



**Πανεπιστήμιο Πειραιώς
Τμήμα Ψηφιακών
Συστημάτων**

**ΑΠΑΛΛΑΚΤΙΚΗ
ΕΡΓΑΣΙΑ
ΔΙΚΤΥΟΚΕΝΤΡΙΚΑ
ΠΛΗΡΟΦΟΡΙΑΚΑ
ΣΥΣΤΗΜΑΤΑ
ΙΑΣΩΝ ΤΖΩΡΤΖΗΣ**

Project Name : BarberShop

ΕΝΟΤΗΤΑ 1: ΘΕΜΑΤΟΛΟΓΙΑ ΣΥΣΤΗΜΑΤΟΣ

Η διαδικασία που επιλέχθηκε να υλοποιηθεί είναι η διαδικασία της online κράτησης του κομμωτηρίου Groovin Barber.

Πιο συγκεκριμένα με το σύστημα αυτό δίνεται αρχικά η δυνατότητα στον χρήστη να δει περισσότερες πληροφορίες για το κομμωτήριο, επίσης του δίνεται η δυνατότητα να πραγματοποιεί μια ή περισσότερες κρατήσεις για τον εαυτό του ή για έναν συνάνθρωπο του μέσα από το σύστημα και ακόμα η δυνατότητα ενημέρωσης μιας υπάρχουσας κράτησης και διαγράψης. Τέλος του δίνεται η δυνατότητα να κάνει subscribe στο weekly newsletter του κομμωτηρίου .

Όλα τα παραπάνω είναι δυνατά μετρά από εγγραφή του χρήστη στο σύστημα και μετρά από login με τα credentials που πραγματοποίησε την εγγραφή.

Από την άλλη πλευρά στον διαχειριστή παρέχεται η δυνατότητα να δει όλες τις κρατήσεις που έχουν πραγματοποιηθεί μέσα από το σύστημα με την δυνατότητα διαγράψης οποιαδήποτε κράτησης επιθυμεί ο διαχειριστής. Επίσης παρέχεται η δυνατότητα ενημέρωσης υπάρχουσας κράτησης και προσθήκη νέας κράτησης. Ακόμα παρέχεται στον διαχειριστή η δυνατότητα να δει όλα τα usernames των χρηστών που είναι εγγεγραμμένοι και τους ρόλους τους όπως επίσης και τα emails που είναι εγγεγραμμένα στο newsletter. Τέλος δίνεται η δυνατότητα στον διαχειριστή να δει τα projected earnings του καταστήματος από τις κρατήσεις που έχουν πραγματοποιηθεί μέσω του συστήματος.

Όπως για τον χρήστη έτσι και για τον διαχειριστή όλα τα παραπάνω είναι διαθέσιμα έπειτα από την εγγραφή του διαχειριστή στο σύστημα και έπειτα από login με τα credentials που πραγματοποίησε την εγγραφή του.

Το πρόβλημα που επιλύεται με το σύστημα αυτό είναι η ελάττωση τηλεφωνημάτων από πελάτες που θέλουν να πραγματοποιήσουν κράτηση στο κομμωτήριο και επίσης επιτυγχάνεται η καλύτερη οργάνωση του συστήματος κρατήσεων, διότι πριν πραγματοποιούνταν η οργάνωση αυτή πάνω σε ένα ημερολόγιο που δεν είναι και το πιο αποτελεσματικό όταν οι κρατήσεις φτάνουν πάνω από ένα βαθμό οπού δεν χωράνε πια στις γραμμές του ημερολογίου. Τέλος δίνεται η δυνατότητα μέσα από το σύστημα για καλύτερο υπολογισμό των κερδών της επιχείρησης.

ΕΝΟΤΗΤΑ 2: ΥΛΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ

- 1) Η συνολική κατασκευή του συστήματος βασίστηκε στην αρχιτεκτονική μικρό υπηρεσιών. Με αποτέλεσμα το σύστημα να έχει χωριστεί σε μια σειρά αυτονόμων στοιχείων που συνθέτουν στο σύνολο τους το σύστημα αυτό. Πιο συγκεκριμένα το σύστημα έχει διαχωριστεί σε 21 διαφορετικά RESTful Web

services που το καθένα από αυτά έχει το δικό του σκοπό και την δικιά του λειτουργία που προσφέρει στο σύστημα.

2)

WEB SERVICES:

1)

Create_BarberShop

Description	Create BarberShop database
Endpoint URL	http://localhost:8080/BarberShop/rest/Create/barbershop
HTTP Method	GET
Parameters	N/A
Returns	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται όταν χρησιμοποιείται το σύστημα για πρώτη φορά και δημιουργεί την βάση δεδομένων BarberShop που περιέχει όλες τις πληροφορίες που έχουν να κάνουν με τις κρατήσεις και τα projected earnings. Καλείται πάντα στο αρχείο home.jsp και στο adminHome.jsp και έχει ως σκοπό όταν εισέρθει στο σύστημα για πρώτη φορά ένας χρήστης ή διαχειριστής να δημιουργηθεί η βάση δεδομένων πριν εισαχθούν στοιχεία έτσι ώστε να μην υπάρχουν unexpected errors.

2)

Create_userDatabase

Description	Create usersdatabase database
Endpoint URL	http://localhost:8080/BarberShop/rest/Creating/usersDatabase
HTTP Method	GET
Parameters	N/A
Returns	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται όταν χρησιμοποιούνται το Login και το Register του συστήματος για πρώτη φορά και δημιουργεί την βάση δεδομένων usersDatabase που περιέχει όλες τις πληροφορίες που έχουν να κάνουν με τους χρήστες όπως για παράδειγμα ποιοι είναι loggedIn μέσα στο σύστημα ,το username και το password τους το UserID , το uuid κάθε χρήστη όπως και τον ρολό που έχουν.

Καλείται πάντα στο αρχείο Login.jsp και στο Register.jsp και έχει ως σκοπό όταν προσπαθεί κάποιος να εγγραφεί στο σύστημα ή να εισέλθει για πρώτη φορά ένας χρήστης ή διαχειριστής να δημιουργηθεί η βάση δεδομένων πριν εισαχθούν στοιχεία έτσι ώστε να μην υπάρχουν unexpected errors.

3)

CheckIfLoggedIn

Description	Checking if user is logged in
Endpoint URL	http://localhost:8080/BarberShop/rest/CheckIfLoggedIn/uuid_check/{user_uuid_to_check}
HTTP Method	GET
Parameters	Users uuid
Returns	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται σε κάθε available route χρήστη και διαχειριστή εκτός από το Login.jsp και το Register.jsp και κάνει check με βάση το uuid του κάθε χρήστη ή διαχειριστή αν είναι logged in μέσα στο σύστημα. Αν είναι logged in τότε επιστρέφεται μήνυμα επιτυχίας, αν δεν είναι τότε επιστρέφεται μήνυμα αποτυχίας και ο χρήστης ή διαχειριστής γίνεται redirected στο Login.jsp που είναι το Login Page.

4)

CheckRole

Description	Checking if the role of the user if he is permitted to go to that route
Endpoint URL	http://localhost:8080/BarberShop/rest/CheckRole/checking_role/{userID}
HTTP Method	GET
Parameters	Users userID
Returns	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται σε κάθε available route χρήστη και διαχειριστή εκτός από το Login.jsp και το Register.jsp και κάνει check με βάση το userID του κάθε χρήστη η διαχειριστή αν με βάση το role του είναι permitted να κάνει access ένα συγκεκριμένο route. Αν είναι permitted επιστρέφεται μήνυμα επιτυχίας. Αν δεν είναι επιστρέφεται μήνυμα αποτυχίας και ο χρήστης η διαχειριστής γίνεται redirected στο Login.jsp που είναι το Login Page.

