

Πανεπιστήμιο Πειραιώς

Σχολή Τεχνολογιών Πληροφορικής και Τηλεπικοινωνίων

Τμήμα Ψηφιακών Συστημάτων

Επίπεδο: Προπτυχιακό/Μεταπτυχιακό Πρόγραμμα Σπουδών

Μάθημα -Δικτυοκεντρικά Πληροφοριακά Συστήματα

Τίτλος -BoltGym Website

Επιβλέπον Καθηγητής: Βασιλική Κούφη

|  |  |  |
| --- | --- | --- |
| Ονοματεπώνυμο | E-mail | A.M. |

Γιώργος Χειλαδάκης [george.chiladakis@gmail.com](mailto:george.chiladakis@gmail.com) E20188

Νικήτας Ασημομύτης [asinik12402@gmail.com](mailto:asinik12402@gmail.com) E20009

Δημήτρης Κεχαγιόγλου [kechagiogloudimitris@gmail.com](mailto:kechagiogloudimitris@gmail.com) E20070

Πειραιάς

24/1/2024

Περιεχόμενα

[Εισαγωγή 3](#_Toc157342147)

[Περιγραφή του συστήματος που υλοποιήθηκε 4](#_Toc157342148)

[Υλοποίηση συστήματος 6](#_Toc157342151)

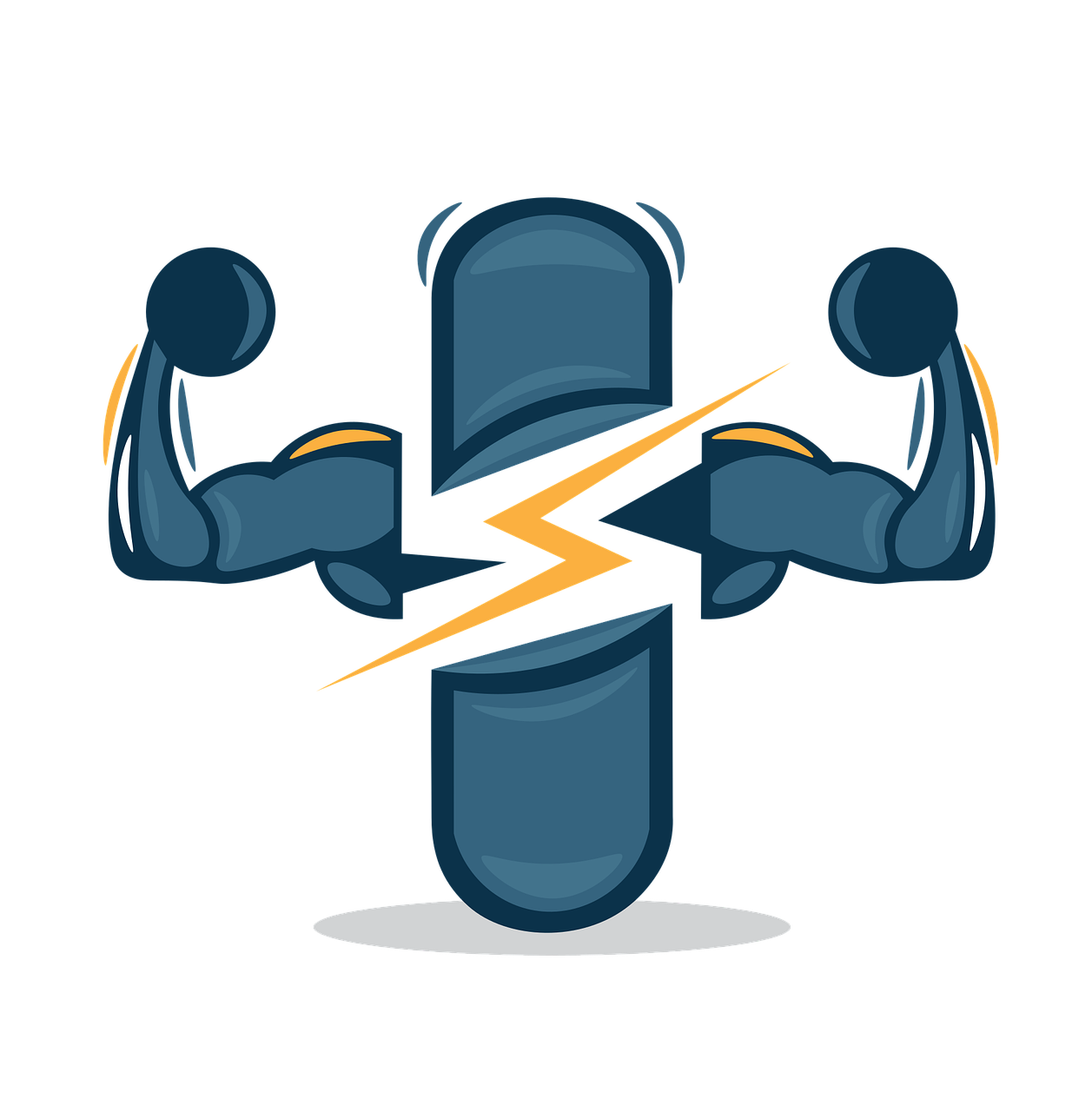
[Εγχειρίδιο διαχειριστή 7](#_Toc157342152)

[Εγχειρίδιο χρήστη 8](#_Toc157342153)

# 

# Εισαγωγή

Η παρούσα αναφορά αποτελεί μια συνοπτική επισκόπηση του κώδικα που έχει αναπτυχθεί για τη δημιουργία μιας διαδικτυακής σελίδας γυμναστηρίου. Σκοπός της σελίδας είναι να παρέχει στους χρήστες μια χρήσιμη και ευχάριστη εμπειρία, προσφέροντας πληροφορίες σχετικά με το γυμναστήριο, τις υπηρεσίες που προσφέρει και παρέχοντας έναν αποτελεσματικό τρόπο για τους χρήστες να αλληλοεπιδρούν με το περιβάλλον του. Ο κώδικας της σελίδας χωρίζεται σε 3 γλώσσες. Αρχικά, έχουμε την HTML και την CSS όπου χρησιμοποιούνται για την στιλιστική μορφοποίηση της σελίδας (γνωστό ως front end) και την Python που χρησιμοποιείται για το backend του συστήματος όπου θα πραγματοποιούνται λειτουργίες όπως σύνδεση με την βάση δεδομένων, εμφάνιση ανακοινώσεων, σύνδεση χρήστη/διαχειριστή. Στη συνέχεια της αναφοράς θα εξεταστεί λεπτομερώς ο κώδικας ,παρέχοντας αναλυτικές ερμηνείες και παραδείγματα. Η αναφορά αυτή στοχεύει να παρουσιάσει έναν σφαιρικό και κατανοητό οδηγό για τον κώδικα της διαδικτυακής σελίδας του γυμναστηρίου.



# Περιγραφή του συστήματος που υλοποιήθηκε

# Το σύστημα που αναπτύχθηκε αποτελεί μια ολοκληρωμένη σελίδα ενός γυμναστηρίου με συγκεκριμένες λειτουργίες. Αρχικά εμφανίζεται ένα κεντρικό μενού στον χρήστη όπου αποτελεί το main page. Συγκεκριμένα η πρώτη σελίδα που του εμφανίζεται έχει να κάνει με τις πληροφορίες σχετικά με το γυμναστήριο όπως το τι υπηρεσίες παρέχει γενικά. Ταυτόχρονα η σελίδα παρέχει ακόμη 4 sections:

1. Announcements
2. Reservation History
3. Reservations
4. Gym Services

Οι 4 αυτές περιπτώσεις που μπορεί να επιλέξει ο χρήστης θα αναλυθούν παρακάτω με μεγαλύτερη έκταση.

