Έγγραφο απαιτήσεων λογισμικού(SRS)

ΠΡΟΣΑΡΜΟΓΗ ΤΟΥ ΑΝΤΙΣΤΟΙΧΟΥ ΕΓΓΡΑΦΟΥ ΤΟΥ ΠΡΟΤΥΠΟΥ ISO/IEC/IEEE 29148:2011

**Ntuaflix**

Περιεχόμενα

[1. Εισαγωγή 2](#_Toc153716059)

[1.1 Εισαγωγή: σκοπός του λογισμικού 2](#_Toc153716060)

[1.2 Διεπαφές (interfaces) 2](#_Toc153716061)

[1.2.1 Διεπαφές με εξωτερικά συστήματα 2](#_Toc153716062)

[1.2.2 Διεπαφές με το χρήστη 4](#_Toc153716063)

[2. Αναφορές - πηγές πληροφοριών 4](#_Toc153716064)

[3. Προδιαγραφές απαιτήσεων λογισμικού 4](#_Toc153716065)

[3.1 Περιπτώσεις χρήσης 4](#_Toc153716066)

[3.1.1 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 1: Φιλτράρισμα ταινίας 5](#_Toc153716067)

[3.1.2 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 2: Πάτημα Like/ Dislike 8](#_Toc153716068)

[3.2 Απαιτήσεις επιδόσεων 11](#_Toc153716069)

[3.3 Απαιτήσεις οργάνωσης δεδομένων 11](#_Toc153716070)

[3.3.1 Απαιτήσεις και περιορισμοί πρόσβασης σε δεδομένα 12](#_Toc153716071)

[3.4 Περιορισμοί σχεδίασης 12](#_Toc153716072)

[3.5 Λοιπές απαιτήσεις 12](#_Toc153716073)

[3.5.1 Απαιτήσεις διαθεσιμότητας λογισμικού 12](#_Toc153716074)

[3.5.2 Απαιτήσεις ασφάλειας 12](#_Toc153716075)

[3.5.2 Απαιτήσεις Συντήρησης 12](#_Toc153716076)

