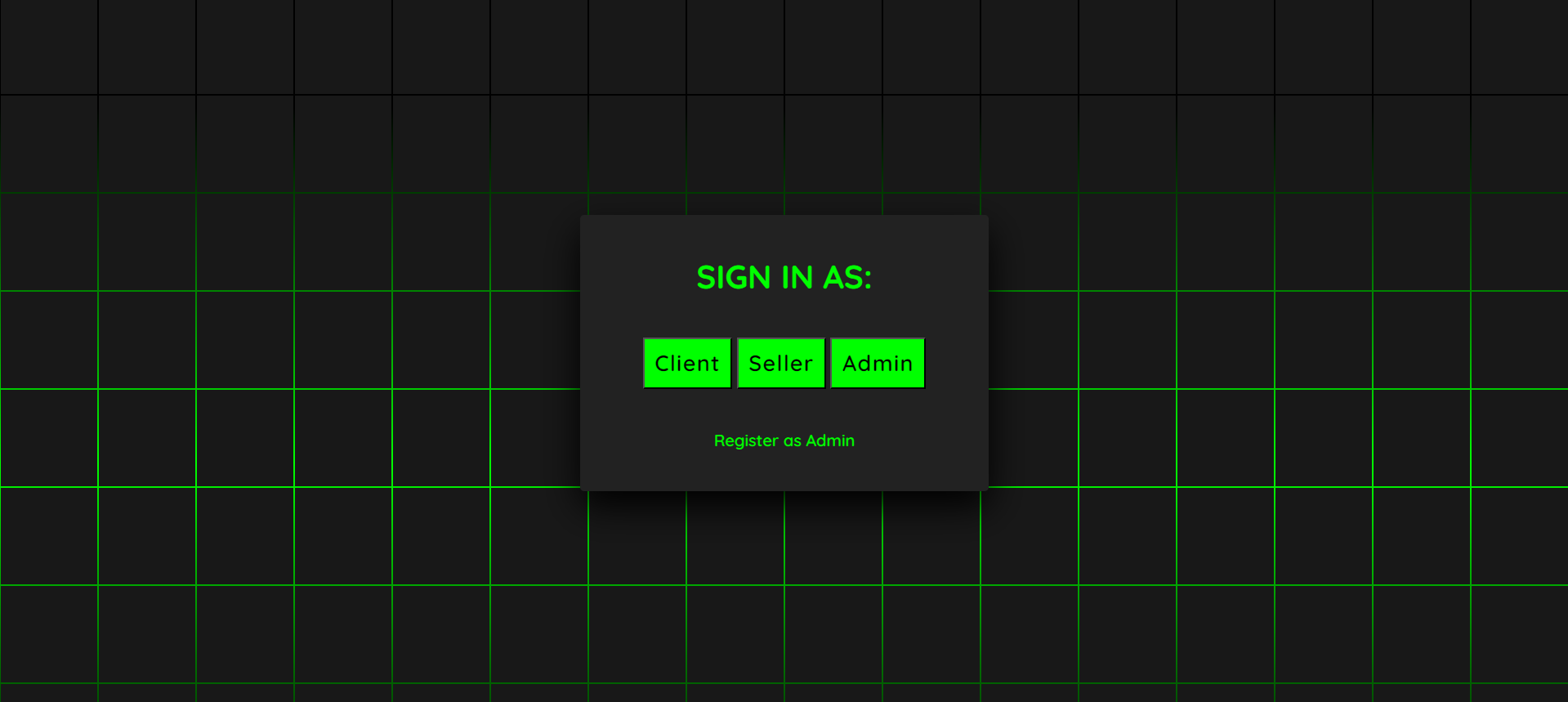
Για την πραγματοποίηση του project χρησιμοποιήθηκε *OpenJDK21*.

Μέσα στον παραδοτέο φάκελο περιλαμβάνεται το αρχείο “*vietnam.war*” καθώς και μία ενδεικτική βάση δεδομένων “*vietnam.db*”, με έτοιμα δεδομένα για τη δοκιμή της εφαρμογής.

Αναλυτικά βήματα εκτέλεσης:

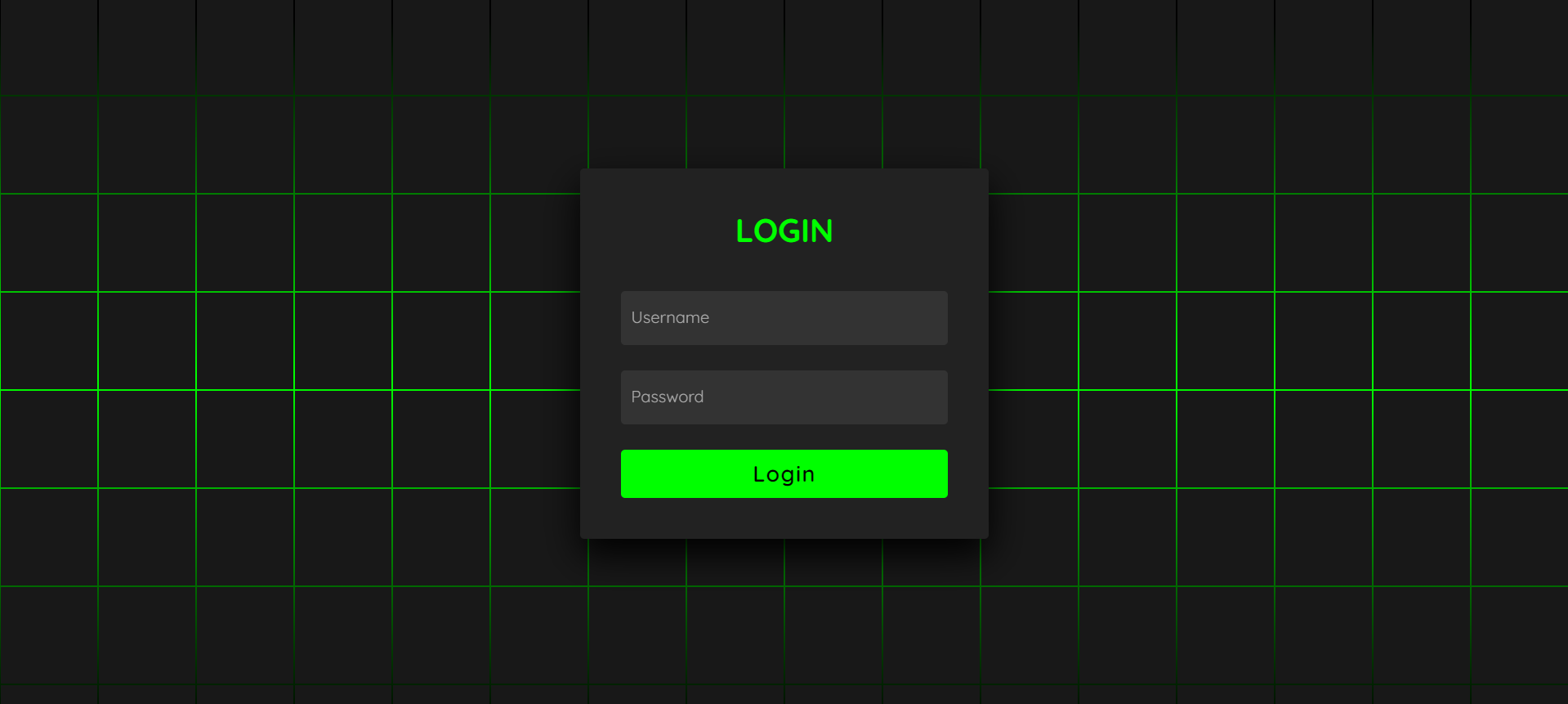
1. Κάνετε deploy το “*vietnam.war*” στο περιβάλλον του Apache Tomcat
2. Τοποθετείτε τη βάση “*vietnam.db*” στον φάκελο bin του Tomcat (εφόσον θέλετε να εκτελέσετε την εφαρμογή με έτοιμα δεδομένα)
3. Εκτελείτε το project
4. Διαδικτυακή διεπαφή (για τον Seller)
   1. Αρχική σελίδα (*index.html*)

Σε αυτή τη σελίδα έχει δημιουργηθεί ένα μενού επιλογών. Εκεί, ο χρήστης ανάλογα με την ιδιότητα του, επιλέγει το κατάλληλο κουμπί προκειμένου να τον κατευθύνουμε στο Log in page που έχουμε δημιουργήσει και να γίνει η αναζήτηση για τα credentials που θα μας δώσει, στην σωστή βάση δεδομένων.



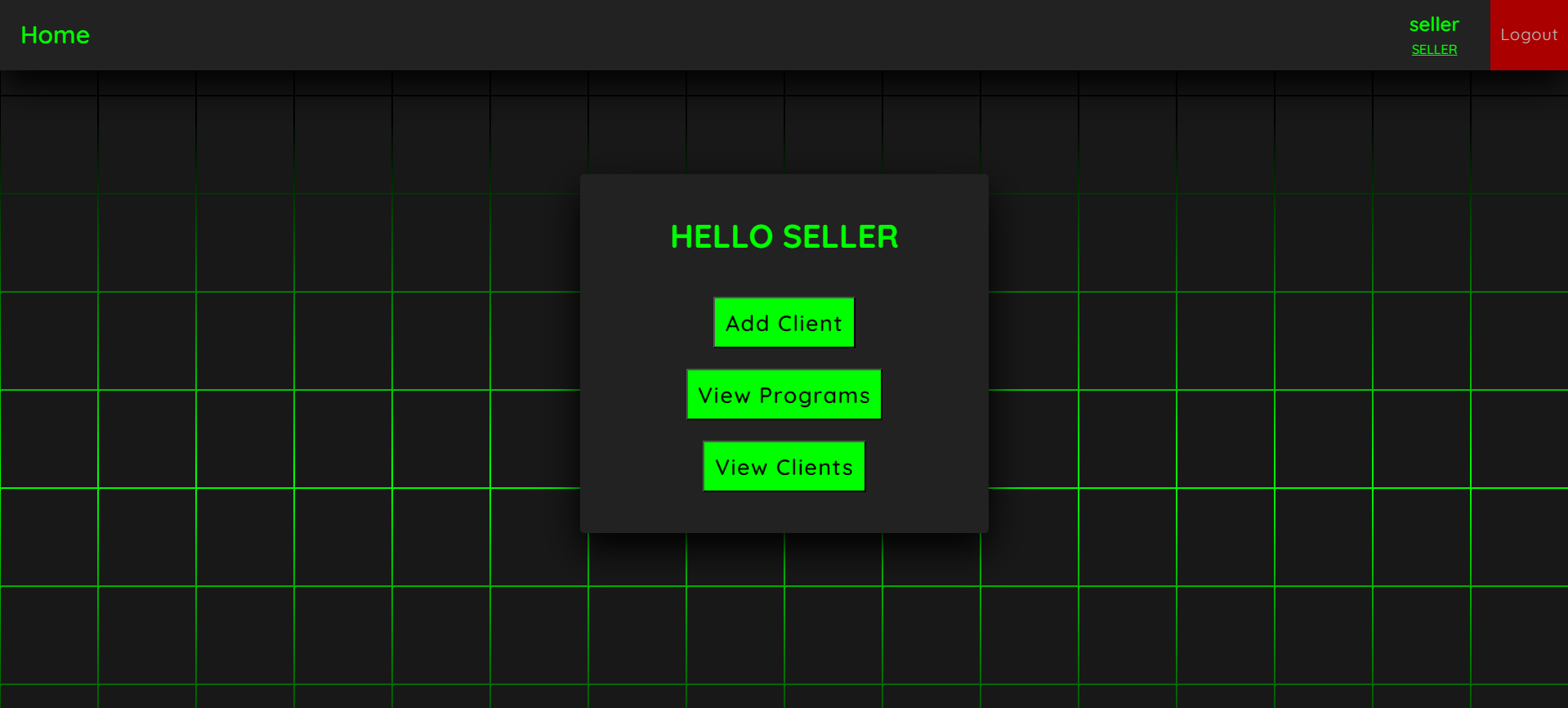
* 1. Log in Page (*loginPage.jsp*)