5)

createAppointment

Descripti on	Takes users or administrators input and inserts the data into the barbershop database creating in that way a booking
Endpoi nt URL	http://localhost:8080/BarberShop/rest/CreateAppointment/new_appointment/{date}/{haircut}/{time}/{userID}/{price}/{Firstname}
HTTP Metho d	GET
Param eters	Users input through a form that includes:date,haircut type,time,the userID, the price is calculated based on haircut type, Firstname
Return s	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα routes appointment.jsp που είναι το route με το οποίο ο χρήστης δημιουργεί μια κράτηση και στο route addAppointment_admin.jsp που είναι το route με το οποίο ο διαχειριστής προσθέτει μια κράτηση και παίρνει το input μέσα από μια φόρμα και τα βάζει μέσα στην βάση δεδομένων. Πιο συγκεκριμένα αρχικά προσπαθεί να δημιουργήσει το appointment table στην αρχή αν δεν υπάρχει, έπειτα παίρνει το maxappointmentID από τα υπάρχοντα appointment και έπειτα κάνει assign ένα appointmentID με βάση αυτό και προσθέτει τα δεδομένα στην βάση δεδομένων.

6)

deleteAppointment

Descripti on	Takes an appointmentID and deletes the appointment that was selected by the user
Endpoint URL	http://localhost:8080/BarberShop/rest/Deleteappointment/delete_appointm ent/{appointmentID}
HTTP Method	GET

Parameters	appointmentID
Returns	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα routes delete_appointment.jsp που είναι το route με το οποίο ο χρήστης διαγράφει μια κράτηση που είχε κάνει . Πιο συγκεκριμένα αρχικά προσπαθεί να δημιουργήσει το table appointments αν δεν υπάρχει ήδη και έπειτα κάνει check με βάση το appointmentId αν υπάρχει αυτή η κράτηση και αν υπάρχει διαγράφει την κράτηση αυτή και στέλνει μήνυμα επιτυχίας. Αν δεν υπάρχει επιστρέφεται μήνυμα αποτυχίας.

7)

deleteAppointments_Admin

Description	Takes an appointmentID and deletes the appointment that was selected by the admin
Endpoint URL	http://localhost:8080/BarberShop/rest/Deleteappointment_Admin/delete_Appointment_Admin/{appointmentID}
HTTP Method	GET
Parameters	appointmentID
Returns	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα routes admin_appointment.jsp που είναι το route με το οποίο ο admin διαχειρίζεται όλες τις κρατήσεις του. Πιο συγκεκριμένα αρχικά προσπαθεί να δημιουργήσει το table appointments αν δεν υπάρχει ήδη και έπειτα κάνει check με βάση το appointmentId αν υπάρχει αυτή η κράτηση και αν υπάρχει διαγράφει την κράτηση αυτή και στέλνει μήνυμα επιτυχίας. Αν δεν υπάρχει επιστρέφεται μήνυμα αποτυχίας.

8)

expectedEarnings

Description	Takes all the prices from the appointments and sums them
Endpoint URL	http://localhost:8080/BarberShop/rest/GetexpectedEarnings/earnings

HTTP Method	GET
Parameters	N/A
Returns	Total projected earnings based on appointments

Description:

Το παραπάνω Web Service χρησιμοποιείται στο route earnings.jsp που είναι το route με το οποίο ο admin μπορεί να δει τα projected earnings . Πιο συγκεκριμένα παίρνει όλες τις τιμές κάθε κράτησης και τις προσθέτει και γυρνάει το αποτέλεσμα.

9)

getAdminappointments

Description	Gets all the appointments and returns them as a json Array
Endpoint URL	http://localhost:8080/BarberShop/rest/GetAdminappointments/get_appointments
HTTP Method	GET
Parameters	N/A
Returns	All the appointments

Description:

Το παραπάνω Web Service χρησιμοποιείται στο route admin_appointments.jsp που είναι το route με το οποίο ο admin διαχειρίζεται όλες τις κρατήσεις του. Πιο συγκεκριμένα παίρνει όλες τις κρατήσεις που υπάρχουν στο σύστημα και τις επιστρέφει σε μορφή JSON Array.

10)

getAppointments

Description	Gets all the users appointments and returns them as a JSON Array
Endpoint URL	http://localhost:8080/BarberShop/rest/GetAppointment/get_appointments/{userID}
HTTP Method	GET

Parameters	Users userID
Returns	All the users appointments

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route view_appointments.jsp που είναι το route με το οποίο ο users βλέπει όλες τις κρατήσεις του. Πιο συγκεκριμένα παίρνει όλες τις κρατήσεις του User που υπάρχουν στο σύστημα και τις επιστρέφει σε μορφή JSON Array.

11)

getEmails

Description	Gets all the emails that are subscribed to the newsletter and returns them as a JSON Array
Endpoint URL	http://localhost:8080/BarberShop/rest/GetEmails/get_emails
HTTP Method	GET
Parameters	N/A
Returns	All the emails

Description:

Το παραπάνω Web Service χρησιμοποιείται στο route του admin emails.jsp που είναι το route με το οποίο ο admin βλέπει όλα τα email. Πιο συγκεκριμένα παίρνει όλα τα email που υπάρχουν στη βάση δεδομένων και τις επιστρέφει σε μορφή JSON Array.

12) **getUserID**

Description	Gets the userID from the userstable which contains info about the user based on the uuid
Endpoint URL	http://localhost:8080/BarberShop/rest/GetuserID/get_userID/{uuid}
HTTP Method	GET
Parameters	Users or admins uuid
Returns	Returns the userID

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route home.jsp and adminHome.jsp. Πιο συγκεκριμένα παίρνει το uuid που γίνεται inserted στο isloggedin table όταν κάνει κάποιος log in και με βάση αυτό παίρνει το userID και δημιουργεί ένα cookie έτσι ώστε να είναι accessible από όλα τα routes. Αρά κάθε φορά που πραγματοποιεί κάποιος log in δημιουργούνται δυο cookie με το userID and uuid του έτσι ώστε να μπορεί να τσεκάρει το σύστημα αν είναι loggedIn και το role τους αν είναι permitted στο route αυτό.

13) **getUser
names**

Description	Gets all the usernames of the users who are registered in the system
Endpoint URL	http://localhost:8080/BarberShop/rest/GetUsernames/get_usernames
HTTP Method	GET
Parameters	N/A
Returns	Returns a JSON Array of all the usernames

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route usernames.jsp. Πιο συγκεκριμένα παίρνει όλα τα usernames των admin and users who are registered in the system και τα επιστρέφει σε μορφή JSON Array .

14) **loggedIn
Users**

Description	Checks if a user is logged in, if not then takes the userID and the role and inserts them into the table of loggedInUsers
Endpoint URL	http://localhost:8080/BarberShop/rest/LoggedInUsers/new_loggedInUser/{user_uuid}
HTTP Method	GET
Parameters	Users or admins uuid
Returns	Successful message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route Login.jsp. Πιο συγκεκριμένα παίρνει το uuid και αρχικά προσπαθεί να δημιουργήσει το table isloggedIn αν δεν υπάρχει έπειτα βλέπει αν είναι ήδη μέσα στο σύστημα ο χρήστης και επιστρέφει ανάλογο μήνυμα και αν δεν είναι παίρνει το userID και role του χρήστη και τα βάζει μέσα στο table isloggedIn και επιστρέφει μήνυμα επιτυχίας.

15) **loginUse
r**

Description	Takes the input from the user or admin in the login form and checks if the credentials exist and if they exist it returns the uuid
-------------	--

Endpoint URL	http://localhost:8080/BarberShop/rest/Login/login_user/{username}/{password}
HTTP Method	GET
Parameters	Input from register form: username password
Returns	Users or admins uuid

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route Login.jsp. Πιο συγκεκριμένα παίρνει το input από το login form και checks άμα ισχύουν τα credentials και αν ναι γυρνάει το uuid ή αλλιώς γυρνάει μήνυμα αποτυχίας.