# Εισαγωγή

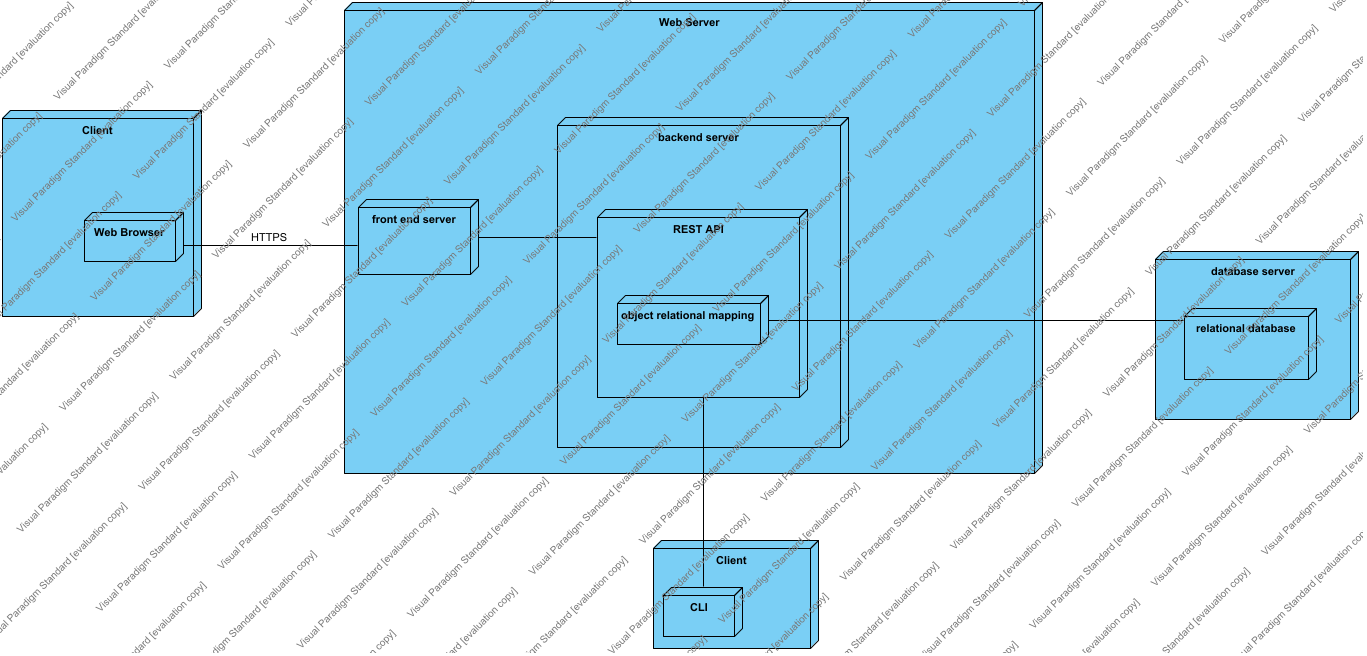
## Εισαγωγή: σκοπός του λογισμικού

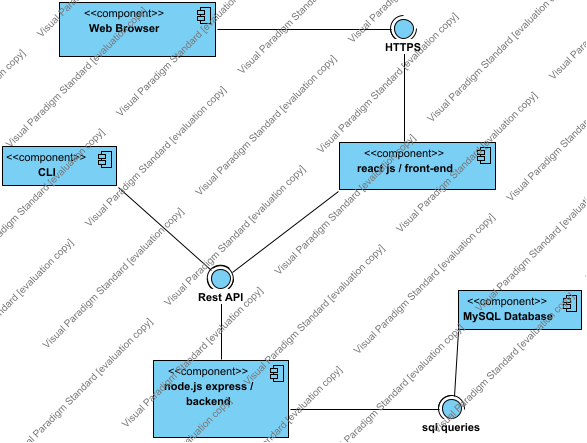
Το σύστημα ntuaflix αφορά την ανάπτυξη μιας εφαρμογής που διευκολύνει την διαχείριση και την ανάλυση μεγάλων συλλογών δεδομένων ταινιών και τηλεοπτικών σειρών.   
  
Η εφαρμογή προσφέρει στους εγγεγραμμένους χρήστες της δυνατότητες εξαγωγής στατιστικών πληροφοριών, αναζήτησης ταινιών και σειρών με βάση διάφορα κριτήρια και φίλτρα και δημιουργίας προσωπικών συλλογών αγαπημένων ταινιών. Μέσα από ένα φιλικό προς το χρήστη περιβάλλον, διευκολύνεται σημαντικά το πρόβλημα της διαχείρισης και ανάλυσης χιλιάδων διάσημων τίτλων.   
  
Η εφαρμογή μας έχει σχεδιαστεί ώστε να είναι απλή στην χρήση και φιλική προς όσους ενδιαφέρονται για την πλοήγηση και ανάλυση πληροφοριών για ταινίες και σειρές.

## 1.2 Διεπαφές (interfaces)

### 1.2.1 Διεπαφές με εξωτερικά συστήματα

Το σύστημα χρησιμοποιεί ως διεπαφή με τη βάση δεδομένων το πρότυπο κλήσεων REST API. Back End και Front End server κατοικούν στον ίδιο web server για τα πλαίσια της εφαρμογής και μεταξύ Back End και Βάσης δεδομένων χρησιμοποιείται ένα επίπεδο Object relational mapping τύπου SQL. Για βάση δεδομένων χρησιμοποιούμε τo σχεσιακό DBMS MySQL το οποίο ακολουθεί το πρωτόκολλο MySQL Protocol μεταξύ MySQL client και MySQL server. Το μορφότυπο ανταλλαγής δεδομένων μέσω του RESTAPI είναι προκαθορισμένα JSON και προαιρετικά CSV. Τέλος η μεταφορά δεδομένων πάνω από το διαδίκτυο αλλά και δεδομένων μεταξύ Back End και Front End γίνεται με HTTPS.

Αυτά οπτικοποιούνται στα παρακάτω διαγράμματα:  
  
UML **Deployment Diagram**:

UML **Component Diagram**:

### 1.2.2 Διεπαφές με το χρήστη

Προδιαγραφή διεπαφών με το χρήστη. Μοντέλο Use Case (UML).

Το λογισμικό παρέχεται στους χρήστες μέσω δύο πιθανών διεπαφών.