Έχει δημιουργηθεί Login page καθολικό για όλους τους τύπους χρηστών, το οποίο ανάλογα με την παράμετρο που θα δεχθεί από το κεντρικό μενού, όταν ο χρηστής πατήσει να συνδεθεί θα γίνει έλεγχος για τα credentials του στον αντίστοιχο πίνακα (clients, sellers, admins).



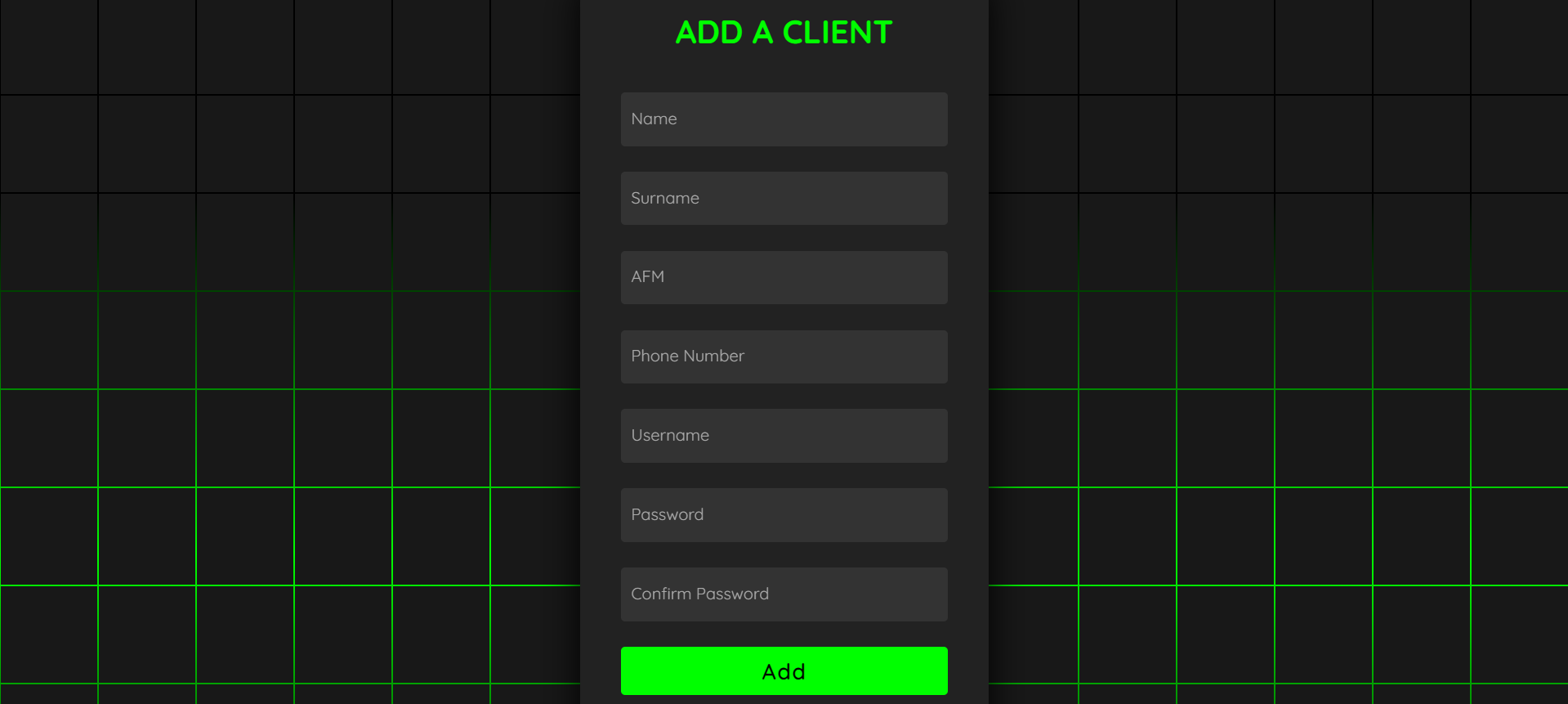
* 1. Μενού του Seller (*menu.jsp*)

Μέσω ενός φιλικού προς το χρήστη μενού, ο seller μπορεί να επιλέξει ποια λειτουργικότητα θα πραγματοποιήσει, πατώντας το αντίστοιχο κουμπί.