16) **logoutUser**

Description	Takes the uuid and removes the user or admin from the isloggedIn table
Endpoint URL	http://localhost:8080/BarberShop/rest/LogoutUser/logout/{user_uuid}
HTTP Method	GET
Parameters	Users or admins uuid
Returns	Successful Message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route adminHome.jsp και home.jsp και παίρνει το uuid που παίρνει το πρόγραμμα από το cookie που είχε δημιουργηθεί και με βάση αυτό αφαιρεί το χρήστη ή διαχειριστή από τους loggedIn χρήστες του συστήματος . 17) **newsletter**

Description	Takes the email that the user inputted and adds it to the emails list
Endpoint URL	http://localhost:8080/BarberShop/rest/Newsletter/new_email/{email}
HTTP Method	GET
Parameters	Users email
Returns	Successful Message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route Newsletter.jsp, αρχικά προσπαθεί να δημιουργήσει το table emails αν δεν υπάρχει και παίρνει το email που έβαλε ο χρήστης στην φόρμα και την προσθέτει στο email_list.

18) registeringUser

Description	Takes the input from register form and inserts new user into database of users
Endpoint URL	http://localhost:8080/BarberShop/rest/Register/new_user/{username}/{password}/{role}
HTTP Method	GET
Parameters	Input from register form:username,password,role
Returns	Successful Message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route Register.jsp, αρχικά προσπαθεί να δημιουργήσει το table userstable αν δεν υπάρχει και παίρνει το input από την φόρμα και εισάγει τα δεδομένα στο παραπάνω table και με βάση το maxuserId επίσης προσθέτει ένα userId στον χρήστη ή διαχειριστή. Αν υπάρχει χρήστης ήδη με ίδιο username ή password επιστρέφεται ανάλογο μήνυμα έτσι ώστε να μην υπάρχουν duplicates. Αν τα δεδομένα εισέλθουν στο database επιστρέφεται μήνυμα επιτυχίας.

19)**updateAppointment**

Description	Takes the input from the update appointment form and based on the appointmentId inserted it updates the appointment with the new information that was inserted
Endpoint URL	http://localhost:8080/BarberShop/rest/Updateappointment/update_appointment/{date}/{haircut}/{time}/{userID}/{price}/{appointmentID}
HTTP Method	GET
Parameters	Input from update appointment and from cookies form:appointmentID,date,haircut,time,userID,price that is calculated based on input
Returns	Successful Message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route updateAppointment.jsp, αρχικά προσπαθεί να δημιουργήσει το table appointments αν δεν υπάρχει και παίρνει το input από την φόρμα και κάνει update τα δεδομένα στο παραπάνω table με βάση το appointmentID. Αν δεν υπάρχει appointment με το ID αυτό επιστρέφεται ανάλογο μήνυμα.

20)

updateappointmentAdmin

Descri ption	Takes the input from the admin update appointment form and based on the appointmentId inserted it updates the appointment with the new information that was inserted
Endpo int URL	http://localhost:8080/BarberShop/rest/UpdateappointmentAdmin//update_appointment_admin/{date}/{haircut}/{time}/{appointmentID}/{price}
HTTP Meth od	GET
Para meter s	Input from update appointment and from cookies form:appointmentID,date,haircut,time,userID,price that is calculated based on input
Retur ns	Successful Message

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route updateAppointmentAdmin.jsp, αρχικά προσπαθεί να δημιουργήσει το table appointments αν δεν υπάρχει και παίρνει το input από την φόρμα και κάνει update τα δεδομένα στο παραπάνω table με βάση το appointmentID. Αν δεν υπάρχει appointment με το ID αυτό επιστρέφεται ανάλογο μήνυμα.

21) viewDetails

Descriptio n	Takes the input from the view_appointments form and returns the firstname and price that are associated with that appointment
Endpoint URL	http://localhost:8080/BarberShop/rest/ViewDetails/get_details/{appointmentID}
HTTP Method	GET
Parameter s	Input from form:appointmentID

Returns	JSON Array with firstname and price of appointment
---------	--

Description:

Το παραπάνω Web Service χρησιμοποιείται στα route view_appointments.jsp, με βάση το appointmentID που εισάγει ο χρήστης γυρνάει λεπτομέρειες για την κράτηση.

ΕΝΟΤΗΤΑ 3: ΕΓΧΕΙΡΙΔΙΟ ΧΡΗΣΗΣ ΣΥΣΤΗΜΑΤΟΣ

ΠΕΡΙΓΡΑΦΗ ΤΕΧΝΟΛΟΓΙΩΝ:

BACKEND:

Java: javac 1.8.0_271

<https://www.filehorse.com/download-java-development-kit-64/55825/>

https://download.cnet.com/Java-Runtime-Environment-JRE/3000-2213_410009607.html

OS: Windows 10.0.19042 Build 19042

Eclipse: Version: 2020-06 (4.16.0)

<https://www.eclipse.org/downloads/packages/release/2020-06/r/eclipse-ideenterprise-java-developers>

Tomcat: Tomcat v9.0

Version: 9.0.58

MYSQL : 8.0 JARS added:

mysql-connector-java-8.0.28

jsr311-api-1.1.1 jersey-

bundle-1.19.4 json-

20211205 **FRONTEND:**

JSP

HTML

CSS

BOOTSTRAP v5.1.3

ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ:

```
mysql> show databases;
+-----+
| Database |
+-----+
| barbershop |
| information_schema |
| mysql |
| performance_schema |
| sakila |
| sys |
| usersdatabase |
| world |
+-----+
8 rows in set (0.01 sec)
```

BarberShop Database:

Description:

Περιέχει δυο tables το appointments and emails. Το appointments περιέχει όλες τις κρατήσεις και το emails όλα τα email που είναι subscribed στο newsletter.

Tables:

```
mysql> show tables;
+-----+
| Tables_in_barbershop |
+-----+
| appointments |
| emails |
+-----+
2 rows in set (0.03 sec)
```

Appointments table:

Description:

Περιέχει για κάθε κράτηση το appointmentID,userID,time,haircut type,price,Firstname γενικά έχει όλες τις πληροφορίες που εισάγει ο χρήστης για να κάνει κράτηση την τιμή της κράτησης αυτής ,το όνομα του ατόμου στο οποίο γίνεται η κράτηση και έχει το userID που έκανε την κράτηση αυτή.

```
mysql> SELECT * from appointments;
```

appointmentID	userID	time	date	haircutChosen	price	Firstname
1	12	14.00	29.03.2022	extreme	15	jason
2	9	13.00	30.02.2000	extreme	15	jason43
3	9	15.00	25.03.2022	normal	9	jason2

```
3 rows in set (0.00 sec)
```

Emails table:

Description:

Περιέχει όλα τα email από τους χρήστες που έκαναν subscribe και κρατάει και ένα count πόσα είναι τα email αυτά.

```
mysql> SELECT * from emails;
```

No_emails	Email_list
1	yo@gmail.com
2	yo2@gmail.com
3	test@test
4	jason@gmail.com
5	yes@yes
6	jason@jason

```
6 rows in set (0.00 sec)
```

Usersdatabase:

Description:

Περιέχει όλα τα δεδομένα των χρηστών και διαχειριστών του συστήματος και κρατάει ποιοι είναι logged in. Πιο συγκεκριμένα έχει δυο tables το userstable and loggedinusers.

```
mysql> show tables;
```

Tables_in_usersdatabase
loggedinusers
userstable

```
2 rows in set (0.00 sec)
```

Loggedinusers table:

Description:

Περιέχει πληροφορίες για κάθε χρήστη που είναι logged in στο σύστημα και κρατάει σύνολο το πόσα άτομα είναι logged in.

```
mysql> SELECT * from loggedinusers;
```

NoUsers	Uuid	userID	Role
2	159fe944-5ef8-4ef7-9f99-f39628a617b5	2	administrator
3	6e744a2b-cdf4-4dc9-86a5-5914db3ff9bb	11	user
5	acd5f13b-9973-4899-b1fb-8747d1130589	12	user
6	c39b3036-5f8a-426c-a9f9-e79a66ed423e	9	administrator

```
4 rows in set (0.00 sec)
```

Usertable table:

Description:

Περιέχει το username and password κάθε χρήστη και διαχειριστή όπως επίσης έχει και τον ρόλο και κάθε φορά που κάνει register κάποιος τα δεδομένα εισέρχονται εδώ και γίνεται generate ένα uuid και userID για τον χρήστη αυτό.