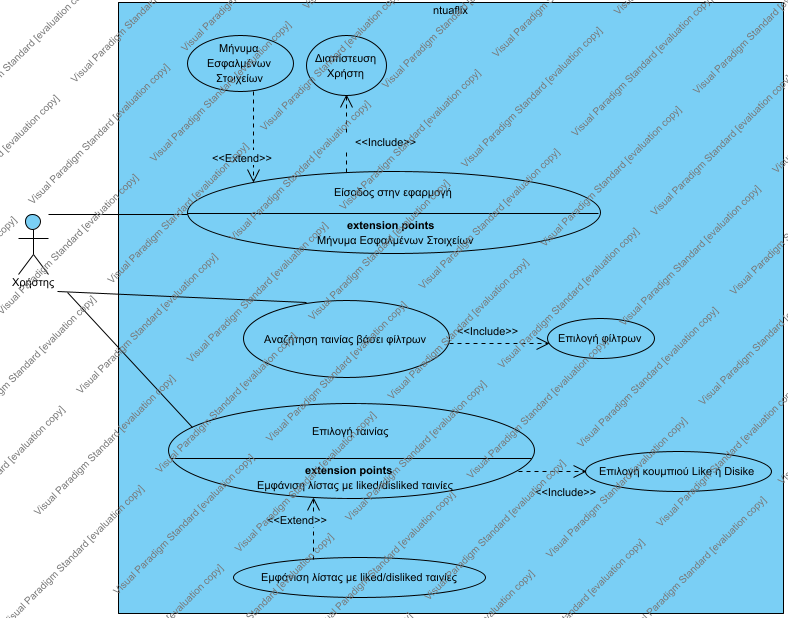
* **Front-End Εφαρμογή:** Χρησιμοποιώντας τον ιστότοπο της εφαρμογής, ο χρήστης θα έχει την δυνατότητα να εγγραφεί ως διαχειριστής, να αναζητήσει ταινίες με βάση διάφορα κριτήρια και να επιλέξει ποιες του αρέσουν ή όχι. Παράλληλα, έχει την δυνατότητα να περιηγηθεί σε χιλιάδες τίτλους ταινιών και σειρών.
* **RESTful API:** Η εφαρμογή παρέχει επιπλέον την δυνατότητα πρόσβασης στα δεδομένα και τις λειτουργίες της πλατφόρμας μας, μέσω κατάλληλου συγκεκριμένου RESTful API. Το συγκεκριμένο API θα είναι διαθέσιμο μέσω του base url https://{{host}}:9867/ntuaflix\_api

# Αναφορές - πηγές πληροφοριών

Ν/Α.

# Προδιαγραφές απαιτήσεων λογισμικού

## 3.1 Περιπτώσεις χρήσης

Αναλύονται 2 περιπτώσεις χρήσης, η επιλογή ταινιών με φίλτρα όπως ηθοποιοί, έτος κυκλοφορίας, είδος, καθώς και το πάτημα κουμπιών Like/Dislike για κάθε ταινία ανάλογα με τις προτιμήσεις του χρήστη. Πιο κάτω φαίνεται το use case diagram.   
  


### 3.1.1 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 1: Φιλτράρισμα ταινίας

#### 3.1.1.1 Χρήστες (ρόλοι) που εμπλέκονται

Ο μοναδικός ρόλος που εμπλέκεται στην συγκεκριμένη περίπτωση χρήσης είναι ο εγγεγραμμένος που επιθυμεί να φιλτράρει τις ταινίες.

#### 3.1.1.2 Προϋποθέσεις εκτέλεσης

Ο χρήστης να έχει συνδεθεί επιτυχώς στην εφαρμογή

*3.1.1.3 Περιβάλλον εκτέλεσης*  
  
Μπορεί να υλοποιηθεί τόσο στη διαδικτυακή διεπαφή όσο και στο μέσω κλήσης στο REST API.

*3.1.1.4 Δεδομένα εισόδου*

Ο χρήστης επιλέγει με βάση ποιο ή ποια φίλτρα θέλει να φιλτράρει την αναζήτηση ταινιών. Οι επιλογές είναι με βάση όνομα ηθοποιού, έτος κυκλοφορίας, είδος ταινίας.

*3.1.1.5 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά*

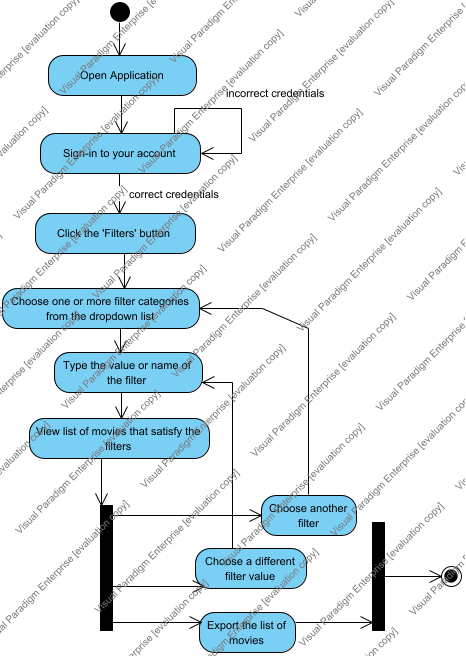
Η αλληλουχία ενεργειών περιγράφεται και οπτικά με τα διαγράμματα UML που παρατίθενται.

