ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Εφαρμογή διαχείρισης πολυμεσικού περιεχομένου και ανάλυση των τεχνικών συνεχόμενης ροής

Του φοιτητή
Ευστράτιου Σαμουηλίδη
Αρ. Μητρώου: 144249

Επιβλέπων
Ονοματεπώνυμο
ΙΓΝΑΤΙΟΣ ΔΕΛΗΓΙΑΝΝΗΣ
Βαθμίδα Καθηγητής

Ημερομηνία 20/6/2022

Τίτλος Δ.Ε. Εφαρμογή διαχείρισης πολυμεσικού περιεχομένου και ανάλυση των τεχνικών συνεχόμενης ροής

Κωδικός Δ.Ε. 21217

Ονοματεπώνυμο φοιτητή Ευστράτιος Σαμουηλίδης

Ονοματεπώνυμο εισηγητή Ιγνάτιος Δεληγιάννης

Ημερομηνία ανάληψης Δ.Ε. 28/03/2021

Ημερομηνία περάτωσης Δ.Ε. 15/06/2022

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Πρόλογος

Το streaming και κατ' επέκταση και το live streaming έχει γίνει πιο σημαντικό από ποτέ στις μέρες μας. Οι έννοιες αυτές είναι πλέον ριζωμένες στην καθημερινή ζωή και η αύξηση της ζήτησης για την τεχνολογία αυτή συνέπεσε με την μαζική στροφή προς τις online διασκέψεις μεταξύ επιχειρήσεων, οργανισμών αλλά και ατόμων. Πριν από κάποια χρόνια, και μόνο η ιδέα να πραγματοποιηθεί streaming στο διαδίκτυο προκαλούσε άγχος. Από τα αρχικά lockdown που σχετίζονται με τον COVID-19 και εφαρμόστηκε σε παγκόσμια κλίμακα, πολλές καθημερινές δραστηριότητες και εκδηλώσεις πραγματοποιήθηκαν εικονικά. Σύμφωνα και με έρευνες μεγάλων streaming υπηρεσιών, η πανδημία προκάλεσε έντονη αύξηση του online streaming, είτε πρόκειται για την εργασία των ανθρώπων είτε για την ψυχαγωγία τους. Η παρούσα πτυχιακή έχει ως στόχο την δημιουργία μιας εφαρμογής για online παρακολούθηση ταινιών, καθώς και μια εισαγωγή στον κόσμο του streaming.

Περίληψη

Στην συγκεκριμένη πτυχιακή, που πρόκειται για μία web εφαρμογή για online παρακολούθηση ταινιών, θα μιλήσουμε για την λειτουργία του streaming, τα εργαλεία που χρησιμοποιήθηκαν για την δημιουργία της και τις διάφορες δυνατότητες που παρέχει η εφαρμογή στον χρήστη. Η εργασία παρέχει μία καλή εισαγωγική εικόνα στον κόσμο του streaming, καθώς θα γίνει ανάλυση για τα διάφορα streaming πρωτόκολλα και που αυτά χρησιμοποιούνται. Τέλος, θα γίνει αναφορά σε ορισμένα αρχιτεκτονικά μοτίβα λογισμικού καθώς και σε ορισμένες αρχές σχεδίασης λογισμικού που χρησιμοποιήθηκαν για την ανάπτυξη της.

Multimedia content management application and analysis of streaming techniques

Samouilidis Efstratios

Abstract

In this dissertation, which is a web application about watching movies online, we will talk about streaming, the tools used to create the application and the various features the application provides to the users. The dissertation provides a good introduction in the world of streaming, as there is going to be analysis on the various streaming protocols and their use cases. Finally, there will be a reference to certain software architectural patterns as well as to some software design principles used for the application's development.

Ευχαριστίες

Στο σημείο αυτό θα χρειαστεί να ευχαριστήσω τον κύριο Ιγνάτιο Δεληγιάννη για την συνολική συνεργασία που είχαμε για την ολοκλήρωση της πτυχιακής εργασίας. Οι ιδέες που μου πρότεινε για την εφαρμογή ενσωματώθηκαν και την βελτίωσαν αισθητά. Επίσης, η ανταπόκριση του στην επικοινωνία μας ήταν πάντα άμεση.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1ο: Περιγραφή του τεχνολογικού περιβάλλοντος που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής

1.1 Εισαγωγή

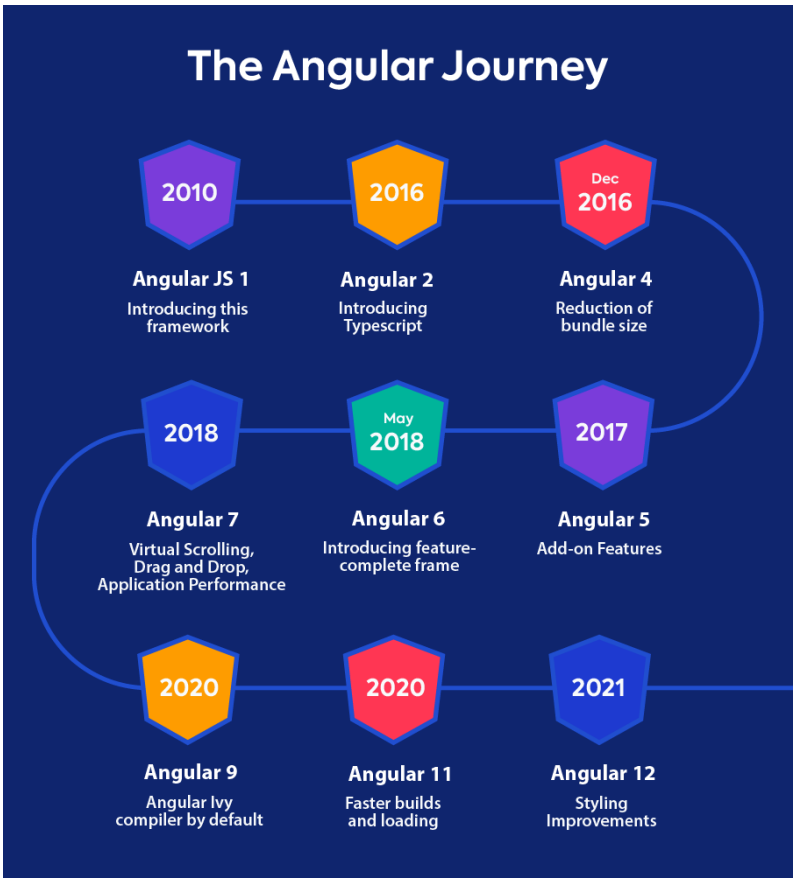
Στο συγκεκριμένο κεφάλαιο θα μιλήσουμε για τις γλώσσες προγραμματισμού, τα frameworks, την βάση δεδομένων, τις βιβλιοθήκες, το περιβάλλον ανάπτυξης της εφαρμογής καθώς και για το github. Επίσης θα γίνει μία σύγκριση της βάσης (MongoDB) που χρησιμοποιήθηκε για την δημιουργία της εφαρμογής σε σχέση με την MySQL που αποτελεί μία από τις κορυφαίες βάσεις για web application development. Τέλος θα γίνει και μία αναφορά στην χρήση του Pusher.

1.2 Frameworks

Angular

Πρόκειται για είναι ένα πλαίσιο εφαρμογών ιστού, δωρεάν και ανοιχτού κώδικα που βασίζεται σε TypeScript, υπό την ηγεσία της Angular Team της Google και από μια κοινότητα ατόμων και εταιρειών. Με την Angular, μπορούμε μα εκμεταλλευτούμε μια πλατφόρμα για ανάπτυξη έργων ενός απλού προγραμματιστή έως εφαρμογές εταιρικού επιπέδου.

Στο σημείο αυτό αξίζει να αναφέρουμε ότι η πρώτη έκδοση της angular δημιουργήθηκε το 2010 από έναν υπάλληλο της Google με το όνομα Miško Hevery, ο οποίος έκανε developing ένα side project. Το project αυτό είχε ως στόχο να διευκολύνει τη δημιουργία διαδικτυακών εφαρμογών για μερικά εσωτερικά έργα στα οποία εργαζόταν. Αυτό το project έγινε αργότερα γνωστό ως AngularJS (Angular λόγω του <> σε HTML). Στην σημερινή εποχή έχει υιοθετηθεί από μεγάλες εταιρίες παγκόσμιου βεληνεκούς. Ορισμένα παραδείγματα ιστοσελίδων που βασίζονται στην Angular είναι της Google, της Gmail, της Paypal και της Microsoft office.



