ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
TECHNICAL UNIVERSITY OF CRETE

Υπηρεσίες στο υπολογιστικό νέφος και την ομίχλη

Αναφορά Άσκηση 1

Σκευάκης Βασίλειος

Προπτυχιακός Φοιτητής

ΑΜ: 2012030033

vskevakis@isc.tuc.gr

Τμημα Ηλεκτρολογων Μηχανικων
& Μηχανικων Υπολογιστων

8 Νοεμβρίου 2020

Περιγραφή Εργασίας

Η εφαρμοφή αποτελεί ένα σύστημα καταχώρησης και ενημέρωσης χρηστών για ταινίες στον κινηματογράφο. Ένας χρήστης μπορεί να εγγραφή στην εφαρμογή και να ζητήσει τον ρόλο που επιθυμεί (User, Cinema Owner, Admin). Στην συνέχεια, ο Admin του συστήματος είναι υπεύθυνος να δεχθεί ή να απορρίψει τον χρήστη από το Admin Panel. Οι Cinema Owners έχουν το δικό τους panel (My Cinema) στο οποίο μπορούν να προσθέσουν ή να επεξεργαστούν ταινίες τους. Οι Users μπορούν να δούν και να αναζητήσουν ταινίες που έχουν προσθέσει οι Cinema Owners.

Περιεχόμενα

Βάσεις Δεδομένων

1. PostgreSQL για τα δεδομένα των χρηστών και των ταινιών

Για την αποθήκευση των χρηστών και των ταινιών του συστήματος χρησιμοποιήθηκαν δύο βάσεις δεδομένων PostgreSQL (auth_db, back_db).

Η auth_db περιέχει ένα table (user) το οποίο έχει τα εξής κελιά:

user_id	name	surname	username	password	email	role	is_confirmed
Integer	String	String	String	String	String	Enum	Boolean

Ενώ η back_db περιέχει τα παρακάτω:

movie_id	title	start_date	end_date	cinema_name	category	poster_path
Integer	String	Date	Date	String	String	String

Έχουμε προσθέσει το poster_path το οποίο δεν μας ζητήθηκε αλλά το χρησιμοποιούμε για να αποθηκεύουμε τις αφίσες των ταινιών ώστε να κάνουμε ομορφότερη την εφαρμογή μας. Ο τρόπος με τον οποίο λειτουργεί θα αναλυθεί παρακάτω.

Δεν έχουμε δημιουργήσει τον πίνακα Cinemas καθώς δεν εξυπηρετούσε στην λειτουργικότητα της εφαρμογής μας. Ενδέχεται όμως να τον προσθέσουμε στην δεύτερη άσκηση ώστε να δίνεται η δυνατότητα στον Cinema Owner να έχει διαφορετικό Cinema Name απ' ότι το username του.

2. Redis για την αποθήκευση και επεξαργασία των αγαπημένων ταινιών του χρήση

Για την αποθήκευση των αγαπημένων ταινιών του χρήστη, δημιουργήσαμε μια Redis DB (fav-db) καθώς βοηθάει στην ταχύτητα και στην λειτουργικότητα της εφαρμογής. Στην Redis τα δεδομένα αποθηκεύονται ως "sets", δηλαδή η δομής της ενδεικτικά είναι η εξής:

user_id	favs
Key	Array (set)

Όπου για κάθε user_id αποθηκεύονται σε ένα Array of Integers τα movie_id των ταινιών που έχει προσθέσει στα αγαπημένα.

API

Για την διασύνδεση των λειτουργιών της εφαρμογής με την εμφάνιση στον χρήστη της εφαρμογής δημιουργήθηκαν κάποια API's ώστε να τα καλούμε από το γραφικό περιβάλλον της εφαρμογής και να παίρνουμε τα δεδομένα που χρειάζεται να εμφανίσουμε ή να αποθηκεύσουμε. Το API μας είναι γραμμένο σε Python με το Flask Framework.

1. Authentication API Για την αυθεντικοποίηση του χρήστη δημιουργήσαμε το Authentication API μας, μέσω του οποίου ένας χρήστης μπορεί να δημιουργηθεί, να διαγραφθεί και να γίνει αποδεκτός από τον admin. Χρησιμοποιεί την βάση δεδομένων auth_db που αναλύσαμε προηγουμένως και τα API Calls είναι τα εξής.