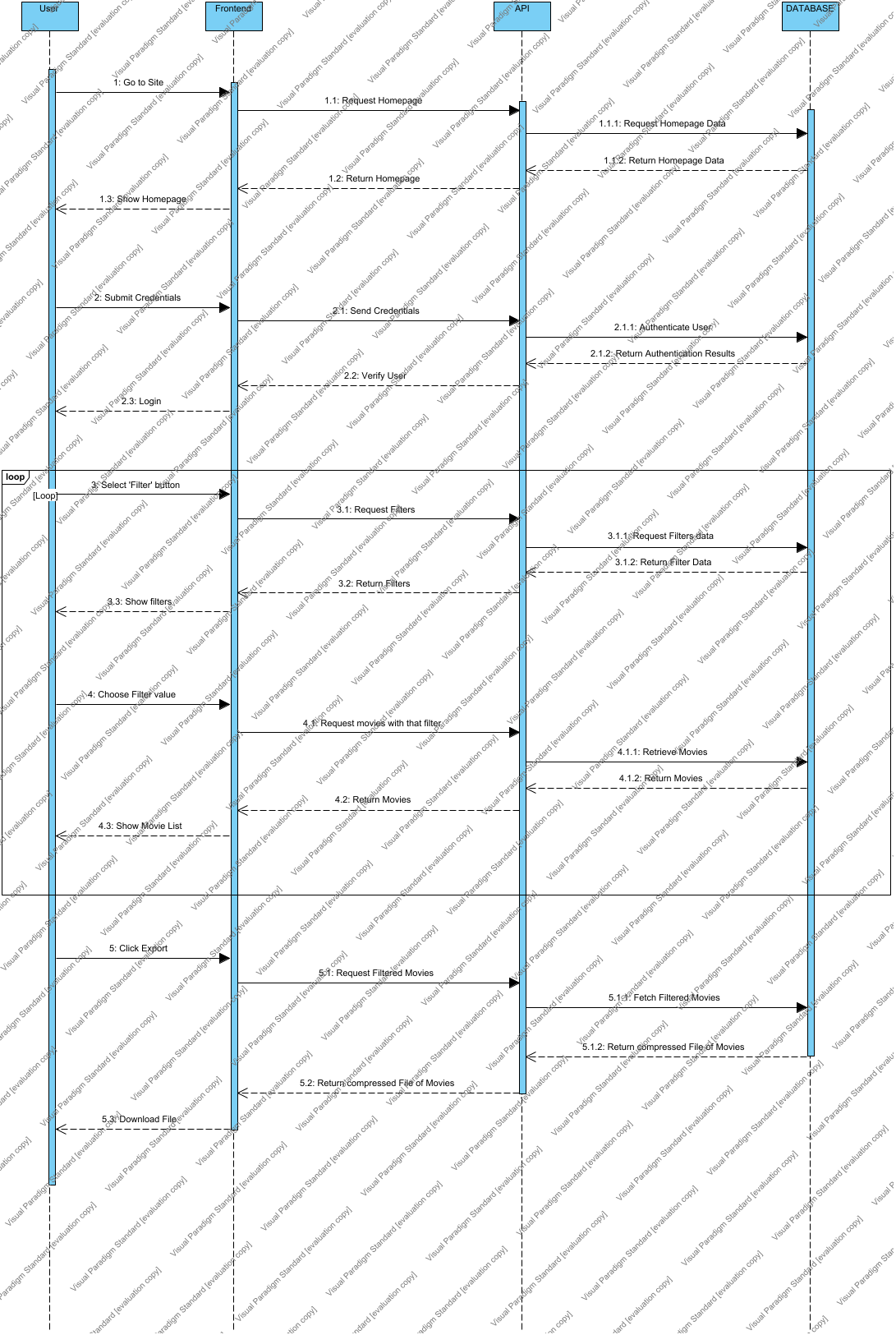
Βήμα 1: Ο χρήστης ανοίγει την εφαρμογή

Βήμα 2: Συνδέεται έγκυρα στον λογαριασμό του

Βήμα 3 : Επιλέγει το κουμπί ‘Filter’ και εμφανίζει τη λίστα με τις επιλογές φίλτρων

Βήμα 4 : Επιλέγει φίλτρο και στο σχετικό πεδίο δίνει την τιμή ή το όνομα για το οποίο θέλει να εφαρμόσει το φίλτρο και έτσι αποστέλλει το αίτημα στο REST API το οποίο επιστρέφει τη λίστα ταινιών που ικανοποιούν το φίλτρο.