# Στην εικόνα 1.1 διακρίνουμε την αρχική αυτή σελίδα.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, λογισμικό πολυμέσων

Περιγραφή που δημιουργήθηκε αυτόματα

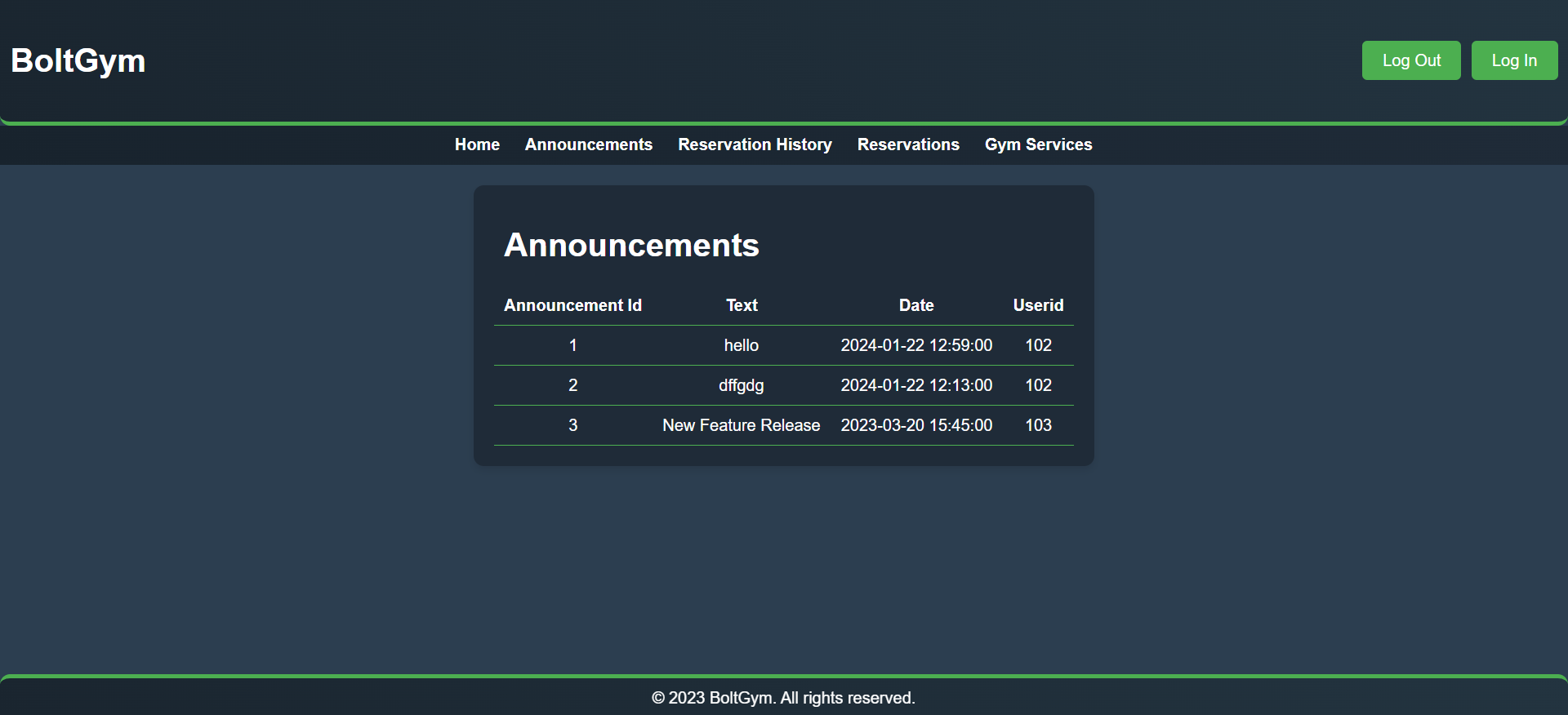
Εικόνα 1.1 Home Page του BoltGym

# Περίπτωση που επιλεχθεί η σελίδα Announcements:

Η σελίδα Announcements αποτελεί ένα μέρος της σελίδας όπου αναρτώνται όλες οι τελευταίες ανακοινώσεις και εξελίξεις για το γυμναστήριο. Σημαντικό να αναφερθεί είναι ότι οι μόνοι που μπορούν να τροποποιήσουν, να σβήσουν η να ανεβάσουν ανακοινώσεις είναι οι διαχειριστές της σελίδας. Ωστόσο το κομμάτι των users και admins θα αναλυθεί περισσότερο στα επόμενα κεφάλαια του report. Αυτό που εμφανίζεται συγκεκριμένα στην σελίδα των ανακοινώσεων είναι ένας πίνακας που χωρίζεται σε 4 κατηγορίες. Αυτές οι κατηγορίες είναι:

1. Announcement Id: όπου αποτελεί το μοναδικό αναγνωριστικό νούμερο κάθε ανακοίνωσης.
2. Text: αποτελεί το κείμενο όπου περιγράφεται κάθε τελευταία ανακοίνωση ή εξέλιξη.
3. Date: ημερομηνία και ώρα ανάρτησης της ανακοίνωσης.
4. User Id: μοναδικός αναγνωριστικός αριθμός του χρήστη που ανέβασε την ανακοίνωση.

Στην Εικόνα 1.2 παρατηρούμε τι μορφή έχει αυτός ο πίνακας ανακοινώσεων.



Εικόνα 1.2 Announcements Page

# Περίπτωση που επιλεχθεί η σελίδα Reservation History:

Η σελίδα Reservation History είναι ένας χώρος όπου «στεγάζεται» το ιστορικό όλων των κρατήσεων που έχουν πραγματοποιηθεί. Εμφανίζεται επίσης ένας πίνακας ο οποίος χωρίζεται σε 4 κατηγορίες. Αυτές είναι:

1. Reservation Id: μοναδικός αναγνωριστικός αριθμός κάθε κράτησης.
2. User Id: μοναδικός αναγνωριστικός αριθμός κάθε χρήστη.
3. Program Id: μοναδικός αναγνωριστικός αριθμός κάθε προγράμματος που έχει επιλέξει κάθε χρήστης.
4. Status: με τον όρο status εννοούμε την κατάσταση στην οποία βρίσκεται μια κράτηση. Οι περιπτώσεις είναι δυο. Να έχει γίνει confirmed από τον διαχειριστή ή να βρίσκεται ακόμη σε κατάσταση pending δηλαδή να μην έχει προβληθεί ακόμα από κάποιον διαχειριστή.

Στην εικόνα 1.3 παρουσιάζεται το τι μορφή έχει η σελίδα του Reservation History.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, αριθμός, διάγραμμα

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα 1.3 Reservation History Page

# Περίπτωση που επιλεχθεί η σελίδα Reservations:

Η σελίδα Reservations αποτελεί το κομμάτι της σελίδας όπου πραγματοποιούνται οι κρατήσεις. Για να πραγματοποιηθεί μια κράτηση ωστόσο κάθε χρήστης πρέπει να περάσει από μια φόρμα Login προκειμένου να συνδεθεί στην πλατφόρμα σαν μέλος. Όταν περάσει από αυτήν την φόρμα εμφανίζεται ένα νέο παράθυρο το οποίο είναι για να πραγματοποιήσει μια κράτηση καθώς και να ακυρωθεί μια κράτηση.

# 

# Περίπτωση που επιλεχθεί η σελίδα Gym Services:

Η σελίδα Gym Services χρησιμοποιείται για την παρουσίαση των προγραμμάτων που παρέχει το γυμναστήριο της BoltGym. Αποτελείται από έναν πίνακα που χωρίζεται σε 3 κατηγορίες. Αυτές είναι οι παρακάτω:

1. Service Id: μοναδικός αναγνωριστικός αριθμός της υπηρεσίας.
2. Service Name: που αποτελεί το όνομα της υπηρεσίας.
3. Trainer Id: μοναδικός αναγνωριστικός αριθμός κάθε Προπονητή που έχει αναλάβει το κάθε πρόγραμμα.