- `/auth/register [POST]` - Δίνοντας σε JSON μορφή τα δεδομένα του χρήστη, μετά απο τους αναγκαστικούς ελέγχους, και αφού ο κωδικός του χρήστη έχει κωδικοποιηθεί με τη χρήση hash, αποθηκέυει τα δεδομένα του καινούργιου χρήστη στην βάση δεδομένων και επιστρέφει ένα token.
- `/auth/login [POST]` - Ελέγχει τα στοιχεία του χρήστη, και το hash του κωδικού που υπάρχει στην βάση, με το hash του κωδικού που δίνει ο χρήστης και επιστρέφει ένα token εάν ο χρήστης αυθεντικοοποιήθηκε
- `/auth/accept_user [POST]` - Αλλάζει το κελί is_confirmed σε True για να δεχθούμε έτσι τον χρήστη στο σύστημα μας
- `/auth/check_token [POST]` - Αποκωδικοποιεί το token (προσωρινά γίνεται στο front-end αυτό)
- `/auth/delete_user [POST]` - Διαγράφει έναν χρήστη από τη βάση δεδομένων
- `/auth/get_users [POST]` - Επιστρέφει την λίστα με τους users, ενώ μπορεί να δεχθεί ώς είσοδο ένα search string και να επιστρέψει μόνο τους users που επιθυμούμε με την αναζήτηση.

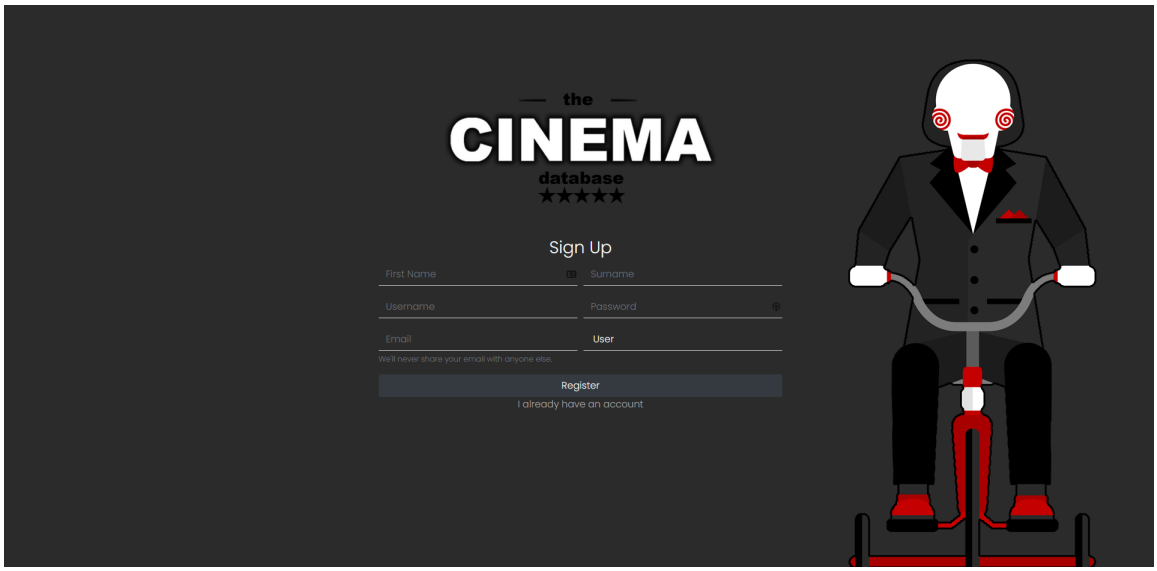
2. Back-end API Για την αποθήκευση, την αναζήτηση, την διαγραφή ταινιών και αγαπημένων των χρηστών, χρησιμοποιούμε το back-end API μας το οποίο έχει τα εξής Calls.

- `/back/add_movie [POST]` - Προσθέτει μια ταινία στην βάση δεδομένων μας. Κατά την προσθήκη μιας ταινίας υπάρχει επικοινωνία με το API του `themoviedb.org` και φόρτωση στην βάση δεδομένων μας, της αφίσας της ταινίας την οποία προσθέτουμε. (Η αναζήτηση γίνεται με βάση τον τίτλο οπότε δεν είναι πάντα επιτυχής)
- `/back/delete_movie [POST]` - Αφαίρεση μιας ταινίας από την βάση δεδομένων μας.
- `/back/edit_mode [POST]` - Επεξεργασία μιας ταινίας με είσοδο το `movie_id` της ταινίας και τα στοιχεία που θέλουμε να αλλάξουμε.
- `/back/search_movies [POST]` - Αναζήτηση ταινιών. Ανάλογα την είσοδο που θα του δώσουμε, θα αναζητήσει με βάση αυτήν. Για κενό input επιστρέφει όλες τις ταινίες.
- `/back/search_cinema_movies [POST]` - Παρόμοια αναζήτηση, αλλά για συγκεκριμένο cinema_name το οποίο το περνάμε σαν είσοδο.
- `/back/modify_fav [POST]` - Προσθήκη ή διαγραφή movie_id από το set με κλειδί το user_id.
- `/back/get_favs [POST]` - Επιστρέφει έναν πίνακα με τα movie_id's των αγαπημένων ταινιών του χρήστη με user_id.