Σχήμα 1.1 Ιστορική αναδρομή στις διάφορες εκδόσεις της Angular

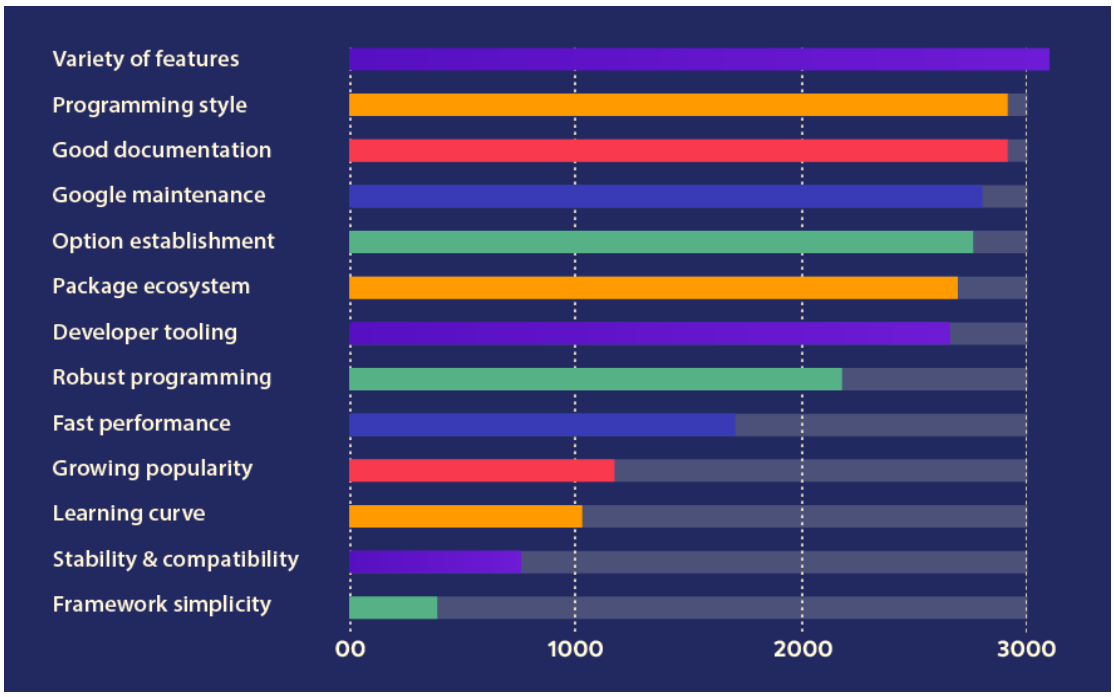
Όλες οι μεγάλες εκδόσεις υποστηρίζονται για 18 μήνες. Αυτό αποτελείται από 6 μήνες ενεργής υποστήριξης, κατά τη διάρκεια των οποίων κυκλοφορούν τακτικά προγραμματισμένες ενημερώσεις και ενημερώσεις κώδικα. Στη συνέχεια ακολουθείται από 12 μήνες μακροπρόθεσμης υποστήριξης (LTS), κατά τη διάρκεια των οποίων κυκλοφορούν μόνο κρίσιμες επιδιορθώσεις και ενημερώσεις κώδικα ασφαλείας

Version	Status	Released	Active Ends	LTS Ends
14.0.0	Active	Jun 02, 2022	Dec 02, 2022	Dec 02, 2023
13.0.0	LTS	Nov 04, 2021	Jun 02, 2022	May 04, 2023
12.0.0	LTS	May 12, 2021	Nov 12, 2021	Nov 12, 2022

Πίνακας 1.1 Υποστηριζόμενες εκδόσεις Angular

Γιατί να χρησιμοποιήσει κάποιος angular;

1. Ένα από τα μεγαλύτερα πλεονεκτήματα του Angular είναι ότι υποστηρίζεται από την Google. Η Google προσφέρει τη Μακροπρόθεσμη Υποστήριξή της (LTS) στην Angular που ρίχνει φως στο σχέδιο της Google να διατηρήσει το συγκεκριμένο framework και να κλιμακώσει περαιτέρω. Οι εφαρμογές της Google χρησιμοποιούν επίσης το Angular και η ομάδα τους είναι αρκετά αισιόδοξη για τη σταθερότητά του.
2. Οι Angular εφαρμογές κατασκευάζονται χρησιμοποιώντας τη γλώσσα TypeScript, που αποτελεί superscript της JavaScript, που εξασφαλίζει υψηλότερη ασφάλεια καθώς υποστηρίζει types(primitives και interfaces) Βοηθά στον εντοπισμό και την εξάλειψη σφαλμάτων νωρίς στη διαδικασία κατά τη σύνταξη του κώδικα ή την εκτέλεση εργασιών συντήρησης. Μπορούμε επίσης να διορθώσουμε απευθείας τον κώδικα TypeScript στο πρόγραμμα περιήγησης ή σε ένα πρόγραμμα επεξεργασίας, εάν έχουν δημιουργηθεί τα κατάλληλα maps κατά το build. Αυτή η γλώσσα διασφαλίζει βελτιωμένες υπηρεσίες πλοήγησης, ανακατασκευής και αυτόματης συμπλήρωσης.
3. Χρησιμοποιεί HTML για να ορίσει τη διεπαφή χρήστη (UI) της εφαρμογής. Η HTML, σε σύγκριση με την JavaScript, είναι μια λιγότερο περίπλοκη γλώσσα.
4. Είναι ενσωματωμένη με την αρχική αρχιτεκτονική εγκατάσταση λογισμικού MVC (Model-View-Controller). Ωστόσο, δεν ζητά από τους προγραμματιστές να χωρίσουν μια εφαρμογή σε διαφορετικά στοιχεία MVC και να δημιουργήσουν έναν κώδικα που θα μπορούσε να τις ενώσει. Επίσης εξασφαλίζει εύκολη ανάπτυξη καθώς εξαλείφει την ανάγκη για περιττό κώδικα (getters και setters).
5. Οργανώνει τον κώδικα σε buckets, είτε πρόκειται components, directives, pipes, ή services. Όσοι είναι εξοικειωμένοι με το Angular αναφέρονται σε αυτά τα buckets ως modules. Τα modules καθιστούν εύκολη την οργάνωση της λειτουργικότητας της εφαρμογής, διαχωρίζοντάς την σε χαρακτηριστικά και επαναχρησιμοποιήσιμα κομμάτια.

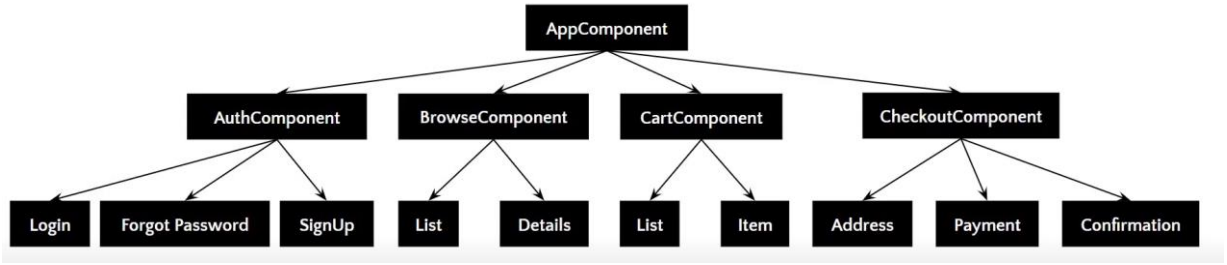


Σχήμα 1.2 Αναπαράσταση των πιο αρεστών aspects της angular μεταξύ των developers

Angular Components

Τα components είναι το πιο βασικό δομικό στοιχείο διεπαφής χρήστη (UI) μιας εφαρμογής Angular. Μια εφαρμογή Angular περιέχει ένα δέντρο από components. Κάθε component αποτελείται από:

- Ένα template HTML που δηλώνει τι θα κάνει render στη σελίδα
- Μια κλάση TypeScript που ορίζει τη συμπεριφορά
- Ένας CSS selector που ορίζει πώς χρησιμοποιείται το component σε ένα template
- Προαιρετικά, τα CSS styles που εφαρμόζονται στο template



Σχήμα 1.3 Tree structure components

Χαρακτηριστικά Components

1. Τα components είναι συνήθως custom HTML elements
2. Μια κλάση TypeScript χρησιμοποιείται για τη δημιουργία ενός component. Στη συνέχεια, αυτή η κλάση εμπεριέχει με τον decorator "@Component".
3. Ο decorator δέχεται ένα αντικείμενο μεταδεδομένων που δίνει πληροφορίες για το component.
4. Ένα component πρέπει να ανήκει στο NgModule για να μπορεί να χρησιμοποιηθεί από άλλο component ή εφαρμογή.
5. Τα components ελέγχουν τη συμπεριφορά τους στο χρόνο εκτέλεσης εφαρμόζοντας Life-Cycle hooks.