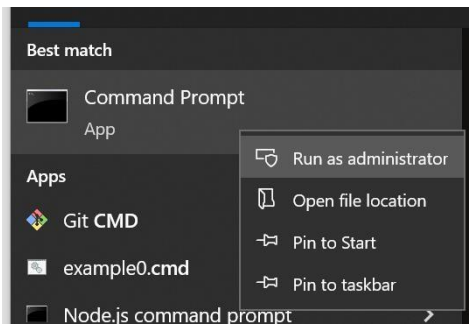
```
mysql> SELECT * from usertable;
```

userID	Username	Password	Uuid	Role
2	yoyooy	yyyy	159fe944-5ef8-4ef7-9f99-f39628a617b5	administrator
3	admin	admin	60b3cb40-0d4f-4663-b32d-e48cc62d48de	user
4	bill	bill	81e829d6-d01b-4348-b365-aaa021f9855c	user
5	billy	billy	b24c1ac3-a007-470e-a5db-725e834f6cc9	user
8	amber	amber	5b448579-52c5-41fe-bc35-3cc044be25b3	user
9	jason2	jason2	c39b3036-5f8a-426c-a9f9-e79a66ed423e	administrator
10	jason1	jason1	e68e46b6-18d9-443b-b303-d05d79c2f30f	administrator
11	jason4	jason4	6e744a2b-cdf4-4dc9-86a5-5914db3ff9bb	user
12	jason	jason	acd5f13b-9973-4899-b1fb-8747d1130589	user

```
9 rows in set (0.00 sec)
```

ΟΔΗΓΙΕΣ ΕΓΚΑΤΑΣΤΑΣΗΣ ΚΑΙ ΕΚΤΕΛΕΣΗΣ ΣΥΣΤΗΜΑΤΟΣ:

ΕΚΚΙΝΗΣΗ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ :



```
Administrator: Command Prompt
C:\WINDOWS\system32>net start MySQL80
The MySQL80 service is starting.
The MySQL80 service was started successfully.

C:\WINDOWS\system32>
```

EKKINH2H Login.jsp file:

```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1" %>
2 pageEncoding="ISO-8859-1"%>
3 <%
4 page import="com.sun.jersey.api.client.Client"%>
5 page import="com.sun.jersey.api.client.ClientResponse"%>
6 page import="com.sun.jersey.api.client.WebResource"%>
7
8 <%@doctype html%>
9 <html>
10 <head>
11 <meta charset="ISO-8859-1"%>
12 <title>Login</title>
13 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ke7815Na20312g797czd1jAk2n2qH28lpeTf7bHQ1/qBVyyp8tKd8Q63NH4L7aCB" %>
14 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1C3ABW/GJ7/Ptd+bzYF1x6bZ6B2GGY1l2D6fKv3JF16PEBCVv95vp0R663408Pvt4" %>
15 </head>
16 <body>
17 <div class="container">
18 <div class="row">
19 <div class="col">
20 <div class="card">
21 <div class="card-body">
22 <div class="text-center">
23 <h2>Login</h2>
24 <div class="mb-3">
25 <div class="text">Username</div>
26 <input type="text" class="form-control">
27 </div>
28 <div class="mb-3">
29 <div class="text">Password</div>
30 <input type="password" class="form-control">
31 </div>
32 <div class="text-center">
33 <button type="submit" class="btn btn-primary">Submit</button>
34 <button type="button" class="btn btn-primary">Register</button>
35 </div>
36 </div>
37 </div>
38 </div>
39 </div>
40 </body>
41 </html>
```

USER Tutorial:

Login interface:

Login

Username

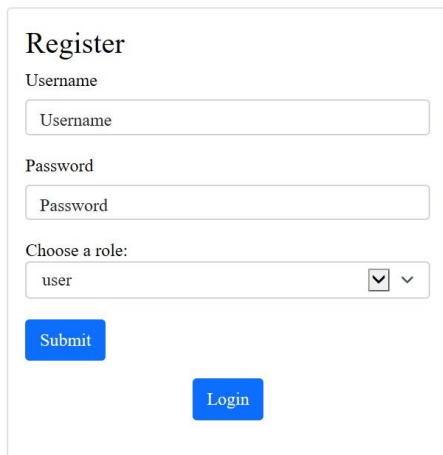
Password

Submit

Register

Hit the register button

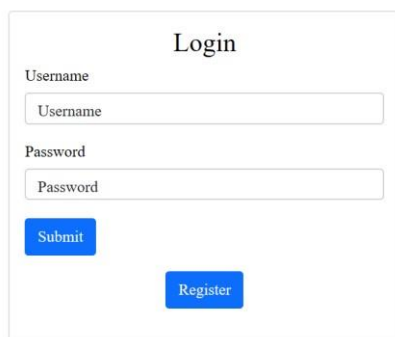
Registering an account:



The Register form is a white rectangular box with a light gray border. At the top, the title 'Register' is displayed in a bold, black, sans-serif font. Below the title, there are three input fields: 'Username', 'Password', and 'Choose a role:'. Each field has a light gray border and a placeholder text of the same name. The 'Choose a role:' field is a dropdown menu with a small downward arrow on the right. Below the input fields, there are two blue buttons with white text: 'Submit' and 'Login'. The 'Submit' button is positioned to the left of the 'Login' button.

Εισαγάγετε τα πεδία που εμφανίζονται με τα credentials που επιθυμείτε και επιλέξτε τον ρόλο user, αφού πατήσετε το κουμπί Submit θα ανακατευθυνθείτε στην οθόνη Login, εάν δεν ανακατευθυνθείτε υπήρξε σφάλμα.

Logging In:



The Login form is a white rectangular box with a light gray border. At the top, the title 'Login' is displayed in a bold, black, sans-serif font. Below the title, there are two input fields: 'Username' and 'Password'. Each field has a light gray border and a placeholder text of the same name. Below the input fields, there are two blue buttons with white text: 'Submit' and 'Register'. The 'Submit' button is positioned to the left of the 'Register' button.

Εισαγάγετε στα πεδία που εμφανίζονται τα credentials που εισήγατε πριν και πατήσετε το κουμπί Submit. Αν όλα πήγαν σωστά θα ανακατευθυνθείτε στο Home page, εάν δεν ανακατευθυνθείτε υπήρξε σφάλμα.

Home page:

Welcome to our barbershop

All available routes

[View more information](#)

[Subscribe to our newsletter](#)

[Book an appointment](#)

[View my appointments](#)

[Update my appointment](#)

[Delete my appointment](#)

Logout

Στην σελίδα αυτή βλέπουμε όλα τα available actions του user και επίσης ένα κουμπί Logout αν θέλουν να πραγματοποιήσουμε έξοδο από το σύστημα.

Αν πατήσουμε στο link **View more information** θα προκύψουν περισσότερες πληροφορίες για την επιχείρηση. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

About Us

Groovin Barber

Our information

Barber shop in Agia Paraskevi

Address: Patriarchou Grigoriou E 21, Ag. Paraskevi 153
41

[Call us at +21 1408 6493](#)

Back to Home Page

Αν πατήσουμε στο link **Subscribe to our newsletter** θα μας δοθεί ένα πεδίο που μπορούμε να εισάγουμε το email μας για το newsletter της επιχείρησης. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

Sign up to our newsletter

Enter you email

Enter your email

Submit

Back to Home Page

Αν πατήσουμε στο link **Book an appointment** θα μας δοθεί ένα πεδίο που μπορούμε να εισάγουμε την ημερομηνία, όνομα, είδος κουρέματος και την ώρα που θα θέλαμε να πραγματοποιήσουμε μια κράτηση, για να πραγματοποιήσουμε την κράτηση πατάμε το κουμπι Submit . Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

Book your appointment

Enter the date

DD.MM.YYYY

Enter your Firstname

Firstname

Choose type of haircut:

normal

Choose a time:

8.00

Submit

Back to Home Page

Αν πατήσουμε στο link **View my appointments** θα μας δοθεί ένα πεδίο που μπορούμε να εισάγουμε το appointmentID μιας κράτησης μας και κάνοντας αυτό θα μας δοθούν λεπτομέρειες για την κράτηση αυτή. Επίσης εμφανίζονται όλες οι κρατήσεις που έχουμε κάνει και με τις πληροφορίες που είχαμε εισάγει για αυτές. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