Front-End

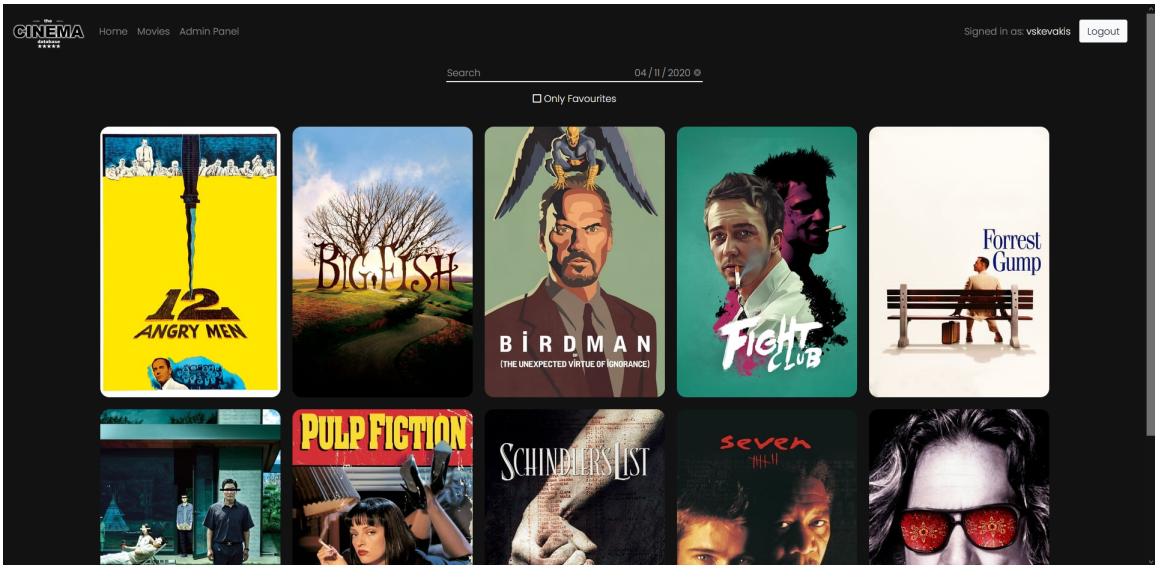
Το front-end κομμάτι της εφαρμογής μας είναι γραμμένο σε React JS και τα API Calls γίνονται μέσω του `Axios`

1. Login/Register Ο χρήστης πρέπει είτε να συνδεθεί είτε να εγγραφή στην εφαρμογή μας για να μπορέσει να την χρησιμοποιήσει. Όταν συνδεθεί, λαμβάνει ένα token το οποίο χρησιμοποιείται για την αυθεντικοποίηση του. Το token αυτό, περιέχει πέραν την ώρα έκδοσης και ώρα λήξης, πληροφορίες για το username του χρήστη, τον ρόλο του αλλά και το εάν είναι επιβεβαιωμένος. Ανάλογα με τα παραπάνω, στον χρήστη εμφανίζεται διαφορετικό μενού πλοήγησης.