Βήμα 5: Ο χρήστης επιλέγει να κάνει export την λίστα των ταινιών οπότε του επιστρέφεται συμπιεσμένο αρχείο της σχετικής λίστας  
  
Διάγραμμα **UML Activity:**

Διάγραμμα **UML Sequence**:

#### 3.1.1.7 Δεδομένα εξόδου

Δεδομένα εξόδου τη συγκεκριμένης περίπτωσης χρήσης αποτελεί η λίστα με τις ταινίες που ικανοποιούν το επιλεγμένο φίλτρο.

### 3.1.2 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 2: Πάτημα Like/ Dislike

#### 3.1.2.1 Χρήστες (ρόλοι) που εμπλέκονται

Ο μοναδικός ρόλος που εμπλέκεται στην συγκεκριμένη περίπτωση χρήσης είναι ο εγγεγραμμένος που επιθυμεί να φιλτράρει τις ταινίες.

#### 3.1.2.2 Προϋποθέσεις εκτέλεσης

Ο χρήστης να έχει συνδεθεί επιτυχώς στην εφαρμογή

#### 3.1.2.3 Περιβάλλον εκτέλεσης

Μπορεί να υλοποιηθεί μόνο στη διαδικτυακή διεπαφή και στη συνέχεια ενημερώνεται αντίστοιχα η βάση

#### 3.1.2.4 Δεδομένα εισόδου

Ο χρήστης επιλέγει την ταινία που θέλει να κάνει like/dislike

#### 3.1.2.5 Παράμετροι

#### 3.1.2.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

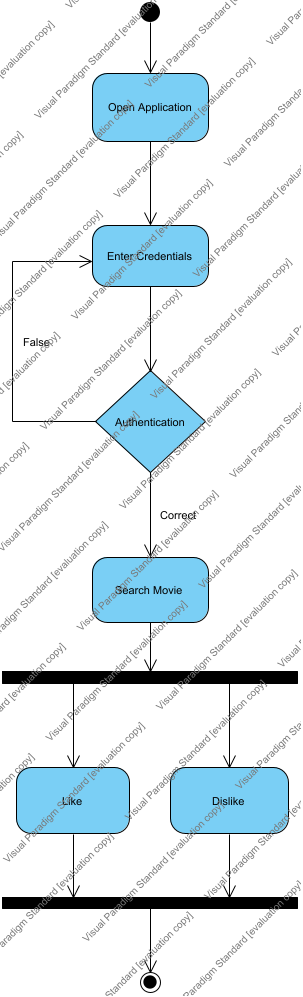
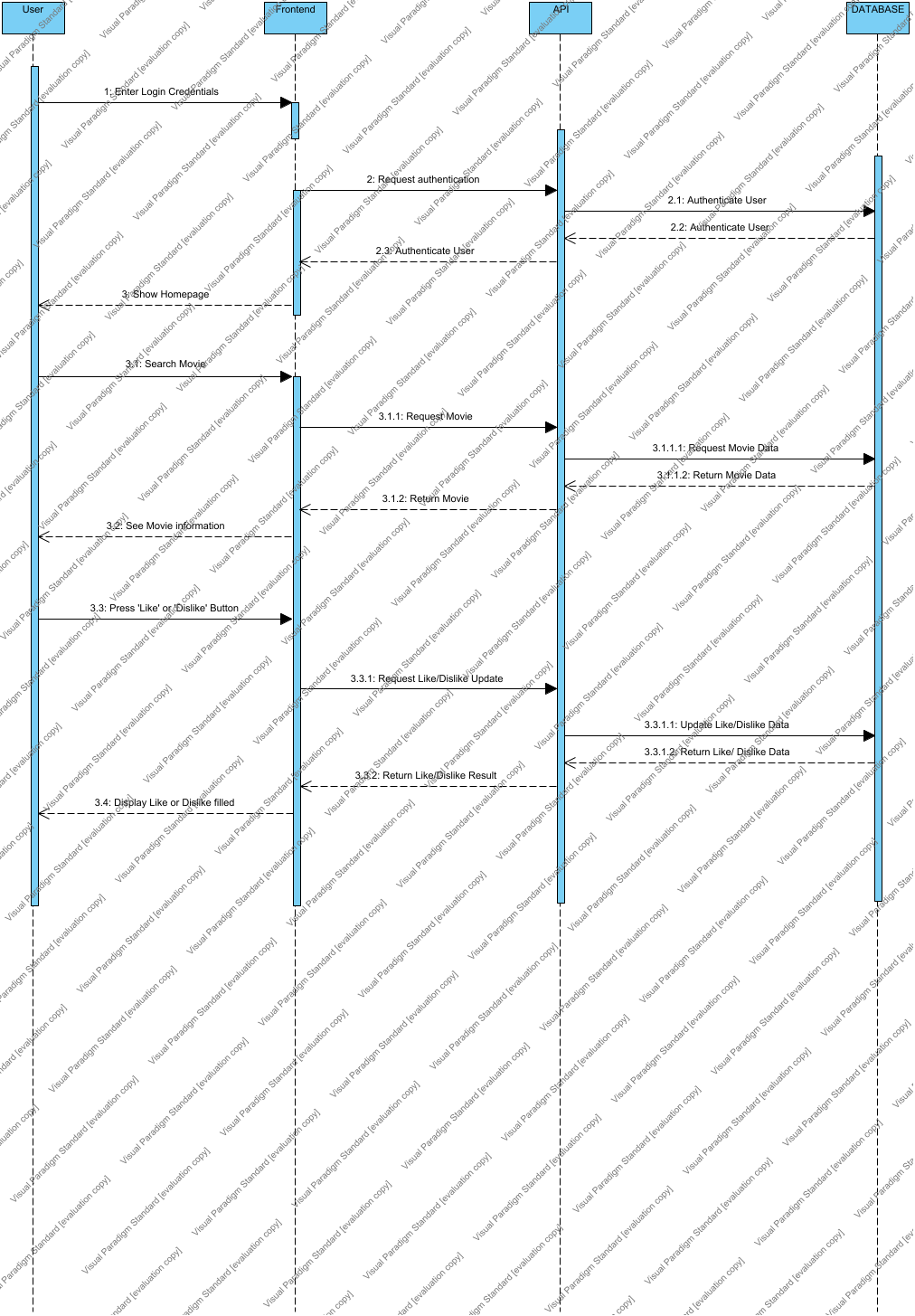
Η αλληλουχία ενεργειών περιγράφεται και οπτικά με τα διαγράμματα UML που παρατίθενται.