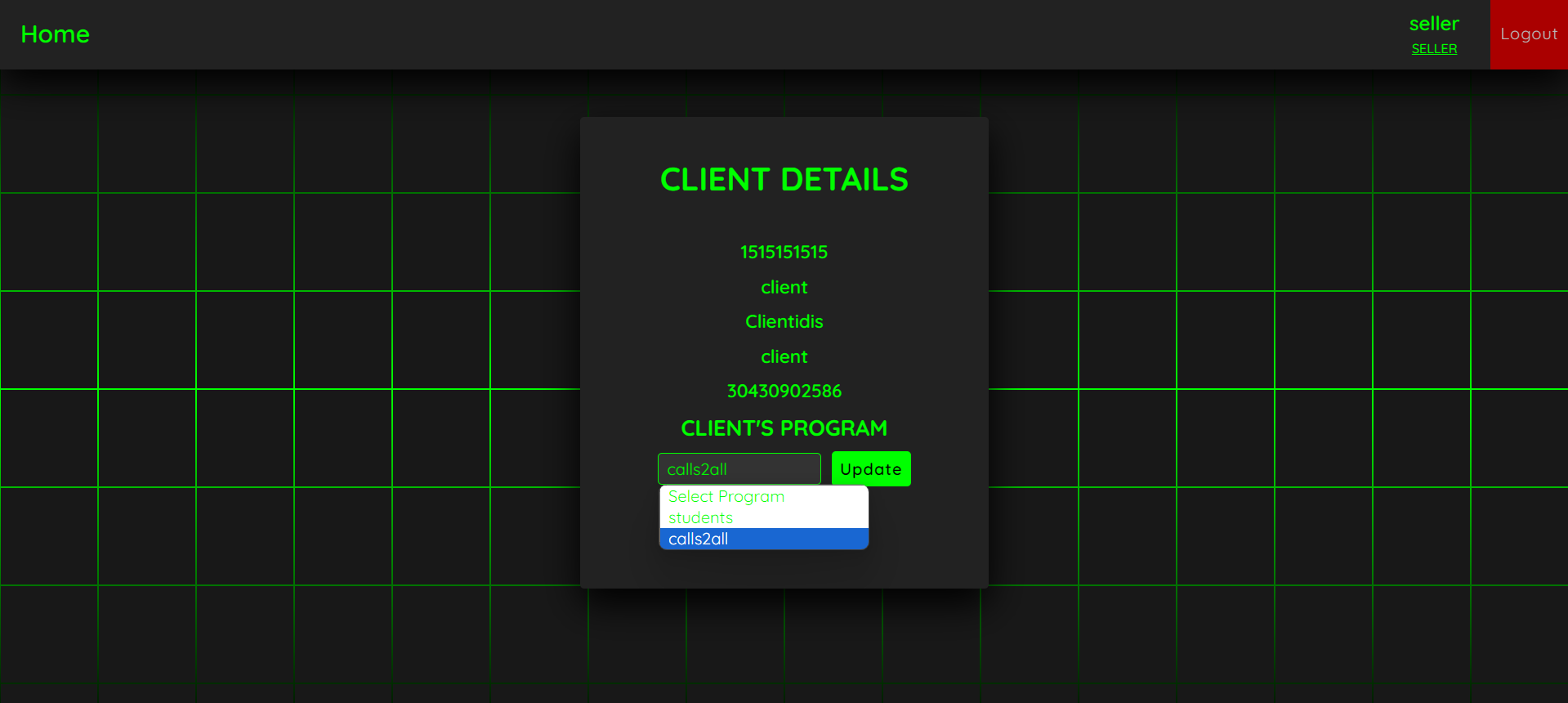
* 1. Προσθήκη Client (*AddClient.jsp*)

Ο seller συμπληρώνει τα στοιχεία που ζητούνται από την φόρμα, επιλέγει προσθήκη, γίνονται οι απαραίτητοι έλεγχοι και έπειτα μέσω του *ORM* δημιουργείται νέο αντικείμενο χρήστη “client”. To αντικείμενο που δημιουργήθηκε προστίθεται αυτομάτως στον κατάλληλο πίνακα από το *Hibernate.*