Στην εικόνα 1.3 παρουσιάζεται το τι μορφή έχει η σελίδα του Gym Services.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, λογισμικό πολυμέσων

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα 1.6 Gym Services

# Περίπτωση που επιλεχθεί η σελίδα του Admin:

Στη συνέχεια, όπως είδατε στην εικόνα 1.1 στο πάνω δεξιά μέρος της οθόνης πέρα από το κουμπί του Log in και του Log out, υπάρχει και αυτό του Admin. Ειδικότερα, κατά τη σύνδεση με στοιχεία διαχειριστή μπορούμε να δούμε 4 επιλογές κουμπιών.

Αυτές είναι παρακάτω:

1. Manage Registration Requests

2. Manage Users

3. Manage System Info

4. Manage Announcements

Στην εικόνα 1.7 παρουσιάζεται το τι μορφή έχει η σελίδα του Admin.

A screenshot of a computer

Description automatically generated

Εικόνα 1.7 Admin

# Περίπτωση που επιλεχθεί η σελίδα του Registration Requests:

Η σελίδα Registration Requests είναι το μέρος στο οποίο ο Admin έχει τη δυνατότητα να τα εγκρίνει ή να τα απορρίπτει.

Τα στοιχεία που περιέχονται και που έχει δυνατότητα να δει ο Admin είναι User Id, First Name, Last Name, Country, City,Address, Email,Username,Password,Role.

Στην εικόνα 1.8 παρουσιάζεται το τι μορφή έχει η σελίδα του Registration Requests.

A screenshot of a computer

Description automatically generated

Εικόνα 1.8 Registration Requests

# Περίπτωση που επιλεχθεί η σελίδα του Users:

Η σελίδα Users επιτρέπει στον Διαχειριστή να έχει πρόσβαση στους χρήστες του συστήματος. Ειδικότερα, έχει την ευθύνη να ενημερώνει τα στοιχεία στην βάση δεδομένων καθώς και μπορεί να τροποποιεί δεδομένα, να επεξεργάζεται και να διαγράφει δεδομένα.

Τα στοιχεία που περιέχονται και που έχει δυνατότητα να τροποποιήσει ο Admin είναι User Id, First Name, Last Name, Country, City,Address, Email,Username,Password,Role.

Στην εικόνα 1.9 παρουσιάζεται το τι μορφή έχει η σελίδα του Users.

A screenshot of a computer

Description automatically generated

Εικόνα 1.9 Users

# Περίπτωση που επιλεχθεί η σελίδα του System Info:

Η σελίδα System Info δίνει τη δυνατότητα στον Admin να έχει πρόσβαση στους γυμναστές και στα προγράμματα. Συγκεκριμένα, πατώντας το κουμπί trainers(Εικόνα 1.10 Trainers)επιτρέπεται στον Διαχειριστή να έχει πρόσβαση στους προπονητές του συστήματος. Ειδικότερα, έχει την ευθύνη να ενημερώνει τα στοιχεία στην βάση δεδομένων καθώς και μπορεί να τροποποιεί δεδομένα, να επεξεργάζεται και να διαγράφει δεδομένα. Τα στοιχεία που περιέχονται και που έχει δυνατότητα να τροποποιήσει ο Admin είναι Trainer Id και Trainer Name.

Στην περίπτωση λοιπόν που ο Admin επιλέξει το κουμπί Programs(Εικόνα 1.11 Programs), θα μπορεί ξανά να επεξεργαστεί τα στοιχεία.

Τα στοιχεία που περιέχονται και που έχει δυνατότητα να τροποποιήσει ο Admin είναι Program Id,Program Name και Capacity.

Στην εικόνα 1.9 παρουσιάζεται το τι μορφή έχει η σελίδα του System Info

A screenshot of a computer

Description automatically generated

Εικόνα 1.9 System Info

Στην εικόνα 1.10 παρουσιάζεται το τι μορφή έχει η σελίδα του Trainers

A screenshot of a computer

Description automatically generated

Εικόνα 1.10 Trainers

Στην εικόνα 1.11 παρουσιάζεται το τι μορφή έχει η σελίδα του Programs

A screenshot of a computer

Description automatically generated

Εικόνα 1.11 Programs

# Περίπτωση που επιλεχθεί η σελίδα του Announcements:

Στη σελίδα Announcements ο Admin μπορεί να αναρτά ανακοινώσεις και προσφορές σε όσους χρήστες είναι εγγεγραμμένοι, καθώς και να διαγράφει και να επεξεργάζεται αυτές τις πληροφορίες.

Στην εικόνα 1.12 παρουσιάζεται το τι μορφή έχει η σελίδα του Announcements

A screenshot of a computer

Description automatically generated

Εικόνα 1.12 Announcements

# Υλοποίηση συστήματος

Το παραπάνω κώδικας Python χρησιμοποιεί το Flask για τη δημιουργία ενός web server και αλληλεπίδραση με μια βάση δεδομένων MySQL. Ακολουθεί μια ανάλυση των web services που παρέχονται:

1. **Εγγραφή νέου χρήστη**
   * Λειτουργία: Δέχεται τα στοιχεία ενός νέου χρήστη και τα εισάγει στη βάση δεδομένων.
   * Endpoint: **/register**
   * Μέθοδος Κλήσης: POST
   * Παράμετροι: Τα πεδία του νέου χρήστη (όπως **firstname**, **lastname**, **username**, **password** κλπ.) που αποστέλλονται μέσω φόρμας.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A screen shot of a computer

Description automatically generated

Η δήλωση @app.route('/register', methods=['GET', 'POST']) καθορίζει τη διεύθυνση URL /register για αυτήν τη λειτουργία της εφαρμογής. Αυτή η λειτουργία δέχεται τις μεθόδους GET και POST, πράγμα που σημαίνει ότι μπορεί να κληθεί τόσο για την εμφάνιση της φόρμας όσο και για την υποβολή της.

Η συνάρτηση register() είναι η υλοποίηση για τη διαχείριση του αιτήματος στη διεύθυνση /register.

Η πρώτη συνθήκη ελέγχει αν η μέθοδος του αιτήματος είναι POST, πράγμα που σημαίνει ότι η φόρμα έχει υποβληθεί.

Αν η μέθοδος είναι POST, η εφαρμογή αναζητά τον μέγιστο αριθμό χρήστη από τον πίνακα χρηστών στη βάση δεδομένων. Αυτό γίνεται με την εντολή SQL SELECT MAX(userid) FROM users. Στη συνέχεια, αποθηκεύει τον μέγιστο αριθμό χρήστη στη μεταβλητή max\_userid.

Στη συνέχεια, ελέγχει αν ο max\_userid είναι None, που σημαίνει ότι δεν υπάρχουν καθόλου χρήστες στη βάση δεδομένων. Αν αυτό ισχύει, ορίζει το userid ως 1. Διαφορετικά, αυξάνει το max\_userid κατά ένα και τον αποθηκεύει στη μεταβλητή userid.

Στη συνέχεια, παίρνει τα δεδομένα από τη φόρμα που υποβλήθηκε χρησιμοποιώντας το request.form και τα αποθηκεύει σε μεταβλητές, όπως το firstname, το lastname, το email κλπ.

Μετά τον χειρισμό των δεδομένων από τη φόρμα, εισάγει τα στοιχεία του νέου χρήστη στον πίνακα χρηστών στη βάση δεδομένων, χρησιμοποιώντας την εντολή SQL INSERT INTO.

Τέλος, κάνει redirect στη σελίδα σύνδεσης (login) με χρήση της συνάρτησης redirect().

Αν η μέθοδος του αιτήματος δεν είναι POST, τότε επιστρέφει το πρότυπο της φόρμας εγγραφής (new\_register.html) με τη συνάρτηση render\_template().