My appointments

Enter appointmentID to view details for appointment

Enter the appointmentID

View Details

Date	Time	Haircut	AppointmentID
29.03.2022	14.00	extreme	1

Price	Firstname
15	jason

Back to Home Page

Αν πατήσουμε στο link **Update my appointment** θα μας δοθεί ένα πεδίο που μας δίνει την δυνατότητα να ενημερώσουμε μια από τις κρατήσεις μας. Αυτό το καταφέρνουμε εισάγοντας την ημερομηνία, όνομα , είδος κουρέματος και την ώρα και το appointmentID της κράτησης και με βάση το appointmentID που θα εισάγουμε θα ενημερωθεί η κράτηση. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

Update appointment based on appointmentID

Enter the appointmentID

Enter the date

Choose type of haircut:

normal

Choose a time:

8.00

Submit

Back to Home Page

Αν πατήσουμε στο link **Delete my appointment** θα μας δοθεί ένα πεδίο που μας δίνει την δυνατότητα να διαγράψουμε μια από τις κρατήσεις μας. Αυτό το καταφέρνουμε εισάγοντας το appointmentID της κράτησης και με βάση το appointmentID που θα εισάγουμε θα διαγράφει η κράτηση. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

Delete appointment based on appointmentID

Enter the appointmentID

Date	Time	Haircut	AppointmentID
29.03.2022	14.00	extreme	1

Τέλος αν πατήσουμε το κουμπί **Logout** το σύστημα μας ανακατευθύνει στο Login page.

ADMIN Tutorial:

Login interface:

Login

Username

Password

Hit the register button

Registering an account:

Register

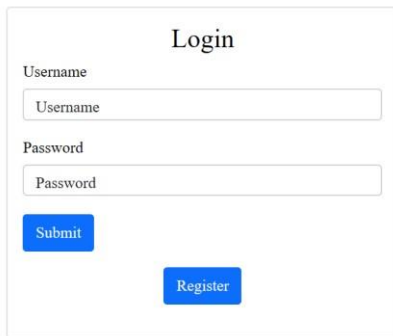
Username

Password

Choose a role:

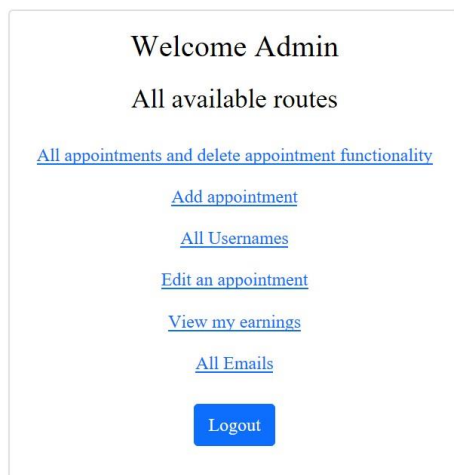
Εισαγάγετε τα πεδία που εμφανίζονται με τα credentials που επιθυμείτε και επιλέξτε τον ρόλο administrator, αφού πατήσετε το κουμπί Submit θα ανακατευθυνθείτε στην οθόνη Login, εάν δεν ανακατευθυνθείτε υπήρξε σφάλμα.

Logging In:



A login form titled "Login". It contains two input fields: "Username" and "Password". Below the "Password" field is a blue "Submit" button. At the bottom of the form is a blue "Register" button.

Εισαγάγετε στα πεδία που εμφανίζονται τα credentials που εισήγατε πριν και πατήσετε το κουμπί Submit. Αν όλα πήγαν σωστά θα ανακατευθυνθείτε στο adminHome page, εάν δεν ανακατευθυνθείτε υπήρξε σφάλμα. **adminHome page:**



An admin home page titled "Welcome Admin". Below the title is the text "All available routes". There are several blue underlined links: "[All appointments and delete appointment functionality](#)", "[Add appointment](#)", "[All Usernames](#)", "[Edit an appointment](#)", "[View my earnings](#)", and "[All Emails](#)". At the bottom is a blue "Logout" button.

Στην σελίδα αυτή βλέπουμε όλα τα available actions του administrator και επίσης ένα κουμπί Logout αν θέλουν να πραγματοποιήσουμε έξοδο από το σύστημα.

Αν πατήσουμε στο link **All appointments and delete appointment functionality** θα μας δοθεί ένα πεδίο που μας δίνει την δυνατότητα να διαγράψουμε μια από τις κρατήσεις. Αυτό το καταφέρνουμε εισάγοντας το appointmentID της κράτησης και με βάση το appointmentID που θα εισάγουμε θα διαγράφει η κράτηση. Επίσης θα μας εμφανιστούν όλες οι κρατήσεις που έχουν πραγματοποιηθεί στο σύστημα μας. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

All appointments

Enter the appointmentID

Delete

Date	Time	Haircut	AppointmentID
29.03.2022	14.00	extreme	1

Date	Time	Haircut	AppointmentID
30.02.2000	13.00	extreme	2

Date	Time	Haircut	AppointmentID
25.03.2022	15.00	normal	3

Back to Home Page

Αν πατήσουμε στο link **Add appointment** θα μας δοθεί ένα πεδίο που μπορούμε να εισάγουμε την ημερομηνία, όνομα, είδος κουρέματος και την ώρα που θα θέλαμε να πραγματοποιήσουμε μια κράτηση, για να προσθέσουμε την κράτηση πατάμε το κουμπί Submit. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

Add an appointment

Enter the date

Enter a Firstname

Choose type of haircut:

Choose a time:

Submit

Back to Home Page

Αν πατήσουμε στο link **All Usernames** θα μας εμφανιστούν όλα τα usernames των χρηστών και διαχειριστών του συστήματος. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

Total Projected Earnings

Total earnings: 39\$

[Back to Home Page](#)

Αν πατήσουμε στο link **All Emails** θα μας εμφανιστούν όλα τα emails των χρηστών που έχουν κάνει subscribe στο newsletter. Αν θέλουμε να γυρίσουμε στην αρχική σελίδα πατάμε το κουμπί back to home page:

All emails

Email	yo@gmail.com
Email	yo2@gmail.com
Email	test@test
Email	jason@gmail.com
Email	yes@yes
Email	jason@jason

[Back to Home Page](#)

Τέλος αν πατήσουμε το κουμπί **Logout** το σύστημα μας ανακατευθύνει στο Login page.

ΔΙΕΠΑΦΕΣ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΠΟΥ ΧΡΗΣΙΜΟΠΟΠΟΥΝΤΑΙ ΣΕ ΚΑΘΕ JSP ΑΡΧΕΙΟ:

Register.jsp:

On loading the page:

Creating usersdatabase if it doesn't exist:

```

creating user table if doesnt exist
try{
    Client client = Client.create();
    WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Creating/usersDatabase");
    ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
    String output = myresponse.getEntity(String.class);
    System.out.println(output);
}

```

If the submit button gets pressed:

```

2 //taking info that was submitted and creating new account
3
4 if(request.getParameter("btn") != null){
5
6     String Username = request.getParameter("Username");
7     String Password = request.getParameter("Password");
8     String Role = request.getParameter("role");
9     try{
10         Client client = Client.create();
11         WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Register/new_user/"+Username+"/"+Password+"/"+Role);
12         ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
13     }
14 }

```

Taking the input:

Trying to create table:

```

System.out.println("Connecting to database...");
conn = DriverManager.getConnection(DB_URL, USER, PASS);

System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE usersTable (userID INT(50),Username VARCHAR(50),Password VARCHAR(50),Uuid VARCHAR(50),Role VARCHAR(50))");
ps.executeUpdate();
System.out.println("Table created Successfully");
} catch(SQLException | ClassNotFoundException e) {
}

```

Taking maxuserID:

```

System.out.println("Connecting to database...");
conn = DriverManager.getConnection(DB_URL, USER, PASS);
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM userstable WHERE userID=(SELECT max(userID) FROM userstable)");
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    number = rs.getInt("userID");
    System.out.println(number);
}