Express

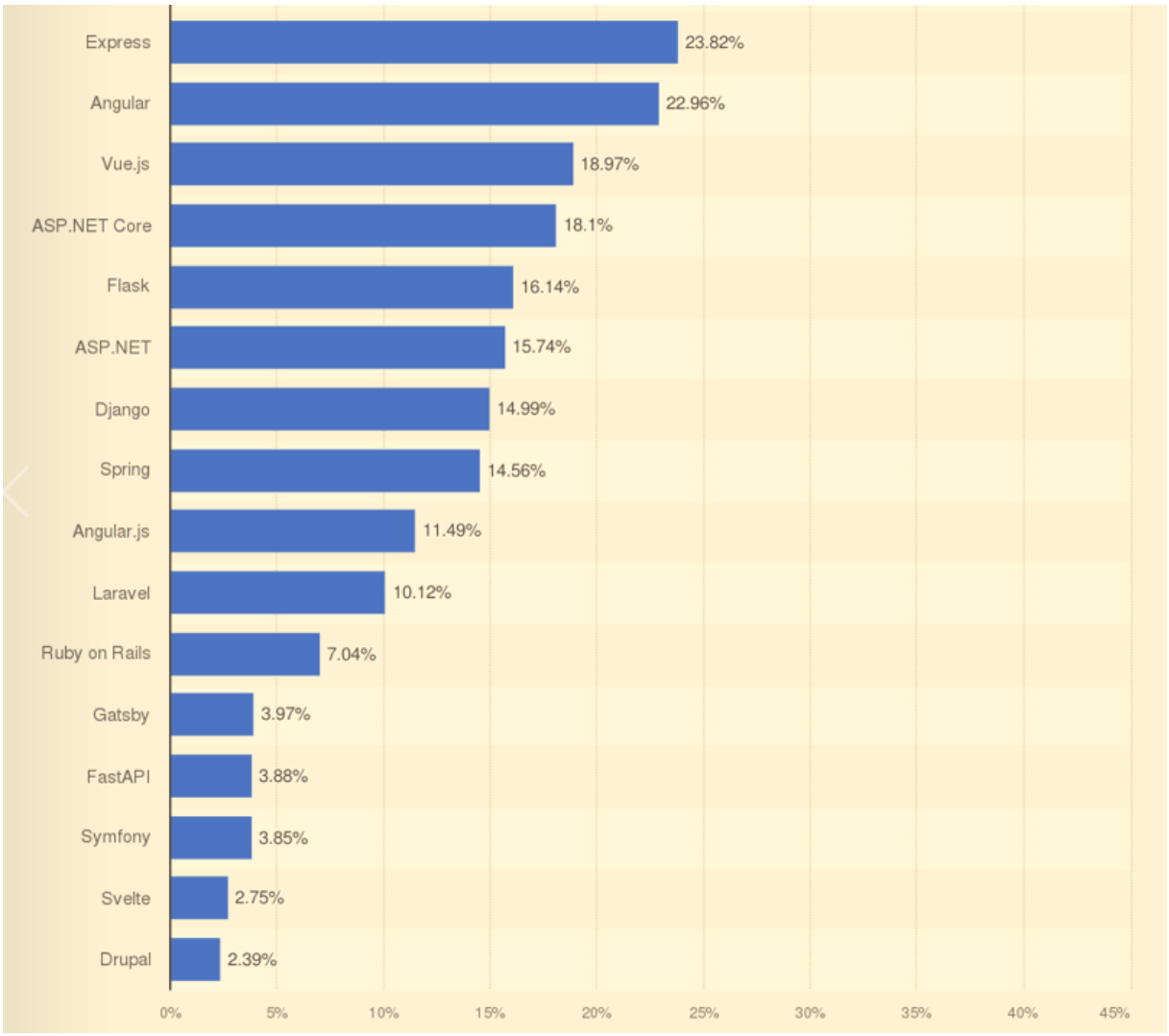
Η Express.jsή απλώς Expressιδρύθηκε από τον TJ Holowaychuk. Η πρώτη κυκλοφορία, σύμφωνα με το ήταν στις 22 Μαΐου 2010. Η Express είναι ένα minimal και ευέλικτο backend framework για την Node.js που παρέχει ένα ισχυρό σύνολο δυνατοτήτων για την ανάπτυξη εφαρμογών ιστού και κινητών. Διευκολύνει την ταχεία ανάπτυξη διαδικτυακών εφαρμογών που βασίζονται σε Node.js. Επίσης, πρόκειται για το πιο διάσημο framework της Node και αποτελεί την βάση από την οποία αποτελούνται και άλλα διάσημα frameworks.

Από την στιγμή που η Express απαιτεί μόνο Javascript, κάνει ευκολότερο για του προγραμματιστές να δημιουργήσουν web εφαρμογές και API χωρίς κόπο. Είναι αρκετά “ελαφριά” και βοηθάει στην οργάνωση web εφαρμογών από την πλευρά του server χρησιμοποιώντας την αρχιτεκτονική MVC. Η

Express είναι κυρίως υπεύθυνη για το χειρισμό του backend στη στοίβα MEAN. Το Mean Stack είναι η στοίβα λογισμικού JavaScript ανοιχτού κώδικα που χρησιμοποιείται ευρέως για τη δημιουργία δυναμικών ιστοτόπων και εφαρμογών Ιστού στην αγορά. Το MEAN σημαίνει MongoDB, Express.js, και Node.js.

Μερικά από τα βασικά χαρακτηριστικά της express είναι τα παρακάτω:

- Επιτρέπει τη ρύθμιση ενδιάμεσων προγραμμάτων (middlewares) για να ανταποκρίνονται σε αιτήματα HTTP. Το Middleware είναι ένας χειριστής αιτημάτων που έχει πρόσβαση στον κύκλο αίτησης-απόκρισης της εφαρμογής.
- Καθορίζει έναν πίνακα δρομολόγησης (routing table) που χρησιμοποιείται για την εκτέλεση διαφορετικών ενεργειών με βάση τη μέθοδο HTTP και τη διεύθυνση URL. Το routing αναφέρεται στον τρόπο με τον οποίο τα endpoint URL μιας εφαρμογής ανταποκρίνονται σε αιτήματα πελατών.
- Επιτρέπει τη δυναμική απόδοση σελίδων HTML με βάση την μεταβίβαση ορισμάτων στα templates.
- Η Express καθιστά τα πράγματα ευκολότερα καθώς προσδιορίζει το ακριβές μέρος όπου βρίσκονται τα σφάλματα.



Σχήμα 1.3 Έρευνα σχετικά με τα πιο χρησιμοποιούμενα frameworks

1.3 Περιβάλλον Ανάπτυξης

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε Node.js. Η Node.js δημιουργήθηκε από τον Ryan Dahl το 2009. Η δημιουργία και η συντήρηση του έργου χορηγήθηκε από την εταιρία Joyent. Η ιδέα για την ανάπτυξη του node προήλθε από την ανάγκη του Ryan Dahl να βρει τον πιο αποδοτικό τρόπο να ενημερώνει τον χρήστη σε πραγματικό χρόνο για την κατάσταση ενός αρχείου που ανέβαζε στο διαδίκτυο. Επίσης επηρεάστηκε από το Mongrel του Zed Shaw. Επιπροσθέτως μετά από αποτυχημένα έργα σε C, Lua, Haskell η κυκλοφορία της μηχανής V8 (V8 JavaScript Engine) της Google τον ώθησε να ασχοληθεί με την Javascript. Είναι ένα λεπτό, γρήγορο, διαδικτυακό περιβάλλον εκτέλεσης JavaScript που είναι χρήσιμο τόσο για διακομιστές όσο και για διάφορες εφαρμογές του.

Στις μέρες μας, η καθυστέρηση και η απόδοση είναι βασικοί δείκτες απόδοσης για τους διακομιστές στο διαδίκτυο. Η διατήρηση της χαμηλής καθυστέρησης και η υψηλή απόδοση όσο αυξάνονται οι ανάγκες μιας ιστοσελίδας ή μιας εφαρμογής δεν είναι εύκολη. Η Node.js είναι λοιπόν βασισμένο σε ένα runtime JavaScript περιβάλλον που επιτυγχάνει χαμηλή λανθάνουσα κατάσταση και υψηλή απόδοση. Με άλλα λόγια, δεν σπαταλάει χρόνο ή πόρους για αναμονή Input/Output (I/O) αιτημάτων.