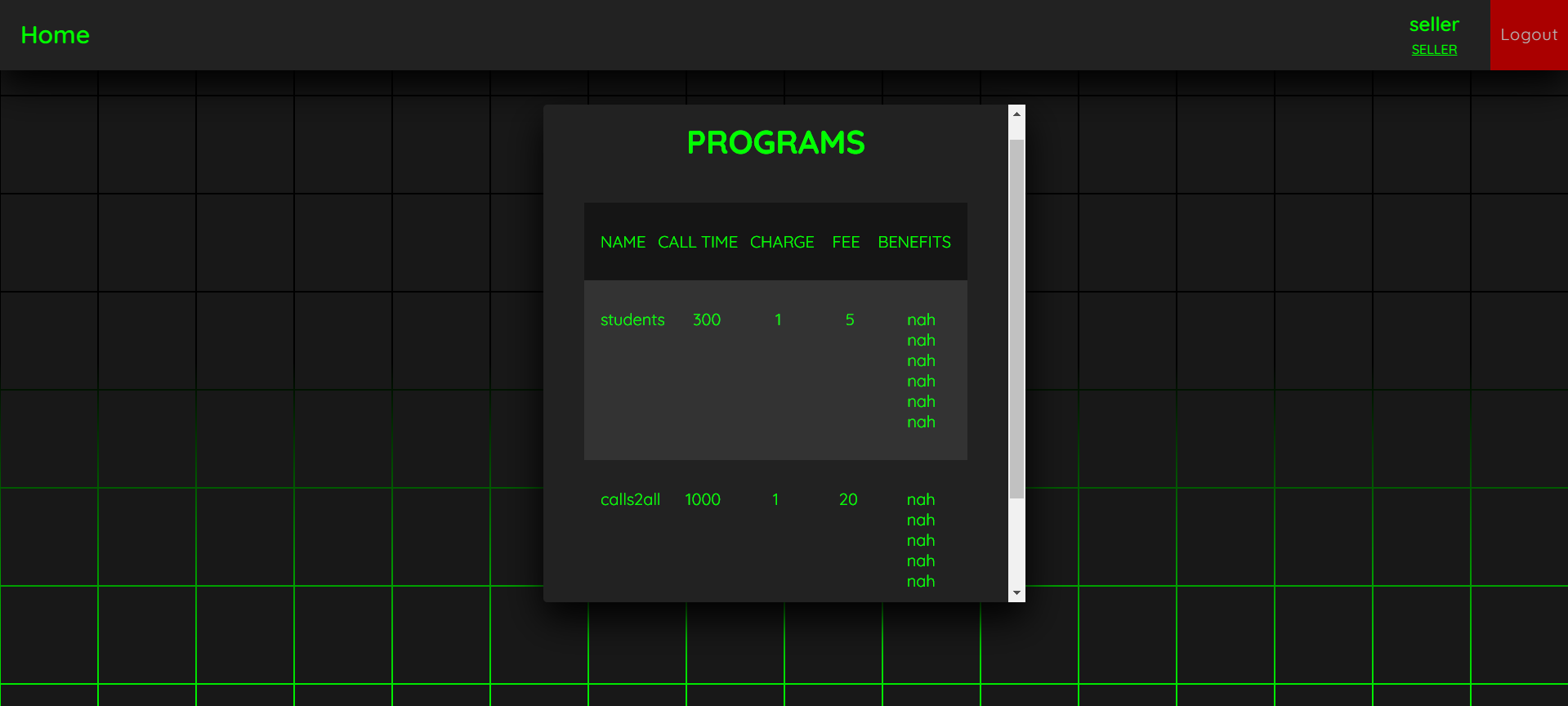
* 1. Ανάθεση προγράμματος τηλεφωνίας σε Client (*ClientDetails.jsp*)

Μέσω αυτής της διεπαφής, ο seller έχει τη δυνατότητα αφού επιλέξει από την ειδικά διαμορφωμένη λίστα έναν client, να του αναθέσει ένα νέο πρόγραμμα κινητής τηλεφωνίας, σύμφωνα με το οποίο θα γίνεται η χρέωση του.



* 1. Προβολή διαθέσιμων προγραμμάτων κινητής (*ProgramsList.jsp*)

Μέσω αυτής της διεπαφής, ο seller έχει τη δυνατότητα να δει και να ενημερωθεί για όλα τα πακέτα τηλεφωνίας της εταιρείας, όπως αυτά «αλιεύονται» από την βάση δεδομένων και απεικονίζονται σε μορφή πίνακα στην εφαρμογή.



1. Κώδικας - Λειτουργικότητες (Servlets)
   1. Log in (LoginController.java)

Η κλάση LoginController επεκτείνει την HttpServlet και διαχειρίζεται τη λειτουργία σύνδεσης της εφαρμογής. Επεξεργάζεται αιτήματα σύνδεσης, επικυρώνει τα στοιχεία των χρηστών και τους ανακατευθύνει στις κατάλληλες σελίδες βάσει του τύπου χρήστη.

**Μέθοδοι**

***doPost(HttpServletRequest request, HttpServletResponse response)***

Υπερκαλύπτει τη μέθοδο doPost για την επεξεργασία αιτημάτων POST για σύνδεση χρήστη.

* **Ρυθμίσεις κωδικοποίησης**: Ρυθμίζει την κωδικοποίηση αιτήματος και απόκρισης σε UTF-8.
* **Δημιουργία Αντικειμένου Login**: Δημιουργεί ένα νέο αντικείμενο Login και θέτει το όνομα χρήστη και τον κωδικό πρόσβασης από τις παραμέτρους του αιτήματος.
* **Έλεγχος Τύπου Χρήστη**: Επικυρώνει και ρυθμίζει τον τύπο χρήστη (ADMIN, SELLER, CLIENT).
* **Επικύρωση Στοιχείων Σύνδεσης**: Χρησιμοποιεί το UserDao για την επικύρωση των στοιχείων σύνδεσης.