Βήμα 1: Ο χρήστης συνδέεται επιτυχώς στην εφαρμογή

Βήμα 2: Στο αντίστοιχο πεδίο κάνει αναζήτηση του τίτλου της ταινίας που θέλει και έτσι πραγματοποιείται η αντίστοιχη κλήση API που επιστρέφει την ταινία από τη βάση.

Βήμα 3: Ο χρήστης επιλέγει ένα από τα κουμπιά like ή dislike και το πατάει. Έτσι το API κάνει κλήση για ανανέωση του count του αντίστοιχου πεδίου και έτσι ανανεώνεται ο αριθμός like/dislike που εμφανίζεται στο χρήστη

Διάγραμμα **UML Activity:**

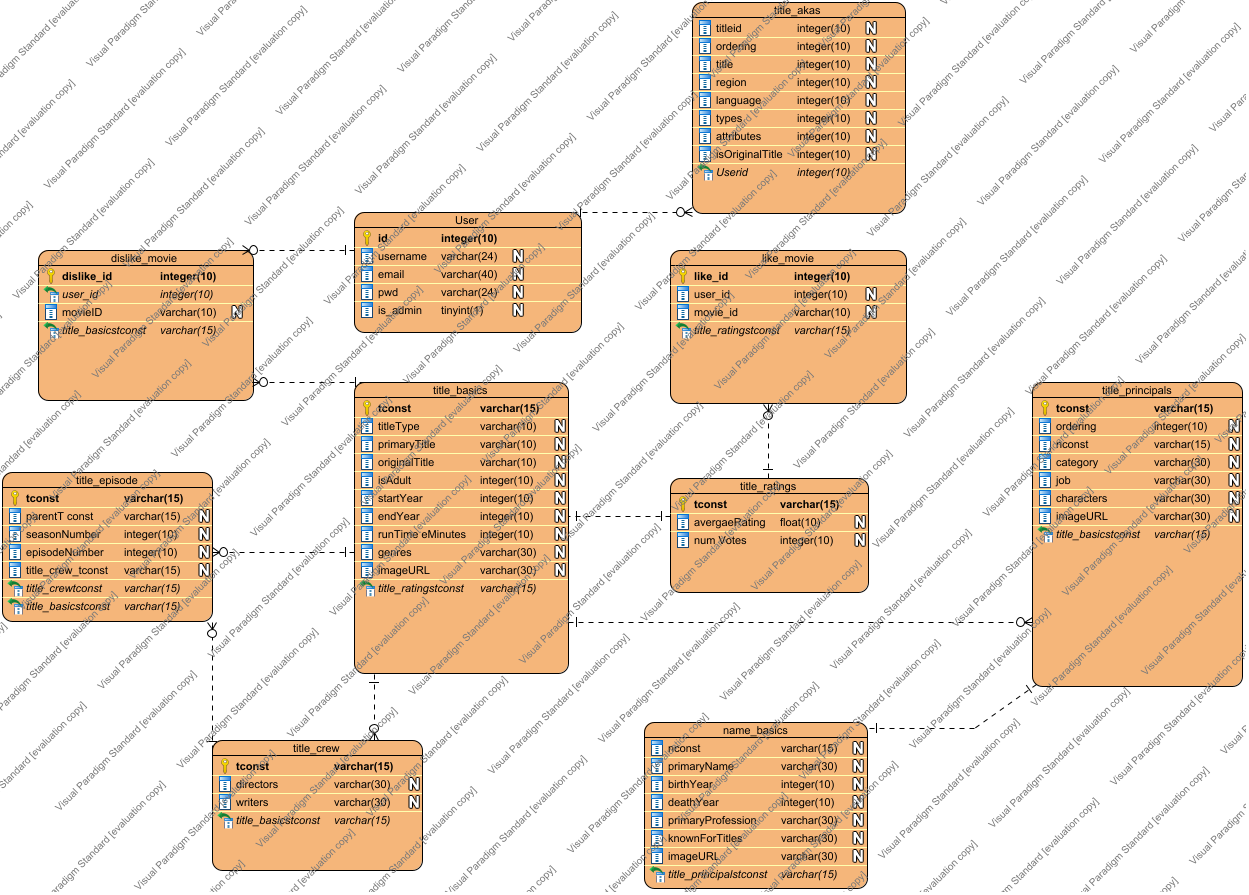
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
Διάγραμμα **UML Sequence**:

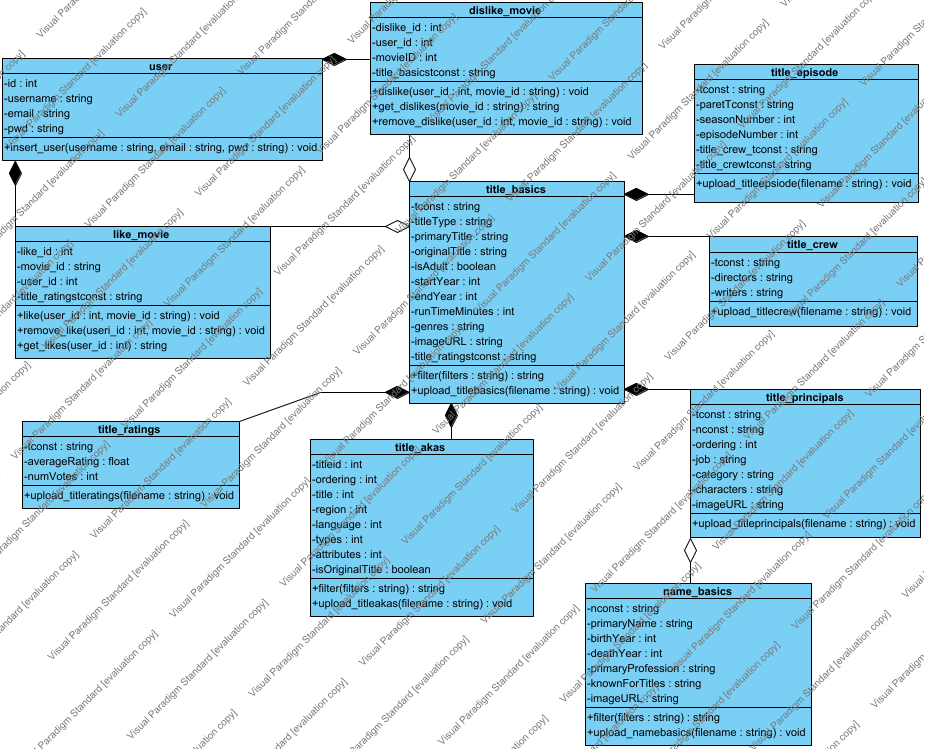
#### 3.1.2.7 Δεδομένα εξόδου

Δεδομένο εξόδου τη συγκεκριμένης περίπτωσης χρήσης αποτελεί η αλλαγή χρώματος του εικονιδίου που δηλώνει ότι η ταινία είναι liked ή disliked.