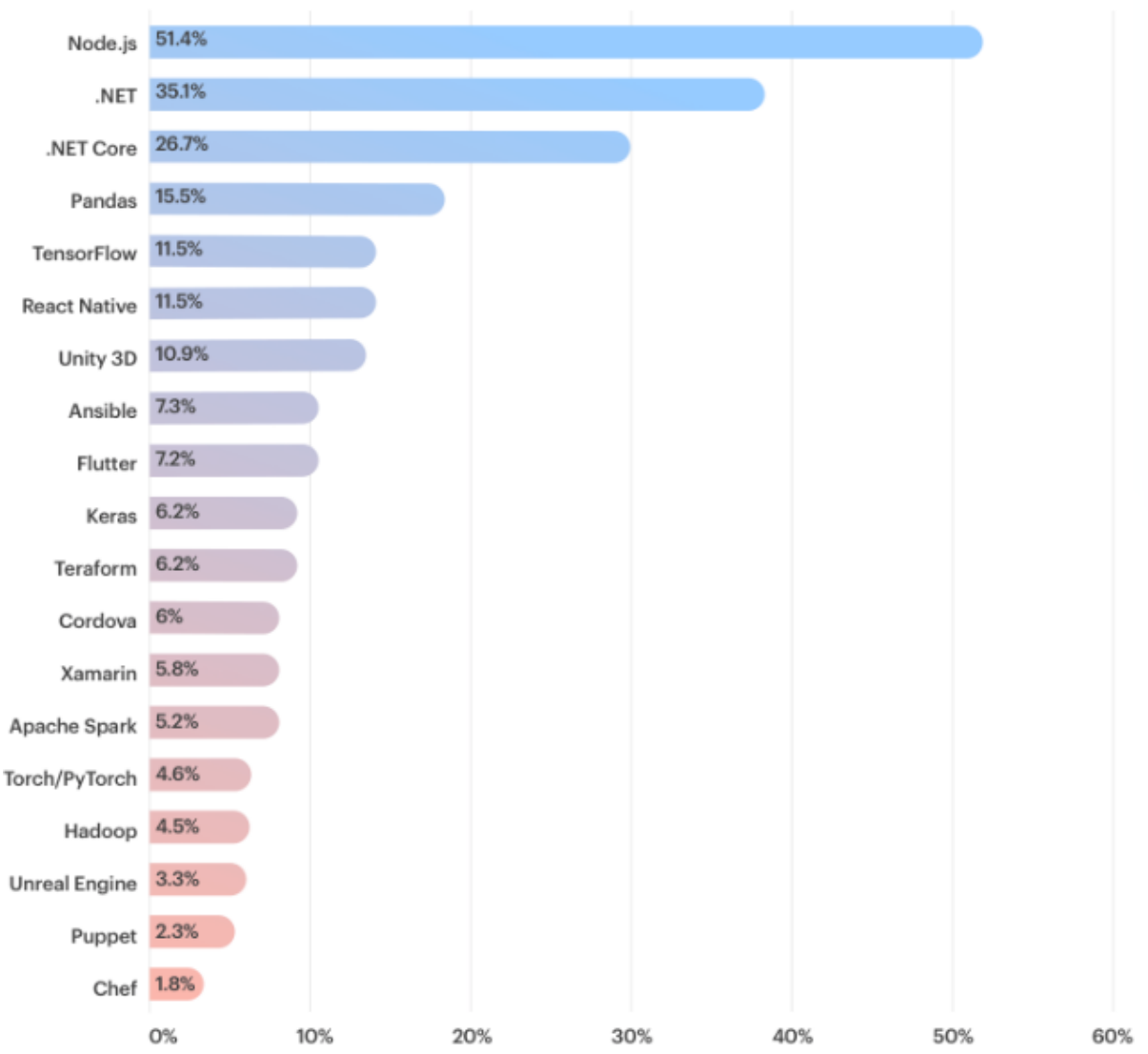
Η Node.js υιοθετεί μια διαφορετική προσέγγιση για την υλοποίηση των εισερχόμενων και εξερχόμενων νημάτων από τον διακομιστή. Εκτελεί έναν βρόχο συμβάντων με μια συγκεκριμένη διαδικασία που έχει καταχωρηθεί με το σύστημα για την διαχείριση συνδέσεων και κάθε νέα σύνδεση προκαλεί την πυροδότηση μιας λειτουργίας επανάκλησης JavaScript. Αυτή μπορεί να χειριστεί I/O αιτήματα με μη αποκλειστικές κλήσεις I/O και αν είναι απαραίτητο μπορεί να κρίνει ώστε να εκτελέσει λειτουργίες αποκλεισμού ή εντατικής χρήσης CPU και να ισορροπήσει φορτία μεταξύ πυρήνων της CPU όπου είναι αναγκαίο. Η προσέγγιση αυτή απαιτεί λιγότερη μνήμη για να χειριστεί περισσότερες συνδέσεις σε σχέση με τις περισσότερες ανταγωνιστικές αρχιτεκτονικές που κλιμακώνονται με νήματα, συμπεριλαμβανομένου του Apache HTTP Server, των διάφορων Java Server, των ISS και ASP.NET και του γνωστού Ruby on Rails Η Node είναι ιδανική για τη δημιουργία εφαρμογών που απαιτούν συνεχή σύνδεση μεταξύ του προγράμματος περιήγησης και του διακομιστή, συμπεριλαμβανομένων τέτοιων συνομιλιών, αναρτήσεων πολυμέσων και ειδοποιήσεων push web. Με απλά λόγια, η Node.js μπορεί να λειτουργήσει υπέροχα για έργα web σε πραγματικό χρόνο.

Σαν συμπέρασμα μπορούμε να πούμε ότι η Node.js δημιούργησε μια πλήρους στοίβας πλατφόρμα δημιουργίας διαδικτυακών εφαρμογών προτίμησης. Η Node.js έχει καλή απόδοση στην κωδικοποίηση και μετάδοση βίντεο και ήχου, μεταφορά πολλών αρχείων και ροή δεδομένων, λόγω του σχεδιασμού της που δεν εμποδίζει. Το τελευταίο θα μπορούσε να είναι εξαιρετικά χρήσιμο για εφαρμογές της τουριστικής βιομηχανία όπου πρέπει να έχουμε πρόσβαση σε δεδομένα από API διαφορετικών προμηθευτών. Το 61% των προγραμματιστών θεωρούν ότι η Node.js είναι ένα σημαντικό χαρακτηριστικό μακροπρόθεσμα.

Release	Status	Code Name	Release Date	Maintenance End
0.10.x	End-of-Life		11-03-2013	31-10-2016
0.12.x	End-of-Life		06-02-2015	31-12-2016
4.x	End-of-Life	Argon	08-09-2015	30-04-2018
5.x	End-of-Life		29-10-2015	30-06-2016
6.x	End-of-Life	Boron	26-04-2016	30-04-2019
7.x	End-of-Life		25-10-2016	30-06-2017
8.x	End-of-Life	Carbon	30-05-2017	31-12-2019
9.x	End-of-Life		01-10-2017	30-06-2018
10.x	End-of-Life	Dubnium	24-04-2018	30-04-2021
11.x	End-of-Life		23-10-2018	01-06-2019
12.x	End-of-Life	Erbium	23-04-2019	30-04-2022
13.x	End-of-Life		22-10-2019	01-06-2020
14.x	Maintenance LTS	Fermium	21-04-2020	30-04-2023
15.x	End-of-Life		20-10-2020	01-06-2021
16.x	Active LTS	Gallium	20-04-2021	30-04-2024
17.x	Current		19-10-2021	01-06-2022
18.x	Current		19-04-2022	30-04-2025
19.x	Planned		18-10-2022	01-06-2023
20.x	Planned		18-04-2023	30-04-2026

Πίνακας 1.2 Ιστορική αναδρομή στις εκδόσεις της Node.js

Η αυξανόμενη δημοτικότητα του το JavaScript έχει φέρει μαζί του πολλές βελτιώσεις και το σημερινό τοπίο της web development είναι δραματικά διαφορετικό. Τα πράγματα με τα οποία μπορούμε να κάνουμε σήμερα στον Ιστό το JavaScript. Η εκτέλεση στον ιστότοπο, καθώς και στην εφαρμογή, ήταν δύσκολο να φανταστεί κανείς μόλις πριν από λίγα χρόνια, ακόμη και ενσωματωμένη σε ρυθμίσεις sandbox, όπως Flash ή Java Applets. Η Node.js δεν σχεδιάστηκε ποτέ για να λύσει το πρόβλημα της κλιμάκωσης των μετρήσεων. Δημιουργήθηκε για να λύσει το πρόβλημα της κλιμάκωσης I/O, κάτι που κάνει πραγματικά καλά.



Σχήμα 1.4 Απόψεις προγραμματιστών σχετικά με το πιο χρησιμοποιούμενο εργαλείο

1.4 Βάση δεδομένων

MongoDB

Η MongoDB είναι μια βάση δεδομένων εγγράφων NoSQL χωρίς σχήματα (schema-less). Σημαίνει ότι μπορείτε να αποθηκεύσετε έγγραφα JSON σε αυτή και η δομή αυτών των εγγράφων μπορεί να ποικίλλει καθώς δεν επιβάλλεται όπως οι βάσεις δεδομένων SQL. Αυτό είναι ένα από τα πλεονεκτήματα της χρήσης NoSQL, καθώς επιταχύνει την ανάπτυξη εφαρμογών και μειώνει την πολυπλοκότητα των αναπτύξεων. Παρακάτω θα αναλυθούν κάποια κύρια χαρακτηριστικά.