1. **Σύνδεση Χρήστη**
   * Λειτουργία: Επιτρέπει σε έναν χρήστη να συνδεθεί στο σύστημα ελέγχοντας τα διαπιστευτήριά του.
   * Endpoint: **/login**
   * Μέθοδος Κλήσης: POST
   * Παράμετροι: Το όνομα χρήστη (**username**) και ο κωδικός πρόσβασης (**password**) που αποστέλλονται μέσω φόρμας.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A computer screen with text on it

Description automatically generated

Η δήλωση @app.route('/login', methods=['GET', 'POST']) καθορίζει τη διεύθυνση URL /login για αυτήν τη λειτουργία της εφαρμογής. Αυτή η λειτουργία δέχεται τις μεθόδους GET και POST, πράγμα που σημαίνει ότι μπορεί να κληθεί τόσο για την εμφάνιση της φόρμας σύνδεσης όσο και για την υποβολή της.

Η συνάρτηση login() είναι η υλοποίηση για τη διαχείριση του αιτήματος στη διεύθυνση /login.

Η πρώτη συνθήκη ελέγχει αν η μέθοδος του αιτήματος είναι POST, πράγμα που σημαίνει ότι η φόρμα έχει υποβληθεί.

Αν η μέθοδος είναι POST, η εφαρμογή παίρνει το όνομα χρήστη και τον κωδικό πρόσβασης από τη φόρμα που υποβλήθηκε χρησιμοποιώντας το request.form και τα αποθηκεύει στις μεταβλητές username και password αντίστοιχα.

Στη συνέχεια, η εφαρμογή εκτελεί μια επερώτηση SQL στον πίνακα χρηστών για να αναζητήσει έναν χρήστη με το συγκεκριμένο όνομα χρήστη και κωδικό πρόσβασης που δόθηκαν.

Η μέθοδος fetchone() επιστρέφει το πρώτο αποτέλεσμα της επερώτησης, δηλαδή τον πρώτο χρήστη που βρέθηκε που ταιριάζει με τα δεδομένα που δόθηκαν.

Αν βρεθεί ένας χρήστης που ταιριάζει, τότε η εφαρμογή θέτει το όνομα χρήστη στη συνεδρία (session) χρησιμοποιώντας το session['username'] = username.

Τέλος, κάνει redirect στην κύρια σελίδα (ή σε οποιαδήποτε άλλη σελίδα που θα οριστεί ως κύρια) με χρήση της συνάρτησης redirect().

Αν η μέθοδος του αιτήματος δεν είναι POST, τότε επιστρέφει το πρότυπο της φόρμας σύνδεσης (login.html) με τη συνάρτηση render\_template().

1. **Αποσύνδεση Χρήστη**
   * Λειτουργία: Αποσυνδέει έναν χρήστη από το σύστημα και καθαρίζει τη συνεδρία του.
   * Endpoint: **/logout**
   * Μέθοδος Κλήσης: GET
   * Παράμετροι: Δεν απαιτούνται παράμετροι.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A screen shot of a computer code

Description automatically generated

Η δήλωση @app.route('/logout') καθορίζει τη διεύθυνση URL /logout για αυτήν τη λειτουργία της εφαρμογής.

Η συνάρτηση logout() είναι η υλοποίηση για τη διαχείριση των αιτημάτων στη διεύθυνση /logout.

Η συνάρτηση is\_user\_authenticated() προφανώς ελέγχει αν ο χρήστης είναι συνδεδεμένος. Αυτή η συνάρτηση δεν δίνεται στον κώδικα που παρέχετε, αλλά πιθανώς ελέγχει κάποιο είδος αναγνώρισης σύνδεσης χρήστη, όπως την ύπαρξη μιας ενεργής συνεδρίας.

Αν ο χρήστης είναι συνδεδεμένος, η συνάρτηση προχωρά στα επόμενα βήματα:

Ανάκτηση του ονόματος χρήστη από τη συνεδρία, πιθανώς με τη χρήση της συνάρτησης session.get('username').

Αφαίρεση του ονόματος χρήστη από τη συνεδρία με τη χρήση της μεθόδου session.pop('username', None). Αυτό διαγράφει το στοιχείο 'username' από τη συνεδρία, αν υπάρχει.

Χρησιμοποιεί τη συνάρτηση flash() για να προβάλει ένα μήνυμα επιτυχίας που επιστρέφει τον χρήστη που αποσυνδέθηκε.

Αν ο χρήστης δεν είναι συνδεδεμένος, το μήνυμα ενημέρωσης που προβάλλεται είναι ότι "Δεν είστε συνδεδεμένος".

Τελικά, η συνάρτηση κάνει redirect στη σελίδα σύνδεσης, πιθανόν στη διεύθυνση /login, με τη χρήση της συνάρτησης redirect(url\_for('login')).

1. **Κύρια Σελίδα**
   * Λειτουργία: Εμφανίζει την αρχική σελίδα της εφαρμογής.
   * Endpoint: **/**
   * Μέθοδος Κλήσης: GET
   * Παράμετροι: Δεν απαιτούνται παράμετροι.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A black screen with white text

Description automatically generated

Η δήλωση @app.route('/') καθορίζει τη διεύθυνση URL / για αυτήν τη λειτουργία της εφαρμογής. Αυτό είναι το σημείο εισόδου (index) της εφαρμογής.

Η συνάρτηση main() είναι η υλοποίηση για τη διαχείριση των αιτημάτων στη διεύθυνση /.

Η συνάρτηση επιστρέφει το αποτέλεσμα της συνάρτησης render\_template(), η οποία αποτελεί τη βασική μέθοδο για την αποστολή προτύπων HTML προς τον πελάτη. Στη συγκεκριμένη περίπτωση, η συνάρτηση render\_template() χρησιμοποιείται για να αποσταλεί το πρότυπο new\_index.html.

1. **Προβολή Υπηρεσιών του Γυμναστηρίου**
   * Λειτουργία: Ο χρήστης μπορεί να δει μια λίστα με τις διαθέσιμες υπηρεσίες του γυμναστηρίου χωρίς να είναι απαραίτητο να έχει συνδεθεί ή να έχει λογαριασμό στο σύστημα.
   * Endpoint: **/gym\_services**
   * Μέθοδος Κλήσης: GET
   * Παράμετροι: Δεν απαιτούνται παράμετροι.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A screenshot of a computer program

Description automatically generated

Η δήλωση @app.route('/gym\_services') καθορίζει τη διεύθυνση URL /gym\_services για αυτήν τη λειτουργία της εφαρμογής.