**Έλεγχος Χρήστη και Ανακατεύθυνση:**

 **Διαχείριση Συνεδρίας**: Δημιουργεί ή ανακτά τη συνεδρία και αποθηκεύει το όνομα χρήστη και τον τύπο χρήστη.

 **Ανακατεύθυνση**: Ανακατευθύνει τον χρήστη στις κατάλληλες σελίδες βάσει του τύπου χρήστη:

* **ADMIN**: /admin/menu.jsp
* **SELLER**: /seller/menu.jsp
* **CLIENT**: /client/menu.jsp

**Διαχείριση σφαλμάτων:**

* **Πρόσθετα Αιτήματος και Προώθηση**: Θέτει το μήνυμα σφάλματος στο αίτημα και προωθεί στο /error.jsp σε περίπτωση σφάλματος.
  1. Log out (LogoutController.java)

Ο κώδικας αυτός υλοποιεί την λειτουργικότητα της αποσύνδεσης χρήστη από τη web εφαρμογή. Λειτουργεί ως εξής:

**Αντιστοίχιση URL (Mapping):**

* + Η annotation @WebServlet("/logout") καθορίζει ότι αυτό το servlet θα ανταποκρίνεται σε HTTP αιτήματα GET που απευθύνονται στο URL "/logout".

**Μέθοδος doGet() (HTTP GET request handling):**

* + Αυτή είναι η κύρια μέθοδος που καλείται όταν ο χρήστης κάνει κλικ σε έναν σύνδεσμο αποσύνδεσης ή αποστέλλει ένα αίτημα GET στο "/logout".

**Διαχείριση Συνεδρίας (Session Management):**

* + HttpSession session = request.getSession();: Αποκτά την τρέχουσα συνεδρία του χρήστη από το αίτημα. Η συνεδρία είναι ένας μηχανισμός που επιτρέπει την παρακολούθηση της κατάστασης του χρήστη μεταξύ πολλαπλών αιτημάτων.
  + if (session != null): Ελέγχει αν υπάρχει ενεργή συνεδρία (δηλαδή ο χρήστης είναι συνδεδεμένος).

**Διαγραφή Δεδομένων Χρήστη:**

* + session.removeAttribute("user");: Αν υπάρχει συνεδρία, αφαιρεί το αντικείμενο "user" που (πιθανότατα) αποθηκεύει πληροφορίες του συνδεδεμένου χρήστη.
  + session.invalidate();: Ακυρώνει εντελώς τη συνεδρία, τερματίζοντας τη σύνδεση.

**Ανακατεύθυνση (Redirection):**

* + response.sendRedirect("index.html");: Ανακατευθύνει τον χρήστη στη σελίδα "index.html", η οποία λογικά είναι η αρχική σελίδα ή η σελίδα σύνδεσης της εφαρμογής σας.
  1. Register (RegisterController.java)

**Επισκόπηση κλάσης:**

Η κλάση RegisterControler είναι ένα HttpServlet που διαχειρίζεται τη δημιουργία νέων χρηστών στην εφαρμογή. Ανάλογα με τον τύπο χρήστη που επιλέγεται, δημιουργεί έναν νέο πωλητή ή πελάτη και αποθηκεύει τα στοιχεία του στη βάση δεδομένων.

**Μέθοδος:**

***doPost(HttpServletRequest request, HttpServletResponse response)***

Υπερκαλύπτει τη μέθοδο doPost για την επεξεργασία αιτημάτων POST για δημιουργία νέων χρηστών.

* **Ρυθμίσεις κωδικοποίησης**: Ρυθμίζει την κωδικοποίηση αιτήματος και απόκρισης σε UTF-8.
* **Λήψη Παραμέτρων**: Λαμβάνει τις απαραίτητες παραμέτρους για τη δημιουργία νέου χρήστη.
* **Έλεγχος Εγκυρότητας**: Ελέγχει την ορθότητα των παραμέτρων όπως ο κωδικός πρόσβασης και η επιβεβαίωσή τους.
* **Δημιουργία Χρήστη**: Δημιουργεί νέο πωλητή ή πελάτη ανάλογα με τον επιλεγμένο τύπο χρήστη.
* **Ανακατεύθυνση**: Αν η δημιουργία είναι επιτυχής, ανακατευθύνει τον χρήστη σε επιτυχία δημιουργίας χρήστη.
* **Διαχείριση Σφαλμάτων**: Αν δημιουργηθεί σφάλμα κατά τη διαδικασία δημιουργίας, εμφανίζει μήνυμα σφάλματος.
  1. Seller Servlet (SellerController.java)

Ο κώδικας του Seller Servlet διαχειρίζεται τα αιτήματα δημιουργίας νέων πωλητών (sellers) στη web εφαρμογή. Ο σκοπός του servlet είναι να επαληθεύει τα δικαιώματα του συνδεδεμένου χρήστη, να επαληθεύει τις παραμέτρους που υποβάλλονται μέσω της φόρμας και να δημιουργεί έναν νέο πωλητή μέσω του συστήματος.