Ad-hoc ερωτήματα για βελτιστοποιημένα, real-timeanalytics

Κατά το σχεδιασμό του σχήματος μιας βάσης δεδομένων, είναι αδύνατο να γνωρίζουμε εκ των προτέρων όλα τα ερωτήματα που θα εκτελεστούν από τους τελικούς χρήστες. Ένα ad hoc ερώτημα είναι μια βραχύβια εντολή της οποίας η τιμή εξαρτάται από μια μεταβλητή. Κάθε φορά που εκτελείται ένα ad hoc ερώτημα, το αποτέλεσμα μπορεί να είναι διαφορετικό, ανάλογα με τις εν λόγω μεταβλητές.

Η βελτιστοποίηση του τρόπου με τον οποίο χειρίζονται τα ad-hoc ερωτήματα μπορεί να κάνει σημαντική διαφορά σε κλίμακα, όταν μπορεί να χρειαστεί να ληφθούν υπόψη χιλιάδες έως εκατομμύρια μεταβλητές. Αυτός είναι ο λόγος για τον οποίο η MongoDB, μια ευέλικτη βάση δεδομένων σχημάτων προσανατολισμένη στα έγγραφα, ξεχωρίζει ως η επιλεγμένη πλατφόρμα βάσης δεδομένων cloud για εταιρικές εφαρμογές που απαιτούν ανάλυση σε πραγματικό χρόνο. Με την ad-hoc υποστήριξη ερωτημάτων που επιτρέπει στους προγραμματιστές να ενημερώνουν ad-hoc ερωτήματα σε πραγματικό χρόνο, η βελτίωση της απόδοσης έχει πολύ μεγάλο και σημαντικό ρόλο.

Η MongoDB field queries, range queries, και regular expression. Τα ερωτήματα μπορούν να επιστρέψουν συγκεκριμένα πεδία και επίσης να λάβουν υπόψη συναρτήσεις που καθορίζονται από τον χρήστη.

Κατάλληλο indexing για καλύτερες εκτελέσεις ερωτημάτων

Με το σωστό τρόπο, το indexing προορίζεται να βελτιώσει την ταχύτητα και την απόδοση αναζήτησης. Η αποτυχία να οριστούν σωστά οι κατάλληλοι δείκτες (indices) μπορεί και συνήθως θα οδηγήσει σε πολλά ζητήματα προσβασιμότητας, όπως προβλήματα με την εκτέλεση ερωτημάτων και την εξισορρόπηση φορτίου (load balancing)

Χωρίς τους σωστούς δείκτες (indices), μια βάση δεδομένων αναγκάζεται να σαρώσει έγγραφα ένα προς ένα για να εντοπίσει αυτά που ταιριάζουν με τη δήλωση ερωτήματος. Αλλά εάν υπάρχει κατάλληλο indexing για κάθε ερώτημα, οι αιτήσεις των χρηστών μπορούν να εκτελεστούν βέλτιστα από τον διακομιστή. Η MongoDB προσφέρει ένα ευρύ φάσμα δεικτών και χαρακτηριστικών με εντολές ταξινόμησης για συγκεκριμένες γλώσσες που υποστηρίζουν πολύπλοκα μοτίβα πρόσβασης σε σύνολα δεδομένων.

Συγκεκριμένα, οι δείκτες MongoDB μπορούν να δημιουργηθούν κατ' απαίτηση για να εξυπηρετούν σε πραγματικό χρόνο, συνεχώς μεταβαλλόμενα μοτίβα ερωτημάτων και απαιτήσεις εφαρμογών. Μπορούν επίσης να δηλωθούν σε οποιοδήποτε πεδίο σε οποιοδήποτε από τα Documents, συμπεριλαμβανομένων αυτών που είναι ένθετα (nested) μέσα σε πίνακες.

Replication για καλύτερη διαθεσιμότητα και σταθερότητα δεδομένων

Όταν τα δεδομένα μας βρίσκονται μόνο σε μια ενιαία βάση δεδομένων, εκτίθενται σε πολλαπλά πιθανά σημεία αποτυχίας, όπως server crash, διακοπές υπηρεσίας ή ακόμα και κάποιο σφάλμα υλικού. Οποιοδήποτε από αυτά τα συμβάντα θα καθιστούσε σχεδόν αδύνατη την πρόσβαση στα δεδομένα μας.

Το replication μας επιτρέπει να παρακάμψουμε αυτές τις ευπάθειες αναπτύσσοντας πολλούς διακομιστές για ανάκτηση από καταστροφή και δημιουργία αντιγράφων ασφαλείας. Η οριζόντια κλιμάκωση σε πολλούς διακομιστές που φιλοξενούν τα ίδια δεδομένα (ή θραύσματα των ίδιων δεδομένων) σημαίνει πολύ αυξημένη διαθεσιμότητα και σταθερότητα δεδομένων. Φυσικά, η αναπαραγωγή βοηθά επίσης και στο load balancing. Όταν πολλοί χρήστες έχουν πρόσβαση στα ίδια δεδομένα, το φορτίο μπορεί να κατανεμηθεί ομοιόμορφα στους διακομιστές.

Στη MongoDB, χρησιμοποιούνται σύνολα αντιγράφων για αυτόν τον σκοπό. Ένας κύριος διακομιστής ή κόμβος δέχεται όλες τις λειτουργίες εγγραφής και εφαρμόζει αυτές τις ίδιες λειτουργίες σε δευτερεύοντες διακομιστές, αναπαράγοντας τα δεδομένα. Εάν ο κύριος διακομιστής αντιμετωπίσει ποτέ μια κρίσιμη αποτυχία, οποιοσδήποτε από τους δευτερεύοντες διακομιστές μπορεί να επιλεγεί για να γίνει ο νέος κύριος κόμβος. Και αν ο πρώην πρωτεύων κόμβος επανέλθει στο διαδίκτυο, το κάνει ως δευτερεύων διακομιστής για τον νέο πρωτεύοντα κόμβο.

Sharding

Όταν ασχολούμαστε με ιδιαίτερα μεγάλα σύνολα δεδομένων, ο διαμοιρασμός —η διαδικασία διαχωρισμού μεγαλύτερων συνόλων δεδομένων σε πολλαπλές κατανεμημένες συλλογές ή «θραύσματα»— βοηθά τη βάση δεδομένων να διανέμει και να εκτελεί καλύτερα ό,τι διαφορετικά θα μπορούσε να είναι προβληματικό και δυσκίνητο. Χωρίς διαμοιρασμό, η κλιμάκωση μιας αναπτυσσόμενης εφαρμογής Ιστού με εκατομμύρια καθημερινούς χρήστες είναι σχεδόν αδύνατη.

Όπως το replication μέσω συνόλων αναπαραγωγής, ο διαμοιρασμός στη MongoDB επιτρέπει πολύ μεγαλύτερη οριζόντια κλιμάκωση. Οριζόντια κλιμάκωση σημαίνει ότι κάθε θραύσμα σε κάθε σύμπλεγμα (cluster) φιλοξενεί ένα τμήμα του εν λόγω συνόλου δεδομένων, λειτουργώντας ουσιαστικά ως ξεχωριστή βάση δεδομένων. Η συλλογή των κατανεμημένων θραυσμάτων διακομιστή σχηματίζει μια ενιαία, περιεκτική βάση δεδομένων πολύ πιο κατάλληλη για να χειριστεί τις ανάγκες μιας δημοφιλής, αυξανόμενης εφαρμογής με μηδενικό χρόνο διακοπής λειτουργίας.

Όλες οι λειτουργίες σε περιβάλλον sharding γίνονται μέσω μιας ελαφριάς διαδικασίας που ονομάζεται mongos. Τα Mongos μπορούν να κατευθύνουν ερωτήματα στο σωστό θραύσμα με βάση το κλειδί θραύσματος (shard key). Φυσικά, ο σωστός θρυμματισμός συμβάλλει επίσης σημαντικά στην καλύτερη εξισορρόπηση του φορτίου (load balancing).

Loadbalancing

Η βέλτιστη εξισορρόπηση φορτίου (load balancing) παραμένει μια από τα σημαντικότερες λειτουργίες της διαχείρισης βάσεων δεδομένων μεγάλης κλίμακας για αναπτυσσόμενες εταιρικές εφαρμογές. Η σωστή διανομή εκατομμυρίων αιτημάτων πελατών σε εκατοντάδες ή χιλιάδες διακομιστές μπορεί να οδηγήσει σε αισθητή (και εκτιμώμενη) διαφορά στην απόδοση.