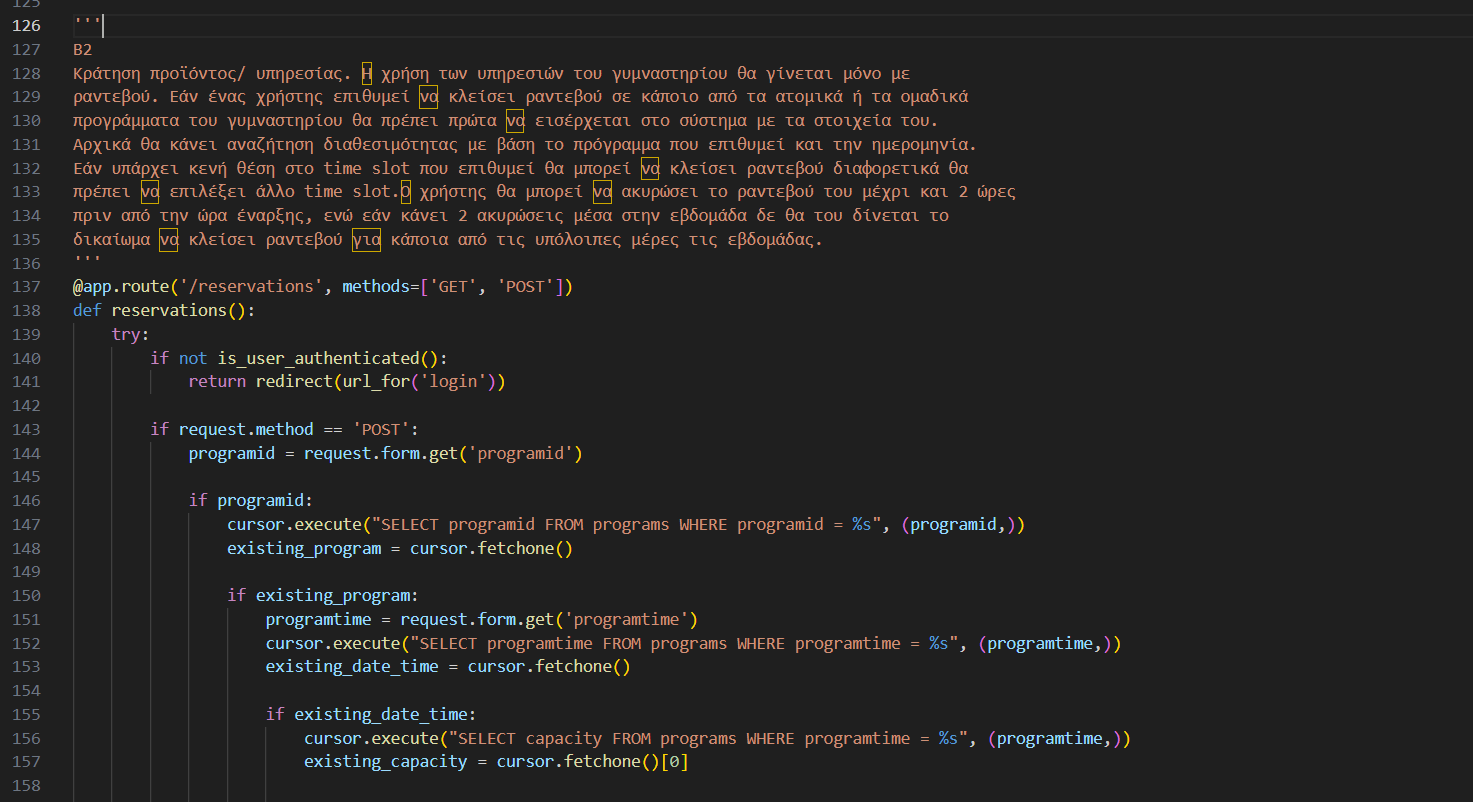
Η συνάρτηση view\_gym\_services() είναι η υλοποίηση για τη διαχείριση των αιτημάτων στη διεύθυνση /gym\_services.

Το ερώτημα SQL "SELECT \* FROM gymservices" ανακτά όλες τις εγγραφές από τον πίνακα gymservices, ο οποίος πιθανότατα περιέχει πληροφορίες για τις υπηρεσίες που προσφέρει το γυμναστήριο.

Η συνάρτηση execute\_query(query) πιθανότατα εκτελεί το ερώτημα SQL στη βάση δεδομένων και επιστρέφει τα αποτελέσματα.

Η συνάρτηση render\_template('new\_gym\_services.html', gym\_services=gym\_services) χρησιμοποιεί τη συνάρτηση render\_template() για να αποστείλει το πρότυπο HTML με όνομα new\_gym\_services.html, μαζί με τα δεδομένα που περιέχονται στη μεταβλητή gym\_services. Αυτά τα δεδομένα θα χρησιμοποιηθούν στο πρότυπο για την εμφάνιση των υπηρεσιών που προσφέρει το γυμναστήριο.

1. **Κράτηση Προϊόντος/Υπηρεσίας**
   * Λειτουργία: Ο χρήστης μπορεί να κλείσει ραντεβού για μια υπηρεσία ή ένα προϊόν του γυμναστηρίου.
   * Endpoint: **/reservations**
   * Μέθοδος Κλήσης: POST
   * Παράμετροι: Το **programid** της επιλεγμένης υπηρεσίας που θέλει να κλείσει ραντεβού ο χρήστης.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.



A screen shot of a computer code

Description automatically generated

Η δήλωση @app.route('/reservations', methods=['GET', 'POST']) καθορίζει τη διεύθυνση URL /reservations για αυτήν τη λειτουργία της εφαρμογής, και ορίζει ότι αυτή η λειτουργία δέχεται αιτήματα τύπου GET και POST.

Η συνάρτηση reservations() ξεκινάει με έναν έλεγχο εάν ο χρήστης έχει συνδεθεί. Εάν δεν έχει συνδεθεί, ανακατευθύνει τον χρήστη στη σελίδα σύνδεσης. Στη συνέχεια, εάν το αίτημα είναι τύπου POST, η συνάρτηση αναλαμβάνει την επεξεργασία του. Ανακτά το programid από τα δεδομένα της φόρμας.

Αφού ανακτήσει το programid, ελέγχει αν υπάρχει το πρόγραμμα με αυτό το programid στη βάση δεδομένων.

Αν υπάρχει το πρόγραμμα, συνεχίζει να ανακτά την ημερομηνία και την ώρα (programtime) από τη φόρμα και ελέγχει τη διαθεσιμότητά του. Αν υπάρχει διαθέσιμη θέση, μειώνει την χωρητικότητα του προγράμματος στη βάση δεδομένων.

Στη συνέχεια, ανακτά τον αναγνωριστικό χρήστη του συνδεδεμένου χρήστη από τη βάση δεδομένων με βάση το username του. Έπειτα, καταχωρεί την κράτηση στη βάση δεδομένων στον πίνακα reservations, προσθέτοντας τον αναγνωριστικό χρήστη, το programid και την κατάσταση "Pending".

Τέλος, επιστρέφει το πρότυπο HTML για τη σελίδα αναζήτησης κρατήσεων.

1. **Ακύρωση Ραντεβού**
   * Λειτουργία: Ο χρήστης μπορεί να ακυρώσει ένα προηγουμένως κλεισμένο ραντεβού.
   * Endpoint: **/reservations\_cancel**
   * Μέθοδος Κλήσης: POST
   * Παράμετροι: Το **programid** της υπηρεσίας που επιθυμεί να ακυρώσει ο χρήστης.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

Αρχικά, έχει δημιουργηθεί το λεξικό user\_cancelations για την αποθήκευση του αριθμού των ακυρώσεων που έχει κάνει ο κάθε χρήστης. Αυτό το λεξικό χρησιμοποιείται για τον έλεγχο του περιορισμού ακυρώσεων σε μία εβδομάδα.

Η λειτουργία reservations\_cancel() έχει καθοριστεί να αποδέχεται αιτήματα τύπου GET και POST στη διεύθυνση URL /reservations\_cancel.

Η συνάρτηση ξεκινάει με έναν έλεγχο εάν ο χρήστης είναι συνδεδεμένος. Αν δεν είναι, το αίτημα ανακατευθύνεται στη σελίδα σύνδεσης.

Στη συνέχεια, εάν το αίτημα είναι τύπου POST, η συνάρτηση αναλαμβάνει την επεξεργασία του. Ανακτά το programid από τη φόρμα του αιτήματος.

Έπειτα, ελέγχει αν υπάρχει το πρόγραμμα με το συγκεκριμένο programid στη βάση δεδομένων.

Αν υπάρχει το πρόγραμμα, συνεχίζει να ανακτά την ημερομηνία και την ώρα (programtime) από τη φόρμα και ελέγχει τη διαθεσιμότητά του. Αν η ακύρωση είναι εφικτή, ελέγχει εάν ο χρήστης έχει υπερβεί το μέγιστο επιτρεπόμενο αριθμό ακυρώσεων για την εβδομάδα. Αν δεν έχει υπερβεί αυτό το όριο, διαγράφει την κράτηση από τη βάση δεδομένων. Σε διαφορετική περίπτωση, επιστρέφει ένα μήνυμα λάθους.

Τέλος, η λειτουργία επιστρέφει το πρότυπο HTML για τη σελίδα αναζήτησης κρατήσεων.