```

If user exists with same credentials returning fail:

```

PreparedStatement ps = conn.prepareStatement("SELECT * FROM userstable WHERE Username=? OR Password=?");
ps.setString(1, username);
ps.setString(2, password);
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    found_user = rs.getString("Username");
}
ps.close();
if(found_user.equals("") != true) {
    return "Fail";
}

```

If user doesn't exist already with those credentials inserting data into userstable and redirecting user to login page:

```

//Executing query
System.out.println("Inserting data...");
PreparedStatement ps = conn.prepareStatement("INSERT INTO usersTable (userID, Username, Password, Uuid, Role) VALUES (?, ?, ?, ?, ?)");
ps.setInt(1, userID);
ps.setString(2, Username);

```

Register

Username

Password

Choose a role:

user ▼

Submit

```
mysql> SELECT * from userstable;
```

	userID	Username	Password	Uuid	Role
2	yoyooy	yyyy	159fe944-5ef8-4ef7-9f99-f39628a617b5	administrator	
3	admin	admin	60b3cb40-0d4f-4663-b32d-e48cc62d48de	user	
4	bill	bill	81e829d6-d01b-4348-b365-aaa021f9855c	user	
5	billy	billy	b24c1ac3-a007-470e-a5db-725e834f6cc9	user	
8	amber	amber	5b448579-52c5-41fe-bc35-3cc044be25b3	user	
9	jason2	jason2	c39b3036-5f8a-426c-a9f9-e79a66ed423e	administrator	
10	jason1	jason1	e68e46b6-18d9-443b-b303-d05d79c2f30f	administrator	
11	jason4	jason4	6e744a2b-cdf4-4dc9-86a5-5914db3ff9bb	user	
12	jason	jason	acd5f13b-9973-4899-b1fb-8747d1130589	user	
13	jason5	jason5	f980872a-3b44-42ae-b2d1-bf2ef81223fd	user	
14	yoyooyoyo	yoyooyoyo	0087c77b-8d29-42ff-bffc-f970cd2ac660	user	
15	jason7	jason7	155cd5ab-3cf0-49c8-bd56-190ec799f653	user	
16	jason7	jason7	155cd5ab-3cf0-49c8-bd56-190ec799f653	user	

```
System.out.println(output);
if(output.equals("Fail") != true){
String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
response.sendRedirect(redirectURL);
}
```

Login.jsp:

On loading the page:

Creating usersdatabase if it doesn't exist:

```
try{
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Creating/usersDatabase");
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output = myresponse.getEntity(String.class);
System.out.println(output);
}
```

If the submit button gets pressed:

```
try{
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Login/login_user/"+Username+"/"+Password);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
}
```

Taking the input:

Username

Password

Submit

Checking from table userstable if credentials are correct:

```
PreparedStatement ps = conn.prepareStatement("SELECT * FROM userstable WHERE Username=? AND Password=?");
ps.setString(1, x);
ps.setString(2, y);
ResultSet rs = ps.executeQuery();
```

If credentials are correct:

```
try {
    Client client1 = Client.create();
    WebResource webResource1 = client1.resource("http://localhost:8080/BarberShop/rest/LoggedInUsers/new_loggedInUser/" + output);
    ClientResponse myresponse1 = webResource1.accept("text/plain").get(ClientResponse.class);
```

Trying to create loggedInUsers table:

```
System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE loggedInUsers (NoUsers INT(50),Uuid VARCHAR(50),userID INT(50),Role VARCHAR(50))");
ps.executeUpdate();
System.out.println("Table created Successfully");
```

```
mysql> SELECT * from loggedInusers;
```

NoUsers	Uuid	userID	Role
2	159fe944-5ef8-4ef7-9f99-f39628a617b5	2	administrator
3	6e744a2b-cdf4-4dc9-86a5-5914db3ff9bb	11	user
6	c39b3036-5f8a-426c-a9f9-e79a66ed423e	9	administrator

3 rows in set (0.00 sec)

Taking maxNoUsers:

```
conn = DriverManager.getConnection(DB_URL, USER, PASS);
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE NoUsers=(SELECT max(NoUsers) FROM loggedInUsers)");
ResultSet rs = ps.executeQuery();
```

Checking if user already logged in:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE Uuid=?");
ps.setString(1, user_uuid);
ResultSet rs = ps.executeQuery();
```

If he isn't taking userID and role of user from table userstable and inserting him into loggedInUsers:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM userstable WHERE Uuid=?");
ps.setString(1, user_uuid);
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    found_userID = rs.getInt("userID");
    found_role = rs.getString("Role");
    System.out.println(found_userID);
```

```
System.out.println("Inserting data...");
PreparedStatement ps = conn.prepareStatement("INSERT INTO loggedInUsers (NoUsers, Uuid, userID,Role) VALUES (?, ?, ?, ?)");
ps.setInt(1, NoUsers);
```

```
mysql> SELECT * from loggedInusers;
```

NoUsers	Uuid	userID	Role
2	159fe944-5ef8-4ef7-9f99-f39628a617b5	2	administrator
3	6e744a2b-cdf4-4dc9-86a5-5914db3ff9bb	11	user
6	c39b3036-5f8a-426c-a9f9-e79a66ed423e	9	administrator
7	155cd5ab-3cf0-49c8-bd56-199ec799f653	15	user

4 rows in set (0.00 sec)

Redirecting to home page if login successful and creating cookie with user or admin uuid :


```
Cookie cookie = new Cookie("uuid",output);
cookie.setMaxAge(60*60*24);
response.addCookie(cookie);
String redirectURL = "http://localhost:8080/BarberShop/home.jsp";
response.sendRedirect(redirectURL);
```

home.jsp:

On loading:

Getting uuid from cookies:

```
// Get an array of Cookies associated with the request
cookies = request.getCookies();

if( cookies != null ) {

    for (int i = 0; i < cookies.length; i++) {
        cookie = cookies[i];
        String x = cookie.getName();
        if(x.equals("uuid")){
```

Trying to create database barbershop :

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Create/barbershop");
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output = myresponse.getEntity(String.class);
```

Checking if loggedIn with uuid value gotten from cookie:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckIfLoggedIn/uuid_check/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output = myresponse.getEntity(String.class);
```

Checking in table loggedInUsers with uuid value:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE Uuid=?");
ps.setString(1, user_uuid_to_check);
ResultSet rs = ps.executeQuery();
```

If not logged in redirecting to login page:

```
System.out.println(output);
if(output.equals("Fail") || uuid_value == ""){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

If logged in:

Getting userID from loggedInUsers:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetuserID/get_userID/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output = myresponse.getEntity(String.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM userstable WHERE Uuid=?");
ps.setString(1, uuid);
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    new_userID = rs.getInt("userID");
    System.out.println(new_userID);
}
```

Creating cookie with userID:

```
System.out.println(output);
Cookie cookie1 = new Cookie("userID",output);
cookie1.setMaxAge(60*60*24);
response.addCookie(cookie1);
new_userID = output;
```

Getting role of user with userID from loggedInUsers:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckRole/checking_role/"+new_userID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output = myresponse.getEntity(String.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE userID=?");
ps.setInt(1, userID);
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    Role = rs.getString("Role");
    System.out.println(Role);
}
```

If user is administrator redirecting to adminHome page:

```
if(output.equals("user") != true){
    String redirectURL = "http://localhost:8080/BarberShop/adminHome.jsp";
    response.sendRedirect(redirectURL);
}
```

If logout button is pressed:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/LogoutUser/logout/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

Deleting user with uuid value from loggedInUsers:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("DELETE FROM loggedInUsers WHERE Uuid = ?");
ps.setString(1, user_uuid);
ps.executeUpdate();
```

Redirecting him to Login page:

```
String output = myresponse.getEntity(String.class);
System.out.println(output);
String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
response.sendRedirect(redirectURL);
```

adminHome.jsp:

On loading:

Getting userId and users uuid from cookies:

```
cookies = request.getCookies();
if( cookies != null ) {
    for (int i = 0; i < cookies.length; i++) {
        cookie = cookies[i];
        String x = cookie.getName();
        if(x.equals("uuid")){
            uuid_value = cookie.getValue();
        }
        if(x.equals("userID")){
            userID = cookie.getValue();
        }
    }
}
```

If there is nouserId in cookies then getting it with uuid:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetuserID/get_userID/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM userstable WHERE Uuid=?");
ps.setString(1, uuid);
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    new_userID = rs.getInt("userID");
}
```

Checking if user logged in and if he isn't redirecting to login page:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckIfLoggedIn/uuid_check/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output1 = myresponse.getEntity(String.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE Uuid=?");
ps.setString(1, user uuid to check);
```

```
if(output1.equals("Fail") || uuid_value == ""){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

Checking if he is an administrator if he isn't redirecting to Login page:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckRole/checking_role/"+intuserid);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

```
System.out.println(output);
if(output.equals("administrator") != true){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

Trying to create barbershop database:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Create/barbershop");
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

If logout button is pressed:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/LogoutUser/logout/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

Deleting user with uuid value from loggedInUsers:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("DELETE FROM loggedInUsers WHERE Uuid = ?");
ps.setString(1, user_uuid);
ps.executeUpdate();
```

Redirecting him to Login page:

```
String output = myresponse.getEntity(String.class);
System.out.println(output);
String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
response.sendRedirect(redirectURL);
```

User Routes
Info.jsp
Appointment.jsp
Delete_appointment.jsp
View_appointments.jsp
updateAppointment.jsp
Newsletter.jsp

In all the user routes above the checks below happen:

On loading:

Getting userId and users uuid from cookies:

```
cookies = request.getCookies();
if( cookies != null ) {
    for (int i = 0; i < cookies.length; i++) {
        cookie = cookies[i];
        String x = cookie.getName();
        if(x.equals("uuid")){
            uuid_value = cookie.getValue();
        }
        if(x.equals("userID")){
            userID = cookie.getValue();
        }
    }
}
```

Checking if user logged in:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckIfLoggedIn/uuid_check/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output1 = myresponse.getEntity(String.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE Uuid=?");
ps.setString(1, user uuid to check);
```

```
if(output1.equals("Fail") || uuid_value == ""){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

Checking if he is a user if he isn't redirecting to Login page:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckRole/checking_role/"+intuserID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

```
if(output1.equals("user") != true){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

Newsletter.jsp:

If submit button gets pressed:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Newsletter/new_email/"+email);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

Sign up to our newsletter

Enter you email

jason7@gmail.com

Submit

Trying to create emails table:

```
System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE emails (No_emails INT(50),Email_list VARCHAR(50))");
ps.executeUpdate();
```

Taking maxNo_emails:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM emails WHERE No_emails=(SELECT max(No_emails) FROM emails)");
ResultSet rs = ps.executeQuery();
```

Inserting email to email_list:

```
System.out.println("Inserting data...");
PreparedStatement ps = conn.prepareStatement("INSERT INTO emails (No_emails, Email_list) VALUES (?,?)");
ps.setInt(1, No_emails);
ps.setString(2, Email_list);
```

```
mysql> SELECT * from emails;
+-----+-----+
| No_emails | Email_list |
+-----+-----+
| 1 | yo@gmail.com |
| 2 | yo2@gmail.com |
| 3 | test@test |
| 4 | jason@gmail.com |
| 5 | yes@yes |
| 6 | jason@jason |
| 7 | jason7@gmail.com |
+-----+-----+
7 rows in set (0.00 sec)
```

appointment.jsp:

If submit button gets pressed:

BOOK your appointment

Enter the date

Enter your Firstname

Choose type of haircut:
 ☒

Choose a time:
 ☒

Taking the input:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CreateAppointment/new_appointment/"+date+"/"+haircut+"/"+time+"/"+intuserid+"/"+price+"/"+firstname);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

Trying to create table appointments:

```
System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE appointments (appointmentID INT(50), userID INT(50), time VARCHAR(50), date VARCHAR(50), haircutChosen VARCHAR(50), price VARCHAR(50), Firstname VARCHAR(50))");
ps.executeUpdate();
System.out.println("Table created Successfully");
```

Taking maxappointmentID:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM appointments WHERE appointmentID =(SELECT max(appointmentID ) FROM appointments)");
ResultSet rs = ps.executeQuery();
while(rs.next()) {
```

Inserting data into appointment table:

```
System.out.println("Inserting data...");
PreparedStatement ps = conn.prepareStatement("INSERT INTO appointments (appointmentID, userID, time, date, haircutChosen,price,Firstname) VALUES (?,?,?,?,?);");
ps.setInt(1, appointmentID);
ps.setInt(2, userid);
```

```
mysql> SELECT * from appointments;
+-----+-----+-----+-----+-----+-----+-----+
| appointmentID | userID | time | date | haircutChosen | price | Firstname |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | 12 | 14.00 | 29.03.2022 | extreme | 15 | jason |
| 2 | 9 | 13.00 | 30.02.2000 | extreme | 15 | jason43 |
| 3 | 9 | 15.00 | 25.03.2022 | normal | 9 | jason2 |
| 4 | 15 | 12.00 | 30.09.2022 | extreme | 15 | jason7 |
+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

view_appointments.jsp:

On loading:

Getting the users appointments based on userID:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetAppointment/get_appointments/"+intUserID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM appointments WHERE userID=?");
ps.setString(1, userID);
```

If view details button is pressed taking the input:

Enter the appointmentID

[View Details](#)

Date	Time	Haircut	AppointmentID
30.09.2022	12.00	extreme	4

[Back to Home Page](#)

```
if(request.getParameter("view") != null){
    try{
        int s = Integer.parseInt(request.getParameter("appointmentID"));
        System.out.println("yoyooyo"+s);
        Client client = Client.create();
        WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/ViewDetails/get_details/"+s);
        ClientResponse myresponse = webResource.accept("application/json").get(ClientResponse.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM appointments WHERE appointmentID = ?");
ps.setInt(1, appointmentID);
```

```
3 | 9 | 15.00 | 23.03.2022 | normal | 9 | jason2
4 | 15 | 12.00 | 30.09.2022 | extreme | 15 | jason7
```

Price	Firstname
15	jason7

[Back to Home Page](#)

updateAppointment.jsp:

If submit button is pressed:

Take input:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Updateappointment/update_appointment/"+date+"/"+haircut+"/"+time+"/"+intuserID+"/"+price+"/"+appointmentID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

Trying to create table appointments:

```
System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE appointments (appointmentID INT(50), userID INT(50), time VARCHAR(50), date VARCHAR(50), haircutChosen VARCHAR(50), price VARCHAR(50), Firstname VARCHAR(50))");
ps.executeUpdate();
System.out.println("Table created Successfully");
```

Checking if appointment exist and if it exists updating it with the inputted data:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("UPDATE appointments SET time=?, date=?, haircutChosen=?, price=? WHERE appointmentID = ?");
ps.setString(1, time);
ps.setString(2, date);
```

Enter the appointmentID

Enter the date

Choose type of haircut:

Choose a time:

Submit

3	9	15.00	23.05.2022	normal	9	jason2
4	15	11.00	20.05.2022	normal	9	jason7

delete_appointment.jsp:

On loading:

Getting the users appointments based on userID:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetAppointment/get_appointments/"+intUserID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM appointments WHERE userID=?");
ps.setString(1, userID);
```

If delete button is pressed:

Taking input:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Deleteappointment/delete_appointment/"+appointmentID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

Trying to create table appointments:

```
System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE appointments (appointmentID INT(50), userID INT(50), time VARCHAR(50), date VARCHAR(50), haircutChosen VARCHAR(50), price VARCHAR(50), Firstname VARCHAR(50))");
ps.executeUpdate();
System.out.println("Table created Successfully");
```

Checking if appointment exists and deleting it from table appointments:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("DELETE FROM appointments WHERE appointmentID = ?");
ps.setInt(1, appointmentID);
ps.executeUpdate();
```