Μέσω ορισμένων όπως το replication και το sharding, η MongoDB υποστηρίζει εξισορρόπηση φορτίου μεγάλης κλίμακας. Η πλατφόρμα μπορεί να χειριστεί πολλαπλά ταυτόχρονα αιτήματα ανάγνωσης και εγγραφής για τα ίδια δεδομένα με τα καλύτερα πρωτόκολλα ελέγχου συγχρονισμού και κλειδώματος της κατηγορίας που διασφαλίζουν τη συνέπεια των δεδομένων. Δεν χρειάζεται να προσθέσετε έναν εξωτερικό εξισορροπητή φορτίου. Η MongoDB διασφαλίζει ότι κάθε χρήστης έχει συνεχόμενη ροή και εμπειρία ποιότητας με τα δεδομένα που χρειάζεται να έχει πρόσβαση

Βασικές ορολογίες της MongoDB

- Collections: Οι «συλλογές» στο Mongo είναι ισοδύναμες με πίνακες σε σχεσιακές βάσεις δεδομένων. Μπορούν να κρατήσουν πολλά JSON documents.
- Documents: Τα Documents είναι ισοδύναμα με εγγραφές ή σειρές δεδομένων σε SQL. Ενώ μια σειρά SQL μπορεί να αναφέρει δεδομένα σε άλλους πίνακες, τα documents στη Mongo συνήθως τα συνδυάζουν σε ένα document
- Fields: Τα «πεδία» ή τα χαρακτηριστικά είναι παρόμοια με τις στήλες σε έναν πίνακα SQL.
- Schema: Ενώ η Mongo είναι schema-less, η SQL ορίζει ένα σχήμα μέσω του ορισμού του πίνακα. Ένα «σχήμα» της Mongoose είναι μια δομή δεδομένων document που επιβάλλεται μέσω του επιπέδου εφαρμογής.
- Models: Τα «μοντέλα» λαμβάνουν ένα schema και δημιουργούν ένα instance ενός document ισοδύναμο με εγγραφές σε μια σχεσιακή βάση δεδομένων.

Version	Release Date
1.0	08-2009
1.2	12-2009
1.4	03-2010
1.6	08-2010
1.8	03-2011
2.0	09-2011
2.2	08-2012
2.4	03-2013
2.6	04-2014
3.0	03-2015
3.2	12-2015
3.4	11-2016
3.6	11-2017
4.0	06-2018
4.2	08-2019
4.4	07-2020
4.4.5	04-2021
4.4.6	05-2021
5.0	06-2021

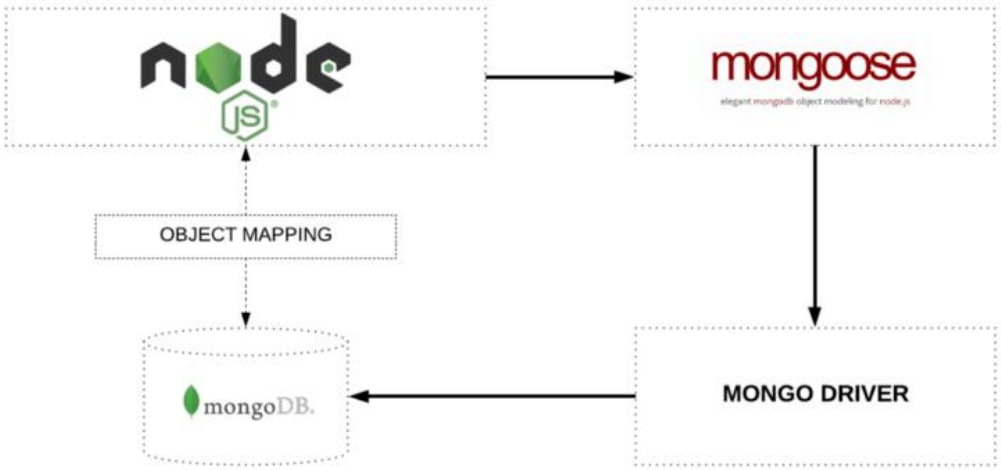
Πίνακας 1.3 Ιστορική αναδρομή στις εκδόσεις της MongoDB

1.5 Βιβλιοθήκες Ανάπτυξης

Η βιβλιοθήκη που αξίζει να αναφερθεί στο συγκεκριμένο κεφάλαιο είναι η Mongoose. Πρόκειται για μία βιβλιοθήκη Object Data Modeling (ODM) για MongoDB και Node.js. Διαχειρίζεται τις σχέσεις μεταξύ δεδομένων, παρέχει schema validation και χρησιμοποιείται για τη μετάφραση μεταξύ αντικειμένων στον κώδικα και την αναπαράσταση αυτών των αντικειμένων στη MongoDB. Επιπλέον περιλαμβάνει built-in type casting, validation, query building, business logic hooks κτλ. Το πρόβλημα που στοχεύει να λύσει η Mongoose είναι να επιτρέπει στους προγραμματιστές να επιβάλλουν ένα συγκεκριμένο σχήμα (schema) στο επίπεδο εφαρμογής. Όλα τα παραπάνω επιτυγχάνονται μέσω του Mongoose module που είναι ένα από τα πιο ισχυρά external modules της Node.js.

Πλεονεκτήματα Mongoose module

1. Η επικύρωση των collections της βάσης δεδομένων MongoDB μπορεί να γίνει εύκολα.
2. Μία προκαθορισμένη δομή μπορεί να εφαρμοστεί στη συλλογή.
3. Μπορούν να εφαρμοστούν περιορισμοί στα documents των collections που χρησιμοποιούν Mongoose.
4. Το Mongoose module είναι χτισμένο στην κορυφή του MongoDB driverκαι παρέχει εύκολη διαχείριση και ορισμό query.



Σχήμα 1.5 Object Mapping μεταξύ Node και MongoDB διαχειριζόμενα μέσω Mongoose

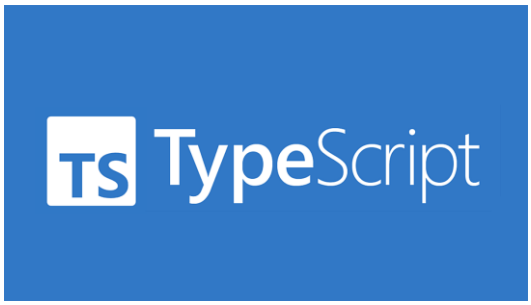
1.6 Γλώσσες προγραμματισμού

Typescript

Η JavaScript μας συστήθηκε ως μία γλώσσα για την πλευρά του πελάτη. Η ανάπτυξη της Node.js έχει χαρακτηρίσει την JavaScript ως μια αναδυόμενη τεχνολογία από την πλευρά του διακομιστή. Ωστόσο, καθώς ο κώδικας JavaScript μεγαλώνει, τείνει να γίνεται πιο ακατάστατος, καθιστώντας δύσκολη τη συντήρηση και την επαναχρησιμοποίηση του κώδικα. Επιπλέον, η αποτυχία του να ενστερνιστεί τις δυνατότητες του Object Orientation, του ισχυρού ελέγχου τύπου (strong type checking) και των

ελέγχων σφαλμάτων στο χρόνο μεταγλώττισης εμποδίζει την JavaScript να πετύχει σε εταιρικό επίπεδο ως μια ολοκληρωμένη τεχνολογία διακομιστή. HTypeScript παρουσιάστηκε για να γεφυρώσει αυτό το χάσμα.

Η TypeScript είναι μία ισχυρά typed, αντικειμενοστρεφής και μεταγλωττισμένη γλώσσα. Σχεδιάστηκε από τον Anders Hejlsberg (σχεδιαστή της C#) στη Microsoft. Η TypeScript είναι ταυτόχρονα μια γλώσσα και ένα σύνολο εργαλείων. Πρόκειται για ένα δακτυλογραφημένο υπερσύνολο JavaScript που έχει μεταγλωττιστεί σε JavaScript. Με άλλα λόγια, η TypeScript είναι JavaScript συν ορισμένες πρόσθετες δυνατότητες.



Χαρακτηριστικά της TypeScript

- Ξεκινά με JavaScript και τελειώνει με JavaScript. Υιοθετεί τα βασικά δομικά στοιχεία του προγράμματός σας από JavaScript. Ως εκ τούτου, χρειάζεται μόνο να γνωρίζει κάποιος JavaScript για να χρησιμοποιήσει την TypeScript. Όλος ο κώδικας TypeScript μετατρέπεται στο ισοδύναμο JavaScript για σκοπούς εκτέλεσης.
- Υποστηρίζει άλλες βιβλιοθήκες JS. Η μεταγλωττισμένη TypeScript μπορεί να καταναλωθεί από οποιονδήποτε κώδικα JavaScript. Η JavaScript που δημιουργείται από TypeScript μπορεί να επαναχρησιμοποιήσει όλα τα υπάρχοντα frameworks, εργαλεία και βιβλιοθήκες JavaScript.
- Η JavaScript είναι TypeScript. Αυτό σημαίνει ότι κάθε έγκυρο αρχείο .js μπορεί να μετονομαστεί σε .ts και να μεταγλωττιστεί με άλλα αρχεία TypeScript.
- Είναι φορητή. Η TypeScript είναι φορητή σε προγράμματα περιήγησης, συσκευές και λειτουργικά συστήματα. Μπορεί να εκτελεστεί σε οποιοδήποτε περιβάλλον στο οποίο εκτελείται η JavaScript. Σε αντίθεση με τους ομολόγους του, το TypeScript δεν χρειάζεται αποκλειστικό VM (virtual machine) ή συγκεκριμένο περιβάλλον χρόνου εκτέλεσης για να εκτελεστεί.