1. **Αναζήτηση Ραντεβού**
   * Λειτουργία: Ο χρήστης μπορεί να αναζητήσει τα διαθέσιμα ραντεβού για υπηρεσίες του γυμναστηρίου.
   * Endpoint: **/reservation\_search**
   * Μέθοδος Κλήσης: GET
   * Παράμετροι: Δεν απαιτούνται παράμετροι.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A black screen with white text

Description automatically generated

Η δήλωση @app.route('/reservation\_search') ορίζει τη διεύθυνση URL /reservation\_search για αυτήν τη λειτουργία της εφαρμογής.

Η συνάρτηση reservation\_search() διαχειρίζεται τα αιτήματα που αφορούν στην εμφάνιση της σελίδας αναζήτησης κρατήσεων ραντεβού. Όταν ο χρήστης προσπαθεί να ανοίξει αυτήν τη διεύθυνση στον περιηγητή του, η συνάρτηση επιστρέφει το πρότυπο HTML της σελίδας reservation\_search.html χρησιμοποιώντας τη συνάρτηση render\_template().

1. **Προβολή Νέων και Ανακοινώσεων**
   * Λειτουργία: Ο χρήστης μπορεί να δει τα νέα και τις ανακοινώσεις που έχουν αναρτηθεί από το γυμναστήριο.
   * Endpoint: **/announcements**
   * Μέθοδος Κλήσης: GET
   * Παράμετροι: Δεν απαιτούνται παράμετροι.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A screenshot of a computer

Description automatically generated

Η δήλωση @app.route('/announcements') ορίζει τη διεύθυνση URL /announcements για αυτήν τη λειτουργία της εφαρμογής.

Η συνάρτηση view\_announcements() εκτελείται όταν ο χρήστης προσπαθεί να ανοίξει την παραπάνω διεύθυνση URL. Αυτή η συνάρτηση ανακτά όλες τις ανακοινώσεις από τον πίνακα announcements της βάσης δεδομένων.

Η μεταβλητή query περιέχει το SQL ερώτημα που ανακτά όλες τις ανακοινώσεις από τον πίνακα announcements.

Η συνάρτηση execute\_query() εκτελεί το SQL ερώτημα και επιστρέφει τα αποτελέσματα.

Τα αποτελέσματα του ερωτήματος αποθηκεύονται στη μεταβλητή announcements.

Στη συνέχεια, η συνάρτηση χρησιμοποιεί τη συνάρτηση render\_template() για να επιστρέψει το πρότυπο HTML της σελίδας new\_announcements.html, περνώντας τις ανακοινώσεις ως παράμετρο στο πρότυπο.

1. **Ιστορικό Κρατήσεων**
   * Λειτουργία: Ο χρήστης μπορεί να δει μια λίστα με όλες τις κρατήσεις που έχει κάνει στο σύστημα μέχρι τη στιγμή εκείνη.
   * Endpoint: **/reservation\_history**
   * Μέθοδος Κλήσης: GET
   * Παράμετροι: Δεν απαιτούνται παράμετροι.
   * Header: Δεν απαιτείται ειδικό header.
   * Σώμα Αίτησης: Δεν χρησιμοποιείται.

A black screen with colorful text

Description automatically generated

Η δήλωση @app.route('/reservation\_history') ορίζει τη διεύθυνση URL /reservation\_history για αυτήν τη λειτουργία της εφαρμογής.

Η συνάρτηση reservation\_history() εκτελείται όταν ο χρήστης προσπαθεί να ανοίξει την παραπάνω διεύθυνση URL. Αρχικά, ελέγχει εάν ο χρήστης είναι συνδεδεμένος. Αν δεν είναι, το αίτημα ανακατευθύνεται στη σελίδα σύνδεσης.

Στη συνέχεια, η συνάρτηση ανακτά όλες τις κρατήσεις από τον πίνακα reservations της βάσης δεδομένων.

Η μεταβλητή query περιέχει το SQL ερώτημα που ανακτά όλες τις κρατήσεις από τον πίνακα reservations.

Η συνάρτηση execute\_query() εκτελεί το SQL ερώτημα και επιστρέφει τα αποτελέσματα.

Τα αποτελέσματα του ερωτήματος αποθηκεύονται στη μεταβλητή reservations.

Τέλος, η συνάρτηση χρησιμοποιεί τη συνάρτηση render\_template() για να επιστρέψει το πρότυπο HTML της σελίδας reservation\_history.html, περνώντας τις κρατήσεις ως παράμετρο στο πρότυπο.

Αυτές οι διαδρομές εξυπηρετούν στην πρόσβαση σε σελίδες διαχείρισης που προορίζονται για διαχειριστές του συστήματος. Εκτός από τις βασικές λειτουργίες που έχουν ήδη περιγραφεί, αυτές οι διαδρομές επιτρέπουν τη διαχείριση διάφορων στοιχείων στο σύστημα. Παρακάτω είναι η λειτουργικότητα που προσφέρει κάθε μια από αυτές τις διαδρομές:

1. **/admin\_general**
   * Λειτουργία: Είναι η κεντρική σελίδα διαχείρισης του διαχειριστή. Εμφανίζει γενικές πληροφορίες και επιτρέπει πρόσβαση σε άλλες λειτουργίες διαχείρισης.
   * Προϋποθέσεις: Απαιτείται η σύνδεση ενός διαχειριστή.
   * Πρόσβαση: GET
2. **Σελίδες για δημιουργία κουμπιών (A1, A2, A3, A4)**
   * Λειτουργία: Προσφέρει σελίδες με φόρμες για τη δημιουργία νέων στοιχείων στο σύστημα.
   * Προϋποθέσεις: Απαιτείται η σύνδεση ενός διαχειριστή.
   * Πρόσβαση: GET
3. **Σελίδες για διαχείριση υπαρχόντων κουμπιών (A1, A2, A3, A4)**
   * Λειτουργία: Παρέχει σελίδες για τη διαχείριση των υπαρχόντων στοιχείων στο σύστημα, όπως ενημέρωση ή διαγραφή.
   * Προϋποθέσεις: Απαιτείται η σύνδεση ενός διαχειριστή.
   * Πρόσβαση: GET
4. **Διαχείριση αιτημάτων εγγραφής στο σύστημα (A1)**
   * Endpoint για προβολή των αιτημάτων: **/status\_view**
     + Λειτουργία: Ο διαχειριστής μπορεί να δει τα αιτήματα εγγραφής στο σύστημα και να τα εγκρίνει ή να τα απορρίψει.
     + Μέθοδος Κλήσης: GET
   * Endpoint για διαχείριση των αιτημάτων: **/status\_manage**
     + Λειτουργία: Ο διαχειριστής μπορεί να ενημερώσει την κατάσταση ενός αιτήματος εγγραφής.
     + Μέθοδος Κλήσης: POST

A computer screen shot of text

Description automatically generated

Η συνάρτηση status\_view() δημιουργεί μια διαδρομή στη διεύθυνση '/status\_view' που διαχειρίζεται αιτήσεις HTTP με τη μέθοδο GET. Αυτή η συνάρτηση επιστρέφει μια HTML σελίδα που εμφανίζει τους χρήστες από μια βάση δεδομένων, πιθανώς με κάποια στοιχεία τους, που προέρχονται από το αποτέλεσμα της SQL ερώτησης που εκτελείται στη συνάρτηση execute\_query.

Η συνάρτηση status\_manage() δημιουργεί μια διαδρομή στη διεύθυνση '/status\_manage' που διαχειρίζεται αιτήσεις HTTP με τις μεθόδους GET και POST. Όταν λαμβάνεται μια αίτηση POST, αναλαμβάνει να ενημερώσει την κατάσταση ενός χρήστη στη βάση δεδομένων με βάση τα δεδομένα που λαμβάνονται από τη φόρμα που υποβάλλεται. Συγκεκριμένα, παίρνει το userid και το status από την αίτηση POST, ελέγχει αν ο χρήστης υπάρχει στη βάση δεδομένων, και ενημερώνει την κατάστασή του αν αυτός είναι ορισμένος. Έπειτα επιστρέφει στην αρχική σελίδα με τη χρήση της συνάρτησης redirect.