Enter the appointmentID

Submit

Date	Time	Haircut	AppointmentID
20.05.2022	11.00	normal	4

Back to Home Page

Delete appointment based on appointmentID

Enter the appointmentID

Submit

Back to Home Page

appointmentID	userID	time	date	haircutChosen	price	Firstname
1	12	14.00	29.03.2022	extreme	15	jason
2	9	13.00	30.02.2000	extreme	15	jason43
3	9	15.00	25.03.2022	normal	9	jason2

Admin Routes
Emails.jsp
Admin_appointments.jsp
addAppointment_admin.jsp
Earnings.jsp
Usernames.jsp
updateApppointmentAdmin.jsp

In all the admin routes above the checks below happen:

On loading:

Getting userID and uuid from cookies:

```
//get an array of Cookies associated with the cookies = request.getCookies();
if( cookies != null ) {
    for (int y = 0; y < cookies.length; y++) {
        cookie = cookies[y];
        String x = cookie.getName();
        if(x.equals("uuid")){
            uuid_value = cookie.getValue();
        }
        if(x.equals("userID")){
            userID = cookie.getValue();
        }
    }
}
```

Checking if user logged in and if he isn't redirecting to login page:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckIfLoggedIn/uuid_check/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output1 = myresponse.getEntity(String.class);

System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE Uuid=?");
ps.setString(1, user_uuid_to_check);

if(output1.equals("Fail") || uuid_value == ""){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

Checking if he is an administrator if he isn't redirecting to Login page:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckRole/checking_role/"+intuserID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output = myresponse.getEntity(String.class);

System.out.println(output);
if(output.equals("administrator") != true){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```


admin_appointments.jsp:

On loading:

Getting all the appointments:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetAdminappointments/get_appointments");
ClientResponse myresponse = webResource.accept("application/json").get(ClientResponse.class);

System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM appointments");
ResultSet rs = ps.executeQuery();
```

If delete button is pressed:

```
if(request.getParameter("delete") != null){
    try{
        int x = Integer.parseInt(request.getParameter("appointmentID"));
        Client client = Client.create();
        WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/Deleteappointment_Admin/delete_Appointment_Admin/"+x);
        ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
    }
}
```

Taking the input the appointmentID: Trying

to create table appointments:

```
System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE appointments (appointmentID INT(50), userID INT(50), time VARCHAR(50), date VARCHAR(50), haircutChosen VARCHAR(50))");
ps.executeUpdate();
```

Checking if appointment exists and if it does deleting it from appointments:

Enter the appointmentID

Date	Time	Haircut	AppointmentID
29.03.2022	14.00	extreme	1

Date	Time	Haircut	AppointmentID
30.02.2000	13.00	extreme	2

Date	Time	Haircut	AppointmentID
25.03.2022	15.00	normal	3

appointmentID	userID	time	date	haircutChosen	price	Firstname
1	12	14.00	29.03.2022	extreme	15	jason
2	9	13.00	30.02.2000	extreme	15	jason43

addAppointment_admin.jsp On

loading:

If submit button gets pressed:

Taking input:

```
if(request.getParameter("btn") != null){
    String date = new String(request.getParameter("Date"));
    String time = new String(request.getParameter("time"));
    String haircut = request.getParameter("haircut");
    String firstname = request.getParameter("firstname");
    if(haircut.equals("normal") != true){
        price = String.valueOf(15);
    }else{
        price = String.valueOf(9);
    }
    try{
        Client client = Client.create();
        WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CreateAppointment/new_appointment/"+date+"/"+haircut+"/"+time+"/"+intuserID+"/"+price+"/"+firstname);
        ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
    }
}
```

Trying to create table appointments:

```
System.out.println("Creating a table...");
PreparedStatement ps = conn.prepareStatement("CREATE TABLE appointments (appointmentID INT(50), userID INT(50), time VARCHAR(50), date VARCHAR(50), haircutChosen VARCHAR(50), price VARCHAR(50), Firstname VARCHAR(50))");
ps.executeUpdate();
System.out.println("Table created Successfully");
```

Selecting maxappointmentID and adding appointment:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("UPDATE appointments SET time=?, date=?, haircutChosen=?, price=? WHERE appointmentID = ?");
ps.setString(1, time);
```

Enter the date
25.05.2022

Enter a Firstname
jason8

Choose type of haircut:
extreme

Choose a time:
12.00

Submit

appointmentID	userID	time	date	haircutChosen	price	Firstname
1	12	14.00	29.03.2022	extreme	15	jason
2	9	13.00	30.02.2000	extreme	15	jason43
3	9	12.00	25.05.2022	extreme	15	jason8

Username.jsp:

On loading:

Getting usernames:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetUsernames/get_usernames");
ClientResponse myresponse = webResource.accept("application/json").get(ClientResponse.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT Username FROM userstable");
ResultSet rs = ps.executeQuery();
```

updateAppointmentAdmin.jsp:

On loading:

Getting userID and uuid from cookies:

```
//Get an array of Cookies associated with the cookies = request.getCookies();
if( cookies != null ) {
    for (int y = 0; y < cookies.length; y++) {
        cookie = cookies[y];
        String x = cookie.getName();
        if(x.equals("uuid")){
            uuid_value = cookie.getValue();
        }
        if(x.equals("userID")){
            userID = cookie.getValue();
        }
    }
}
```

Checking if user logged in and if he isn't redirecting to login page:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckIfLoggedIn/uuid_check/"+uuid_value);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
String output1 = myresponse.getEntity(String.class);
```

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT * FROM loggedInUsers WHERE Uuid=?");
ps.setString(1, user_uuid_to_check);
if(output1.equals("Fail") || uuid_value == ""){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

Checking if he is an administrator if he isn't redirecting to Login page:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/CheckRole/checking_role/"+intuserID);
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
```

```
System.out.println(output);
if(output.equals("administrator") != true){
    String redirectURL = "http://localhost:8080/BarberShop/Login.jsp";
    response.sendRedirect(redirectURL);
}
```

If submit button gets pressed:

Taking input:

```
if(request.getParameter("btn") != null){
    String date = new String(request.getParameter("Date"));
    String time = new String(request.getParameter("time"));
    String haircut = request.getParameter("haircut");
    int appointmentID = Integer.parseInt(request.getParameter("appointmentID"));
    if(haircut.equals("normal") != true){
        price = String.valueOf(15);
    }else{
        price = String.valueOf(9);
    }
    try{
        Client client = Client.create();
        WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/UpdateappointmentAdmin/update_appointment_admin/"+date+"/"+haircut+"/"+time+"/"+appointmentID+"/"+price);
        ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);
    }catch (Exception e){
        e.printStackTrace();
    }
}
```

Trying to create appointment table:

Checking if appointment exists and if it does updating it with inputted data based on appointmentID:

```
System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("UPDATE appointments SET time=?, date=?, haircutChosen=?, price=? WHERE appointmentID = ?");
ps.setString(1, time);
```

Enter the appointmentID

Enter the date

Choose type of haircut:

Choose a time:

1	12	14.00	29.03.2022	extreme	15	jason
2	9	10.00	25.08.2022	normal	9	jason43
3	9	12.00	25.05.2022	extreme	15	jason8

Earnings.jsp:

On loading:

Getting the earnings:

```
Client client = Client.create();
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetexpectedEarnings/earnings");
ClientResponse myresponse = webResource.accept("text/plain").get(ClientResponse.class);

System.out.println("Selecting data...");
PreparedStatement ps = conn.prepareStatement("SELECT price FROM appointments");
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    i = i + Integer.parseInt(rs.getString("price"));
}
ps.close();
```

Emails.jsp:

On loading:

Getting the emails:


```
Client client = Client.create();  
WebResource webResource = client.resource("http://localhost:8080/BarberShop/rest/GetEmails/get_emails");  
ClientResponse myresponse = webResource.accept("application/json").get(ClientResponse.class);  
output = myresponse.getEntity(String.class);
```

```
System.out.println("Selecting data...");  
PreparedStatement ps = conn.prepareStatement("SELECT Email_list FROM emails");  
ResultSet rs = ps.executeQuery();  
ResultSetMetaData rsmd = rs.getMetaData();
```