**Έλεγχος Συνδεδεμένου Χρήστη**:

Το servlet αρχικά ελέγχει αν υπάρχει συνδεδεμένος χρήστης ανακτώντας το αντικείμενο Login από το session.

Αν δεν υπάρχει συνδεδεμένος χρήστης, ο χρήστης ανακατευθύνεται στη σελίδα σύνδεσης (login.jsp).

**Έλεγχος Δικαιωμάτων Χρήστη**:

Μόνο οι χρήστες με τύπο ADMIN επιτρέπεται να δημιουργούν νέους πωλητές.

Αν ο συνδεδεμένος χρήστης δεν είναι ADMIN, πετιέται εξαίρεση IllegalArgumentException.

**Ανάκτηση και Επαλήθευση Παραμέτρων**:

Οι παράμετροι name, surname, username, password, και confirmPassword ανακτώνται από το αίτημα.

Ελέγχεται αν κάποια από τις παραμέτρους είναι null.

Ελέγχεται αν τα δύο password ταιριάζουν.

**Δημιουργία Πωλητή**:

Ο πωλητής δημιουργείται μέσω της μεθόδου createSeller της κλάσης Admin.

Ο νέος πωλητής αποθηκεύεται στη βάση δεδομένων μέσω του UserDao.

**Αντιμετώπιση Επιτυχίας και Σφαλμάτων**:

Αν ο πωλητής δημιουργηθεί επιτυχώς, το αίτημα προωθείται στη σελίδα success.jsp με κατάλληλα μηνύματα επιτυχίας.

Σε περίπτωση αποτυχίας ή εξαίρεσης, το αίτημα προωθείται στη σελίδα error.jsp με το αντίστοιχο μήνυμα σφάλματος.

**Σχέση με άλλες Κλάσεις:**

* **UserDao**: Παρέχει μεθόδους για την αποθήκευση χρηστών στη βάση δεδομένων.
* **Login**: Αντιπροσωπεύει τον συνδεδεμένο χρήστη και τις πληροφορίες του.
* **Admin**: Υποκλάση του χρήστη που μπορεί να δημιουργεί νέους πωλητές.
* **Seller**: Οντότητα πωλητή που δημιουργείται από έναν admin.
  1. Client Servlet (ClientController.java)

Αυτό το servlet διαχειρίζεται τις αλληλεπιδράσεις που σχετίζονται με τους χρήστες τύπου “client” της εφαρμογής. Χειρίζεται τόσο την προβολή των πληροφοριών των πελατών όσο και τη δημιουργία νέων πελατών από εξουσιοδοτημένους πωλητές.

**doGet (Προβολή Πελατών):**

* + **Έλεγχος Ταυτότητας:** Ελέγχει αν ένας πωλητής είναι συνδεδεμένος. Αν όχι, ανακατευθύνει στη σελίδα σύνδεσης.
  + **Διαχείριση Παραμέτρων:**
    - Αν υπάρχει η παράμετρος id στο αίτημα, ανακτά τα στοιχεία του συγκεκριμένου πελάτη μαζί με τα διαθέσιμα προγράμματα και τα εμφανίζει στη σελίδα "seller/ClientDetails.jsp".
    - Αν δεν δίνεται id, ανακτά μια λίστα όλων των πελατών του πωλητή και τους εμφανίζει στη σελίδα "seller/ClientsList.jsp".

**doPost (Δημιουργία Πελάτη):**

* + **Έλεγχος Ταυτότητας:** Όμοια με το doGet, εξασφαλίζει ότι ένας πωλητής είναι συνδεδεμένος.
  + **Συλλογή Δεδομένων:** Συλλέγει τα δεδομένα του πελάτη (όνομα, επώνυμο, όνομα χρήστη, κωδικό πρόσβασης, επιβεβαίωση κωδικού πρόσβασης και ΑΦΜ) από τη φόρμα του αιτήματος.
  + **Επικύρωση:**
    - Ελέγχει αν όλα τα απαιτούμενα πεδία είναι συμπληρωμένα.
    - Επαληθεύει ότι οι κωδικοί πρόσβασης ταιριάζουν.
  + **Δημιουργία Πελάτη:** Δημιουργεί ένα νέο αντικείμενο Client χρησιμοποιώντας τη μέθοδο seller.createClient.
  + **Persist (μόνιμη αποθήκευση):** Αποθηκεύει τον νέο πελάτη στη βάση δεδομένων χρησιμοποιώντας το UserDao.
  + **Απάντηση:**
    - Σε περίπτωση επιτυχίας, προωθεί το αίτημα στο "success.jsp" με ένα μήνυμα επιτυχίας.
    - Εάν παρουσιαστεί σφάλμα (π.χ., αποτυχία επικύρωσης, πρόβλημα βάσης δεδομένων), προωθεί στο "/error.jsp" με τις λεπτομέρειες του σφάλματος.

**Χειρισμός Σφαλμάτων:**

* Χρησιμοποιεί ένα μπλοκ try...catch για να συλλάβει εξαιρέσεις.
* Σε περίπτωση σφαλμάτων, στέλνει το αίτημα στη σελίδα "/error.jsp", εμφανίζοντας το μήνυμα σφάλματος για εντοπισμό σφαλμάτων.
  1. Phone Servlet (PhoneControler.java)

Αυτό το servlet χειρίζεται την ενημέρωση των τηλεφωνικών αριθμών (phone numbers) των χρηστών “client” της web εφαρμογής. Συγκεκριμένα, επιτρέπει στους πωλητές (sellers) να αντιστοιχίζουν τηλεφωνικούς αριθμούς των πελατών σε συγκεκριμένα προγράμματα κινητής τηλεφωνίας (programs).