Σε περίπτωση που παρουσιαστεί κάποιο σφάλμα στην εκτέλεση των ενημερώσεων στη βάση δεδομένων, η εφαρμογή επιστρέφει ένα μήνυμα σφάλματος και κωδικό HTTP 500.

1. **Διαχείριση χρηστών συστήματος (A2)**
   * Endpoint για προβολή των χρηστών: **/users\_view**
     + Λειτουργία: Ο διαχειριστής μπορεί να δει τους χρήστες του συστήματος και τα στοιχεία τους.
     + Μέθοδος Κλήσης: GET
   * Endpoint για δημιουργία νέου χρήστη: **/users\_create**
     + Λειτουργία: Ο διαχειριστής μπορεί να δημιουργήσει έναν νέο χρήστη στο σύστημα.
     + Μέθοδος Κλήσης: POST
   * Endpoint για ενημέρωση πληροφοριών χρήστη: **/users\_update**
     + Λειτουργία: Ο διαχειριστής μπορεί να ενημερώσει τα στοιχεία ενός χρήστη ή τον ρόλο του.
     + Μέθοδος Κλήσης: POST
   * Endpoint για διαγραφή χρήστη: **/users\_delete**
     + Λειτουργία: Ο διαχειριστής μπορεί να διαγράψει έναν χρήστη από το σύστημα.
     + Μέθοδος Κλήσης: POST

A computer code on a black background

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

1. Δημιουργία Νέου Χρήστη (users\_create):

• Αυτή η λειτουργία εκτελείται όταν λαμβάνεται μια αίτηση POST στη διεύθυνση '/users\_create'.

• Παίρνει τα δεδομένα από τη φόρμα που υποβάλλεται (όπως όνομα, επώνυμο, χώρα, πόλη, διεύθυνση email, κλπ.).

• Εισάγει τα δεδομένα αυτά στον πίνακα χρηστών (users) της βάσης δεδομένων.

2. Ενημέρωση Χρήστη (users\_update):

• Αυτή η λειτουργία εκτελείται όταν λαμβάνεται μια αίτηση POST στη διεύθυνση '/users\_update'.

• Παίρνει το userid από τη φόρμα και ελέγχει αν υπάρχει ήδη στη βάση δεδομένων.

• Αν υπάρχει, ενημερώνει τα στοιχεία του χρήστη (όπως όνομα, επώνυμο, κλπ.) στη βάση δεδομένων.

3. Διαγραφή Χρήστη (users\_delete):

• Αυτή η λειτουργία εκτελείται όταν λαμβάνεται μια αίτηση POST στη διεύθυνση '/users\_delete'.

• Παίρνει το userid από τη φόρμα και ελέγχει αν υπάρχει ήδη στη βάση δεδομένων.

• Αν υπάρχει, διαγράφει τον χρήστη από τον πίνακα χρηστών στη βάση δεδομένων.

4. Προβολή Χρηστών (users\_view):

• Αυτή η λειτουργία εκτελείται όταν λαμβάνεται μια αίτηση GET στη διεύθυνση '/users\_view'.

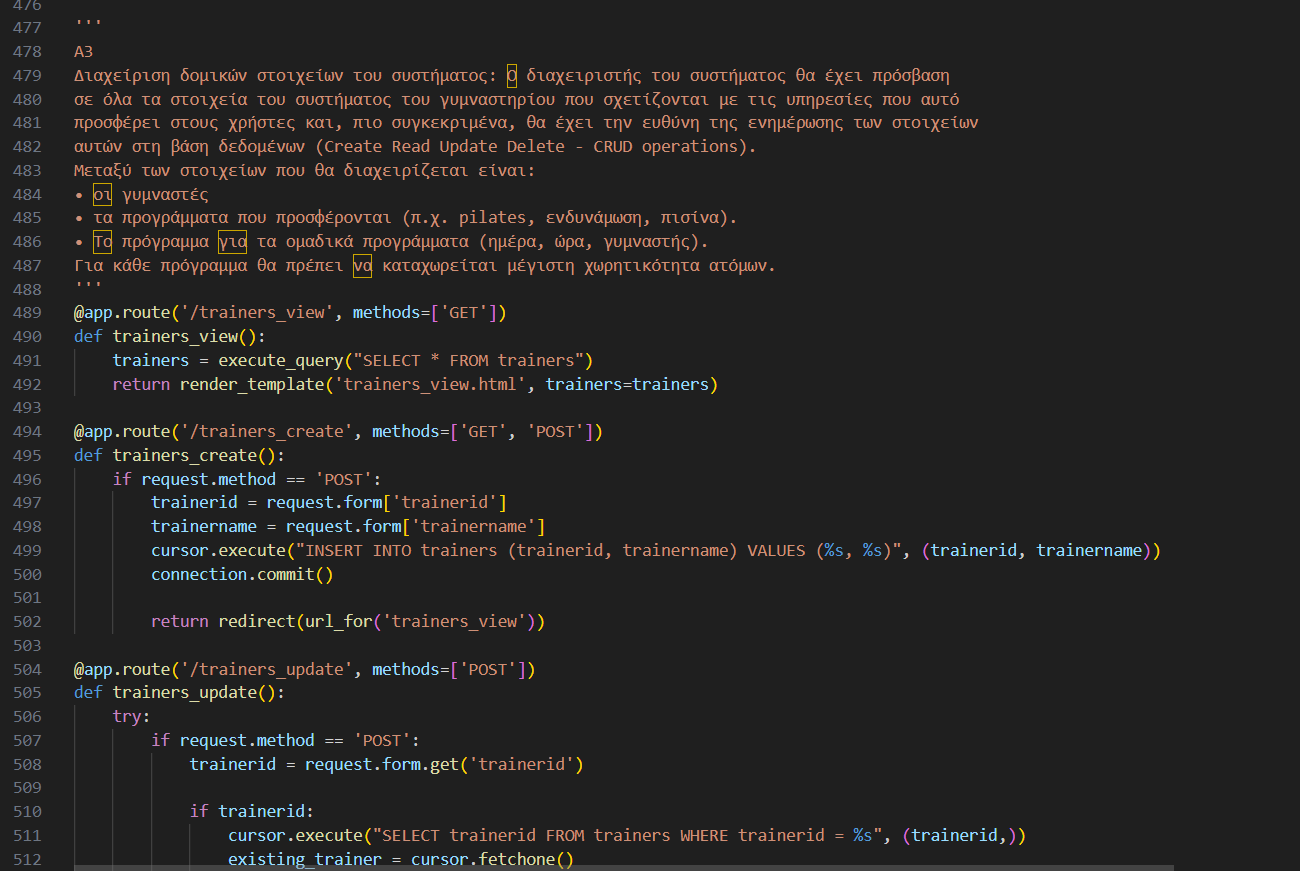
• Παίρνει όλους τους χρήστες από τη βάση δεδομένων και τους εμφανίζει στην ιστοσελίδα.

Επιπλέον, ο κώδικας επιχειρεί να προσθέσει τους χρήστες που έχουν ρόλο "Trainer" στον πίνακα trainers. Αυτή η ενέργεια γίνεται μετά την ενημέρωση του ρόλου τους. Ωστόσο, υπάρχει ένα πρόβλημα εδώ: το trainername που εισάγεται στον πίνακα trainers φαίνεται να παράγεται από το firstname και το lastname του χρήστη, αλλά υπάρχει ένα κενό μεταξύ τους, ενώ πιθανώς θα έπρεπε να χρησιμοποιείται κάποιος χαρακτήρας χωρίς κενό, όπως το "\_". Επίσης, αυτό το κομμάτι κώδικα πρέπει να βρίσκεται σε μια διαδικασία όπως η δημιουργία νέου χρήστη και όχι μέσα στη διαδικασία ενημέρωσης του ρόλου ενός χρήστη.