Γιατί να χρησιμοποιήσει κάποιος Typescript

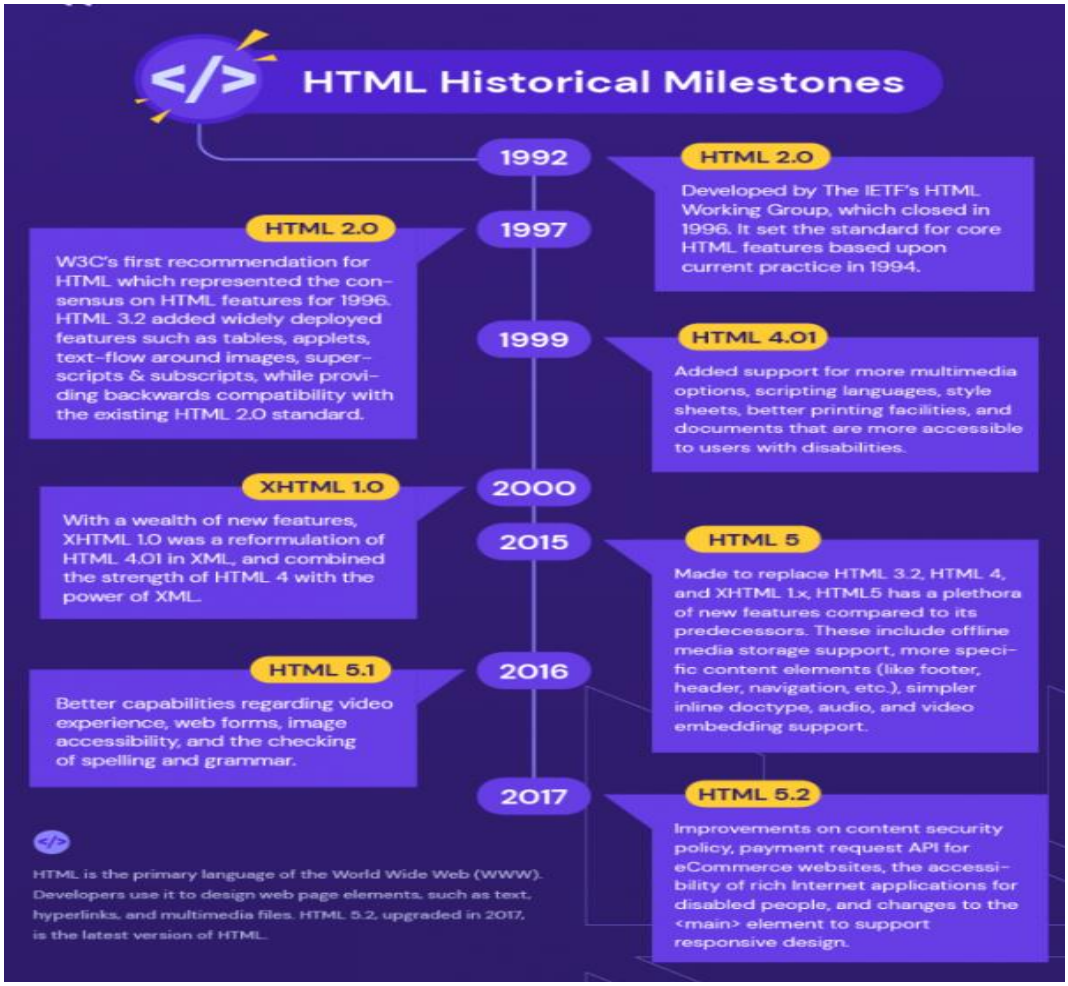
- Η JavaScript είναι μια ερμηνευμένη γλώσσα. Ως εκ τούτου, πρέπει να εκτελεστεί πρώτα για να ελέγξουμε ότι είναι έγκυρη. Σημαίνει ότι γράφουμε όλους τους κώδικα μόνο και μόνο για να μην εμφανιστεί έξοδος, σε περίπτωση που υπάρχει σφάλμα. Ως εκ τούτου, πρέπει να σπαταλήσουμε κάποιο χρόνο ώρες προσπαθώντας να βρούμε τα σφάλματα στον κώδικα. Ο transpiler της TypeScript παρέχει τη δυνατότητα ελέγχου σφαλμάτων. Η TypeScript θα μεταγλωττίσει τον κώδικα και θα δημιουργήσει σφάλματα μεταγλώττισης, εάν βρει κάποιο είδος συντακτικών σφαλμάτων. Αυτό βοηθά στην επισήμανση σφαλμάτων πριν από την εκτέλεση του σεναρίου.
- Ισχυρό static typing - Η JavaScript δεν διαθέτει κάτι αντίστοιχο. Η TypeScript συνοδεύεται από ένα προαιρετικό σύστημα static typing και type inference μέσω του TLS (TypeScript Language Service). Ο τύπος μιας μεταβλητής, που δηλώνεται χωρίς τύπο, μπορεί να συναχθεί από το TLS με βάση την τιμή της.

- Υποστηρίζει type definitions για υπάρχουσες βιβλιοθήκες JavaScript. Το αρχείο Definition TypeScript (με επέκταση .d.ts) παρέχεται για εξωτερικές βιβλιοθήκες JavaScript. Ως εκ τούτου, ο κώδικας TypeScript μπορεί να περιέχει αυτές τις βιβλιοθήκες.
- Υποστηρίζει αντικειμενοστρεφείς έννοιες προγραμματισμού όπως κλάσεις, διεπαφές (interfaces), κληρονομικότητα κ.λπ.

HTML

Πρόκειται για μία γλώσσα που επιτρέπει στους χρήστες του Ιστού να δημιουργούν και να δομούν ενότητες, παραγράφους και συνδέσμους χρησιμοποιώντας στοιχεία, ετικέτες και χαρακτηριστικά. Διαθέτει πολλές περιπτώσεις χρήσης με τις πιο χαρακτηριστικές να είναι οι παρακάτω.

1. Ανάπτυξη εφαρμογών web: Οι προγραμματιστές χρησιμοποιούν κώδικα HTML για να σχεδιάσουν πώς ένα πρόγραμμα περιήγησης εμφανίζει στοιχεία ιστοσελίδας, όπως κείμενο, υπερσυνδέσμους και αρχεία πολυμέσων.
2. Πλοήγηση στο Διαδίκτυο: Οι χρήστες μπορούν εύκολα να πλοηγηθούν και να εισάγουν συνδέσμους μεταξύ σχετικών σελίδων και ιστότοπων, καθώς η HTML χρησιμοποιείται σε μεγάλο βαθμό για την ενσωμάτωση υπερσυνδέσμων.
3. Documentation: Η HTML καθιστά δυνατή την οργάνωση και τη μορφοποίηση εγγράφων.



Σχήμα 1.6 Ιστορική αναδρομή στις εκδόσεις HTML

Ένας τυπικός ιστότοπος περιλαμβάνει πολλές διαφορετικές σελίδες HTML. Για παράδειγμα, μια αρχική σελίδα, μια σελίδα σχετική με το περιεχόμενο με και μια σελίδα επικοινωνίας θα έχουν όλα ξεχωριστά αρχεία HTML.

Τα έγγραφα HTML είναι αρχεία που τελειώνουν με επέκταση .html ή .htm. Ένα πρόγραμμα περιήγησης ιστού διαβάζει το αρχείο HTML και αποδίδει το περιεχόμενό του έτσι ώστε οι χρήστες του Διαδικτύου να μπορούν να το δουν.

Όλες οι σελίδες HTML έχουν μια σειρά από στοιχεία (elements) HTML, που αποτελούνται από ένα σύνολο ετικετών (tags) και χαρακτηριστικών (attributes). Τα στοιχεία HTML είναι τα δομικά στοιχεία μιας ιστοσελίδας. Μια ετικέτα λέει στο πρόγραμμα περιήγησης ιστού πού ξεκινά και πού τελειώνει ένα στοιχείο, ενώ ένα χαρακτηριστικό περιγράφει τα χαρακτηριστικά ενός στοιχείου.

1.7 MongoDB vs MYSQL

Αρχικά να αναφέρουμε ορισμένα εισαγωγικά πράγματα για την MYSQL (για την MongoDb έχει γίνει αναφορά στην ενότητα 1.3).