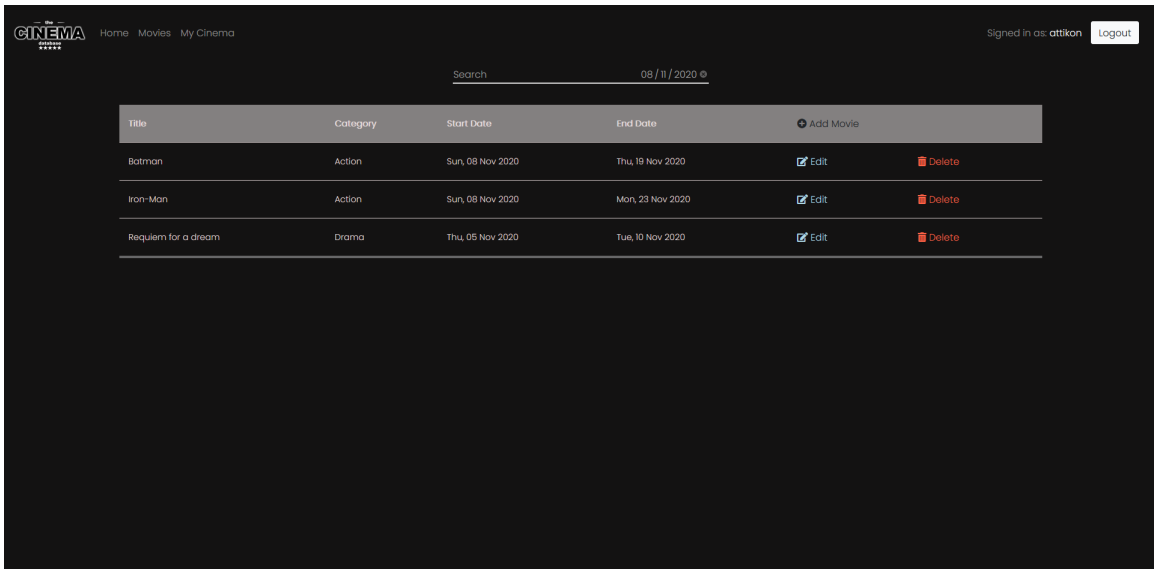
Σχήμα 1: Register

2. Movies Στην σελίδα αυτή εμφανίζονται οι διαθέσιμες ταινίες. Υπάρχει δυνατότητα αναζήτησης μέσω ένος απλού input. Εάν ο χρήστης εισάγει στην αναζήτηση κριτήριο την κατηγορία "Crime", τότε θα του εμφανίσει μόνο ταινίες από αυτή την κατηγορία ή ταινίες με το "Crime" στον τίτλο τους. Επίσης μπορεί να γίνει αναζήτηση με βάση την ημερομηνία, ενώ η προεπιλεγμένη ημερομηνία είναι η εκάστοτε ημέρα, ώστε ο χρήστης να βλέπει πρώτα τις ταινίες οι οποίες παίζονται εκείνη την ημέρα. Επίσης έχει την δυνατότητα να προσθέσει ή να αφαιρέσει μια ταινία από τα αγαπημένα του, αλλά και να εμφανίσει μόνο τις αγαπημένες του ταινίες.



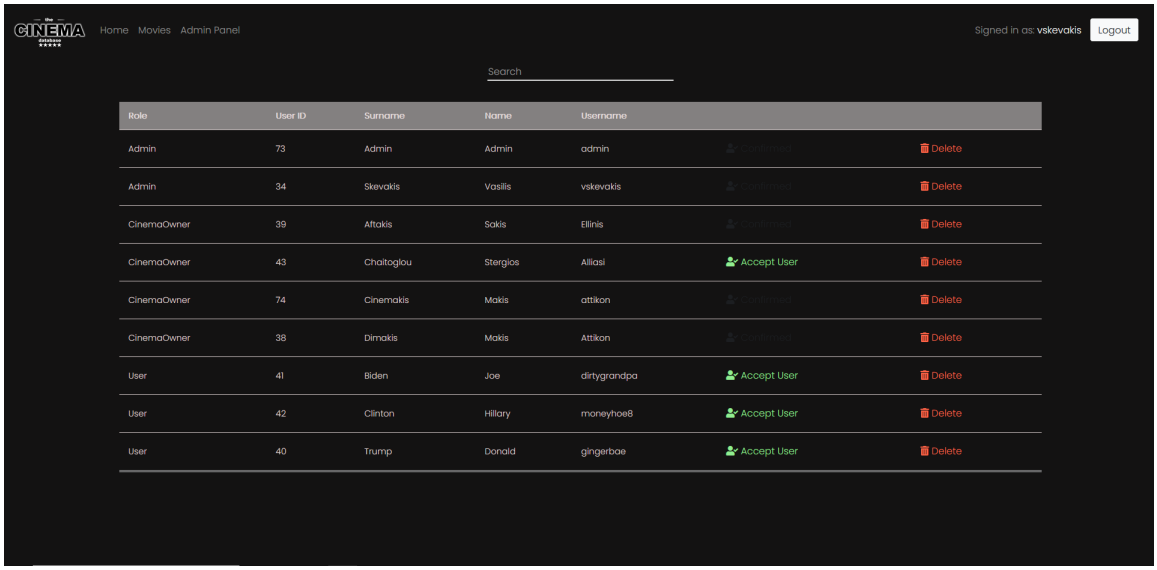
Σχήμα 2: Movies

3. My Cinema Πρόσβαση στο My Cinema έχουν μόνο οι χρήστες οι οποίοι έχουν ρόλο Cinema Owner και εδώ μπορούν να διαχειριστούν τις ταινίες τους, να τους αλλάξουν οποιαδήποτε πληροφορία αλλά και να προσθέσουν ή να διαγράψουν ταινίες. Επίσης έχουν την δυνατότητα να αναζητήσουν ταινίες με αντίστοιχο input αναζήτησης.



Σχήμα 3: My Cinema

4. Admin Panel Πρόσβαση στο admin panel έχουν μόνο οι admins και μπορούν από εδώ να διαχειριστούν τους χρήστες. Μπορούν να δεχθούν ή να διαγράφουν κάποιον χρήστη.



Σχήμα 4: Admin Panel

Πηγές

- Εργασία Κατανεμημένων Συστημάτων 2020
- stack overflow
- SQL Alchemy Docs
- React JS
- Flask Documentation

Git & Live Demo

Git Repo: https://github.com/vskevakis/COMP-518-Cloud-Services
Live Demo: http://35.234.96.65/