## 3.2 Απαιτήσεις επιδόσεων

Η εφαρμογή μας θα είναι διαθέσιμη σε εγγεγραμμένους χρήστες. Αναμένουμε σε ώρες αιχμής να υπάρχει φορτίο μερικών δεκάδων διαχειριστών, και μερικών εκατοντάδων χρηστών που θα μπορούμε να διαχειριστούμε. Παρόλο που η συνεχής διαθεσιμότητα δεν είναι κρίσιμη (critical), είναι σημαντικό ότι δεν θα χάνονται δεδομένα ταινιών.

3.3 Απαιτήσεις οργάνωσης δεδομένων   
  
Ακολουθεί το UML **Entity Relation Diagram:**  
Ακολουθεί το UML **Class Diagram:**



### 3.3.1 Απαιτήσεις και περιορισμοί πρόσβασης σε δεδομένα

Κανένας χρήστης εγγεγραμμένος ή μη δεν έχει προσβαση στα δεδομένα της βάσης δεδομένων παρά μόνο αν αυτά αφορούν το λογαριασμό του. Βέβαια μπορούν να κάνουν αιτήματα προς τους admins ώστε να καταχωρούνται επιπλέον δεδομένα στη βάση.

Οι admins μόνο έχουν πρόσβαση στα δεδομένα και μπορούν να μεταβάλλουν το περιεχόμενό τους στη βάση δεδομένων. Φυσικά και εκείνοι περιορίζονται από κρυπτογράφηση στα προσωπικά δεδομένα των χρηστών.

## 3.4 Περιορισμοί σχεδίασης

Front-End

* Για τη μορφοποίηση, τη διαδραστικότητα και τη γενική διαμόρφωση της ιστοσελίδας θα γίνει η χρήση **Javascript**, η οποία συνεπάγεται κάποιους περιορισμούς στην ταχύτητα, ανάλογα με τη χρήση της, καθώς και τη μεταβλητότητα που μπορεί να παρουσιάσει σε διαφορετικούς browsers.
* Στην συγγραφή και οργάνωση του κώδικα Javascript αξιοποιούμε τη βιβλιοθήκη **React**, η οποία μας παρέχει μεγαλύτερη ευκολία στην δημιουργία του front-end σε σύγκριση με απλό κώδικα Javascript.
* Μέσα στα αρχεία της Javascript ενσωματώνουμε κώδικες **HTML** για την κατασκευή του σκελετού της εφαρομογής και **CSS** για την μορφοποίησή της

Back-End

* Χρήση της γλώσσας **Javascript** για την υλοποίηση του back-end της εφαρμογής.
* Χρήση του **NodeJS** και του εργαλείου αυτόματου χτισίματος **npm** για αυτοματισμό της επίλυσης εξαρτήσεων λογισμικού εντός του project και αυτόματη εκτέλεση των τεστ λειτουργικότητας.

CLI

* Το Command Line Interface γράφτηκε σε Python με χρήση της βιβλιοθήκης **argparse** (parser for command-line options, arguments and sub-commands).

## 3.5 Λοιπές απαιτήσεις

### 3.5.1 Απαιτήσεις διαθεσιμότητας λογισμικού

Η εφαρμογή είναι επιθυμητό να είναι διαθέσιμη οποιαδήποτε ώρα της ημέρας για όλα τα είδη χρηστών.

### 3.5.2 Απαιτήσεις ασφάλειας

* Ασφαλή κανάλια για τη σύνδεση των χρηστών.
* Χρήση HTTPS και SSL certificate.
* Αυτόματη έξοδος χρηστών ανενεργών για μεγάλο χρονικό διάστημα.
* Κωδικοί πρόσβασης και άλλα ευαίσθητα προσωπικά δεδομένα δεν θα πρέπει να εμφανίζονται ή να αποθηκεύονται σε υπολογιστές χρηστών μέσω cookies ή με οποιοδήποτε άλλον τρόπο.
* Οι κωδικοί πρόσβασης θα πρέπει να είναι κρυπτογραφημένοι στη βάση δεδομένων.

### 3.5.2 Απαιτήσεις Συντήρησης

Απαιτούνται τακτικά backups στο database ώστε να διασφαλίζεται η ύπαρξη των δεδομένων ακόμα και σε ακραίες περιπτώσεις κατάρρευσης του συστήματος.

Ο web server μας είναι πιθανόν να χρειαστεί restart μετά από κάποια updates που ίσως υλοποιηθούν.

Συντήρηση του συστήματος θα εκτελείται κάθε 3 μήνες και σε έκτακτες περιστάσεις. Η συχνότητα αυτή υπόκειται σε αλλαγές ανάλογα με τον αριθμό χρηστών και την απόδοση του συστήματος.