Τέλος, υπάρχει και κάποια αντιφάσεις στον κώδικα: για παράδειγμα, η συνάρτηση users\_delete() αποτελείται από δύο επίπεδα if, όπου το πρώτο επίπεδο ελέγχει το request method και το δεύτερο επίπεδο ελέγχει την ύπαρξη του χρήστη πριν από τη διαγραφή του. Αυτό δεν είναι απαραίτητο, καθώς η διαγραφή ενός χρήστη μπορεί να πραγματοποιηθεί μόνο εάν υπάρχει.

**Διαχείριση δομικών στοιχείων του συστήματος (A3)**

* **Προβολή των γυμναστών**
  + Endpoint: **/trainers\_view**
  + Λειτουργία: Ο διαχειριστής μπορεί να δει τη λίστα των γυμναστών που υπάρχουν στο σύστημα.
  + Μέθοδος Κλήσης: GET
* **Δημιουργία νέου γυμναστή**
  + Endpoint: **/trainers\_create**
  + Λειτουργία: Ο διαχειριστής μπορεί να δημιουργήσει ένα νέο γυμναστή στο σύστημα.
  + Μέθοδος Κλήσης: POST
* **Ενημέρωση στοιχείων γυμναστή**
  + Endpoint: **/trainers\_update**
  + Λειτουργία: Ο διαχειριστής μπορεί να ενημερώσει τα στοιχεία ενός γυμναστή.
  + Μέθοδος Κλήσης: POST
* **Διαγραφή γυμναστή**
  + Endpoint: **/trainers\_delete**
  + Λειτουργία: Ο διαχειριστής μπορεί να διαγράψει έναν γυμναστή από το σύστημα.
  + Μέθοδος Κλήσης: POST
* **Προβολή των προγραμμάτων**
  + Endpoint: **/programs\_view**
  + Λειτουργία: Ο διαχειριστής μπορεί να δει τη λίστα των προγραμμάτων που προσφέρονται από το γυμναστήριο.
  + Μέθοδος Κλήσης: GET
* **Δημιουργία νέου προγράμματος**
  + Endpoint: **/program\_create**
  + Λειτουργία: Ο διαχειριστής μπορεί να δημιουργήσει ένα νέο πρόγραμμα που προσφέρεται από το γυμναστήριο.
  + Μέθοδος Κλήσης: POST



A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

1. Διαχείριση Γυμναστών (trainers\_view, trainers\_create, trainers\_update, trainers\_delete):

• Η σειρά αυτή αποτελείται από τις λειτουργίες που επιτρέπουν τη διαχείριση των γυμναστών στο γυμναστήριο.

• Η trainers\_view επιτρέπει την προβολή όλων των γυμναστών που υπάρχουν στη βάση δεδομένων.

• Η trainers\_create επιτρέπει τη δημιουργία νέου γυμναστή.

• Η trainers\_update επιτρέπει την ενημέρωση των στοιχείων ενός γυμναστή.

• Η trainers\_delete επιτρέπει τη διαγραφή ενός γυμναστή από τη βάση δεδομένων.

2. Διαχείριση Προγραμμάτων (programs\_view, program\_create, program\_update, program\_delete):

• Η σειρά αυτή αποτελείται από τις λειτουργίες που επιτρέπουν τη διαχείριση των προγραμμάτων που προσφέρονται στο γυμναστήριο.

• Η programs\_view επιτρέπει την προβολή όλων των προγραμμάτων που υπάρχουν στη βάση δεδομένων.

• Η program\_create επιτρέπει τη δημιουργία νέου προγράμματος.

• Η program\_update επιτρέπει την ενημέρωση των στοιχείων ενός προγράμματος.

• Η program\_delete επιτρέπει τη διαγραφή ενός προγράμματος από τη βάση δεδομένων.

* **Ενημέρωση πληροφοριών προγράμματος Α4**
  + Endpoint: **/program\_update**
  + Λειτουργία: Ο διαχειριστής μπορεί να ενημερώσει τα στοιχεία ενός προγράμματος.
  + Μέθοδος Κλήσης: POST
* **Διαγραφή προγράμματος**
  + Endpoint: **/program\_delete**
  + Λειτουργία: Ο διαχειριστής μπορεί να διαγράψει ένα πρόγραμμα από το σύστημα.
  + Μέθοδος Κλήσης: POST

1. **Προβολή ανακοινώσεων/προσφορών**:
   * Endpoint: **/announcements\_view**
   * Μέθοδος: GET
   * Παράμετροι: Κανένα
   * Header: Κανένα
   * Σώμα: Λίστα με τις ανακοινώσεις/προσφορές
2. **Δημιουργία ανακοίνωσης/προσφοράς**:
   * Endpoint: **/announcement\_create**
   * Μέθοδος: POST
   * Παράμετροι: Αριθμός αναγνώρισης, Κείμενο ανακοίνωσης, Ημερομηνία δημιουργίας, Αναγνωριστικό χρήστη
   * Header: Κανένα
   * Σώμα: Κανένα
3. **Διαγραφή ανακοίνωσης/προσφοράς**:
   * Endpoint: **/announcement\_delete**
   * Μέθοδος: POST
   * Παράμετροι: Αριθμός αναγνώρισης
   * Header: Κανένα
   * Σώμα: Κανένα
4. **Ενημέρωση ανακοίνωσης/προσφοράς**:
   * Endpoint: **/announcement\_update**
   * Μέθοδος: POST
   * Παράμετροι: Αριθμός αναγνώρισης, Κείμενο ανακοίνωσης, Ημερομηνία δημιουργίας, Αναγνωριστικό χρήστη
   * Header: Κανένα
   * Σώμα: Κανένα

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

1. Προβολή Ανακοινώσεων (announcements\_view):

• Η λειτουργία αυτή επιτρέπει στον διαχειριστή να δει όλες τις υπάρχουσες ανακοινώσεις και προσφορές που έχουν αναρτηθεί στην βάση δεδομένων.

2. Δημιουργία Ανακοίνωσης (announcement\_create):

• Η λειτουργία αυτή επιτρέπει στον διαχειριστή να δημιουργήσει μια νέα ανακοίνωση ή προσφορά.

• Ο διαχειριστής συμπληρώνει τα απαραίτητα πεδία (ID, κείμενο ανακοίνωσης, ημερομηνία δημιουργίας, και τον ID του χρήστη που την αναρτά) στη φόρμα και μετά αυτά αποθηκεύονται στη βάση δεδομένων.

3. Διαγραφή Ανακοίνωσης (announcement\_delete):

• Η λειτουργία αυτή επιτρέπει στον διαχειριστή να διαγράψει μια υπάρχουσα ανακοίνωση ή προσφορά.

• Ο διαχειριστής επιλέγει την ανακοίνωση που επιθυμεί να διαγράψει από μια φόρμα και μετά αυτή διαγράφεται από τη βάση δεδομένων.

4. Ενημέρωση Ανακοίνωσης (announcement\_update):

• Η λειτουργία αυτή επιτρέπει στον διαχειριστή να ενημερώσει την περιεχόμενη μιας υπάρχουσας ανακοίνωσης ή προσφοράς.

• Ο διαχειριστής επιλέγει την ανακοίνωση που θέλει να ενημερώσει και συμπληρώνει τα νέα δεδομένα (κείμενο ανακοίνωσης, ημερομηνία δημιουργίας, και ID του χρήστη) στη φόρμα. Έπειτα, αυτά τα δεδομένα ενημερώνονται στη βάση δεδομένων.

# Εγχειρίδιο διαχειριστή

# Εγχειρίδιο χρήστη