**Λειτουργία (Μέθοδος PUT):**

**Έλεγχος Εξουσιοδότησης:**

* + Ελέγχει αν ο χρήστης είναι συνδεδεμένος και αν είναι πωλητής. Μόνο οι πωλητές επιτρέπεται να ενημερώνουν τηλεφωνικούς αριθμούς.

**Λήψη Δεδομένων:**

* + Διαβάζει τα δεδομένα από το σώμα (body) του HTTP αιτήματος PUT, το οποίο αναμένεται να είναι σε μορφή JSON.
  + Εξάγει τον αριθμό τηλεφώνου (number) και το αναγνωριστικό προγράμματος (program\_id).

**Επικύρωση:**

* + Ελέγχει αν ο τηλεφωνικός αριθμός και το πρόγραμμα υπάρχουν στη βάση δεδομένων.
  + Επαληθεύει ότι ο τηλεφωνικός αριθμός ανήκει σε έναν από τους πελάτες του πωλητή.
  + Επιβεβαιώνει ότι το πρόγραμμα ανήκει στον διαχειριστή (admin) του πωλητή.

**Ενημέρωση:**

* + Αντιστοιχίζει τον τηλεφωνικό αριθμό με το επιλεγμένο πρόγραμμα.
  + Ενημερώνει τη βάση δεδομένων.

**Απάντηση:**

* + Αν η ενημέρωση είναι επιτυχής, στέλνει μια απάντηση HTTP με κωδικό 200 (OK) και το μήνυμα "Updated phone number successfully".
  + Αν παρουσιαστεί κάποιο σφάλμα, στέλνει μια απάντηση HTTP με κωδικό 500 (Internal Server Error) και το μήνυμα του σφάλματος.
  1. Program Servlet (ProgramController.java)

Αυτό το servlet διαχειρίζεται τη λειτουργικότητα των προγραμμάτων (programs) της web εφαρμογής. Παρέχει δύο κύριες λειτουργίες:

* + **doGet (Λίστα Προγραμμάτων):** Εμφανίζει μια λίστα με όλα τα διαθέσιμα προγράμματα στον συνδεδεμένο πωλητή (seller).
  + **doPost (Δημιουργία Προγράμματος):** Επιτρέπει στον διαχειριστή (admin) να δημιουργήσει ένα νέο πρόγραμμα.

**doGet:**

**Έλεγχος Εξουσιοδότησης:** Επιβεβαιώνει ότι ένας πωλητής είναι συνδεδεμένος.

**Ανάκτηση Προγραμμάτων:**

* + Χρησιμοποιεί μια Hibernate query για να ανακτήσει όλα τα προγράμματα που ανήκουν στον διαχειριστή (admin) του πωλητή.

**Προβολή:**

* + Αποθηκεύει τα προγράμματα ως attribute στο request.
  + Προωθεί το αίτημα στη σελίδα JSP "seller/ProgramsList.jsp" για την εμφάνιση των προγραμμάτων.

**doPost:**

**Έλεγχος Εξουσιοδότησης:** Επιβεβαιώνει ότι ένας διαχειριστής είναι συνδεδεμένος.

**Συλλογή Δεδομένων:**

* + Λαμβάνει τα δεδομένα του προγράμματος (όνομα, χρόνος κλήσης, πάγιο τέλος, χρέωση ανά δευτερόλεπτο, πλεονεκτήματα) από το αίτημα HTTP.

**Δημιουργία Προγράμματος:**

* + Δημιουργεί ένα νέο αντικείμενο Program.

**Αποθήκευση:**

* + Αποθηκεύει το νέο πρόγραμμα στη βάση δεδομένων χρησιμοποιώντας το ProgramDao.

**Απάντηση:**

* + Εάν η δημιουργία είναι επιτυχής:
    - Ανακατευθύνει τον χρήστη στη σελίδα "success.jsp" με ένα μήνυμα επιτυχίας.
  + Σε περίπτωση σφάλματος:
    - Ανακατευθύνει τον χρήστη στη σελίδα "/error.jsp" με ένα μήνυμα σφάλματος.

1. Βιβλιογραφικές Πηγές

* **BROWN LARRY**, *SERVLETS ΚΑΙ ΣΕΛΙΔΕΣ ΔΙΑΚΟΜΙΣΤΗ JAVA*, ΤΕΧΝΟΛΟΓΙΕΣ ΠΥΡΗΝΑ
* **Τάνια Κερκίρη**, *Java προχωρημένες τεχνικές*, ΚΛΕΙΔΑΡΙΘΜΟΣ
* **Ε. Αλέπης - Ι.Χ. Παναγιωτόπουλος,** *Αντικειμενοστραφείς Γλώσσες Προγραμματισμού:* JAVA, εκδ. ΒΑΡΒΑΡΙΓΟΥ
* **Draw.io** (εργαλείο για τη δημιουργία διαγράμματος κλάσεων)