Η MySQL είναι ένα δημοφιλές, δωρεάν στη χρήση και ανοιχτού κώδικα σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων (RDBMS) που αναπτύχθηκε από την Oracle. Όπως και με άλλα σχεσιακά συστήματα, η MySQL αποθηκεύει δεδομένα χρησιμοποιώντας πίνακες και σειρές, επιβάλλει ακεραιότητα αναφοράς και χρησιμοποιεί δομημένη γλώσσα ερωτημάτων (SQL) για πρόσβαση σε δεδομένα. Όταν οι χρήστες πρέπει να ανακτήσουν δεδομένα από μια βάση δεδομένων MySQL, πρέπει να δημιουργήσουν ένα ερώτημα SQL που ενώνει πολλούς πίνακες μεταξύ τους για να δημιουργήσει την προβολή των δεδομένων που χρειάζονται.

Τα σχήματα βάσεων (schemas) δεδομένων και τα μοντέλα δεδομένων (data models) πρέπει να καθοριστούν εκ των προτέρων και τα δεδομένα πρέπει να ταιριάζουν με αυτό το σχήμα για να αποθηκευτούν στη βάση δεδομένων. Αυτή η άκαμπτη προσέγγιση για την αποθήκευση δεδομένων προσφέρει κάποιο βαθμό ασφάλειας, αλλά το ανταλλάσσει με ευελιξία. Εάν πρέπει να αποθηκευτεί ένας νέος τύπος ή μορφή δεδομένων στη βάση δεδομένων, πρέπει να πραγματοποιηθεί μετεγκατάσταση (migrate) σχήματος, η οποία μπορεί να γίνει πολύπλοκη και ακριβή όσο μεγαλώνει το μέγεθος της βάσης δεδομένων. Παρακάτω θα αναφερθούμε σε ορισμένες διαφορές μεταξύ των βάσεων.

User-Friendliness

Η MongoDB είναι μια ελκυστική επιλογή για προγραμματιστές. Η φιλοσοφία της αποθήκευσης δεδομένων είναι απλή και άμεσα κατανοητή σε οποιονδήποτε έχει εμπειρία προγραμματισμού. Αποθηκεύει δεδομένα σε συλλογές (collections) χωρίς επιβεβλημένο σχήμα (schema). Αυτή η ευέλικτη προσέγγιση για την αποθήκευση δεδομένων την καθιστά ιδιαίτερα κατάλληλη για προγραμματιστές που μπορεί να μην είναι ειδικοί στη βάση δεδομένων, αλλά θέλουν να χρησιμοποιήσουν μια βάση δεδομένων για να υποστηρίξουν την ανάπτυξη των εφαρμογών τους. Σε σύγκριση με τη MySQL, αυτή η ευελιξία είναι ένα σημαντικό πλεονέκτημα: Για να αξιοποιήσουμε στο έπακρο μια σχεσιακή βάση δεδομένων, πρέπει πρώτα να κατανοήσουμε τις αρχές της κανονικοποίησης, της ακεραιότητας αναφοράς (referential integrity) και του σχεδιασμού της

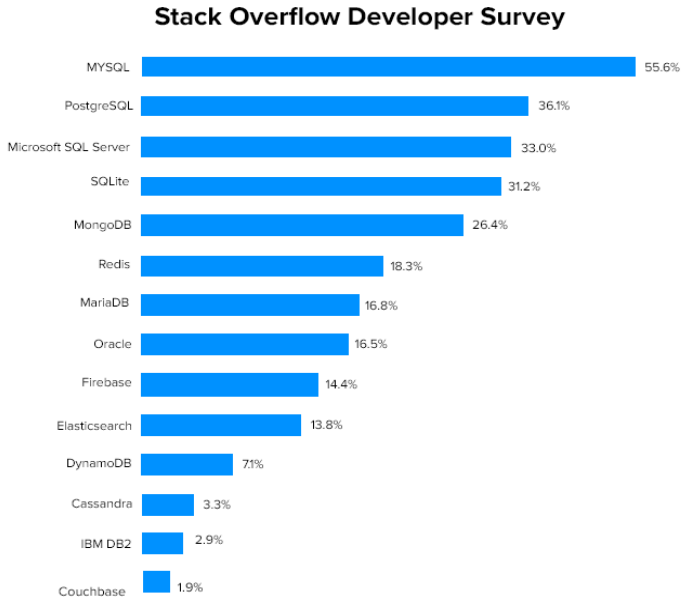
σχεσιακής βάσης δεδομένων. Με τη δυνατότητα αποθήκευσης documents διαφορετικών σχημάτων, συμπεριλαμβανομένων μη δομημένων συνόλων δεδομένων, η MongoDB παρέχει μια ευέλικτη διεπαφή προγραμματιστή για ομάδες που δημιουργούν εφαρμογές που δεν χρειάζονται όλα τα χαρακτηριστικά ασφαλείας που προσφέρουν τα σχεσιακά συστήματα. Ένα συνηθισμένο παράδειγμα μιας τέτοιας εφαρμογής είναι μια εφαρμογή Ιστού που δεν εξαρτάται από δομημένα σχήματα. μπορεί εύκολα να εξυπηρετήσει μη δομημένα, ημι-δομημένα ή δομημένα δεδομένα, όλα από την ίδια MongoDB collection

Η MySQL είναι μια κοινή επιλογή για χρήστες που έχουν μεγάλη εμπειρία στη χρήση παραδοσιακών scripting SQL, σχεδίασης λύσεων για σχεσιακές βάσεις δεδομένων ή που τροποποιούν ή ενημερώνουν υπάρχουσες εφαρμογές που ήδη λειτουργούν με ένα σχεσιακό σύστημα. Οι σχεσιακές βάσεις δεδομένων μπορεί επίσης να είναι καλύτερη επιλογή για εφαρμογές που απαιτούν πολύ περίπλοκες αλλά άκαμπτες δομές δεδομένων και σχήματα βάσεων δεδομένων σε μεγάλο αριθμό πινάκων (π.χ. τραπεζικές εφαρμογές).

Επεκτασιμότητα

Ένα βασικό πλεονέκτημα του σχεδιασμού στη MongoDB είναι ότι η κλίμακα της βάσης δεδομένων είναι εξαιρετικά εύκολη. Όπως αναφέραμε σε προηγούμενη ενότητα (βλέπε 2.3) με την χρήση ενός sharded cluster επιτρέπεται στη MongoDB να κλιμακώνει οριζόντια την απόδοση ανάγνωσης και εγγραφής για να καλύψει εφαρμογές οποιασδήποτε κλίμακας.

Με ένα σύστημα βάσης δεδομένων MySQL, οι επιλογές επεκτασιμότητας είναι πολύ πιο περιορισμένες. Συνήθως, έχουμε δύο επιλογές: κάθετη επεκτασιμότητα ή προσθήκη αντιγράφων ανάγνωσης. Η κατακόρυφη κλιμάκωση περιλαμβάνει την προσθήκη περισσότερων πόρων στον υπάρχοντα διακομιστή βάσης δεδομένων, αλλά αυτό έχει ένα εγγενές ανώτατο όριο. Η αναπαραγωγή ανάγνωσης (read replication) περιλαμβάνει την προσθήκη αντιγράφων μόνο για ανάγνωση της βάσης δεδομένων σε άλλους διακομιστές.



Σχήμα 1.7 Προτιμήσεις προγραμματιστών σε βάσεις δεδομένων σύμφωνα με έρευνα που πραγματοποιήθηκε στο Stack Overflow

Απόδοση

Η αξιολόγηση της απόδοσης δύο τελείως διαφορετικών συστημάτων βάσεων δεδομένων είναι πολύ δύσκολη, καθώς και τα δύο συστήματα διαχείρισης προσεγγίζουν την διαδικασία της αποθήκευσης και ανάκτησης δεδομένων με εντελώς διαφορετικούς τρόπους. Ενώ είναι δυνατό να συγκριθούν απευθείας δύο βάσεις δεδομένων SQL με ένα σύνολο τυπικών σημείων αναφοράς SQL, η επίτευξη του ίδιου μεταξύ μη σχεσιακών και σχεσιακών βάσεων δεδομένων είναι πολύ πιο δύσκολη και υποκειμενική.

Η MySQL είναι βελτιστοποιημένη για joins υψηλής απόδοσης σε πολλούς πίνακες που έχουν γίνει κατάλληλα Indexed. Η MongoDB είναι καλύτερη ως προς την απόδοση των writes και διαθέτει ένα συγκεκριμένο API insertMany() για γρήγορη εισαγωγή δεδομένων, δίνοντας προτεραιότητα στην ταχύτητα έναντι της ασφάλειας συναλλαγών, όπου τα δεδομένα MySQL πρέπει να εισάγονται σειρά προς σειρά. Μπορούμε να δούμε ότι η MySQL είναι πιο γρήγορη στην επιλογή μεγάλου αριθμού εγγραφών, ενώ η MongoDB είναι σημαντικά ταχύτερη ως προς την εισαγωγή ή ενημέρωση μεγάλου αριθμού εγγραφών.

Ευελιξία

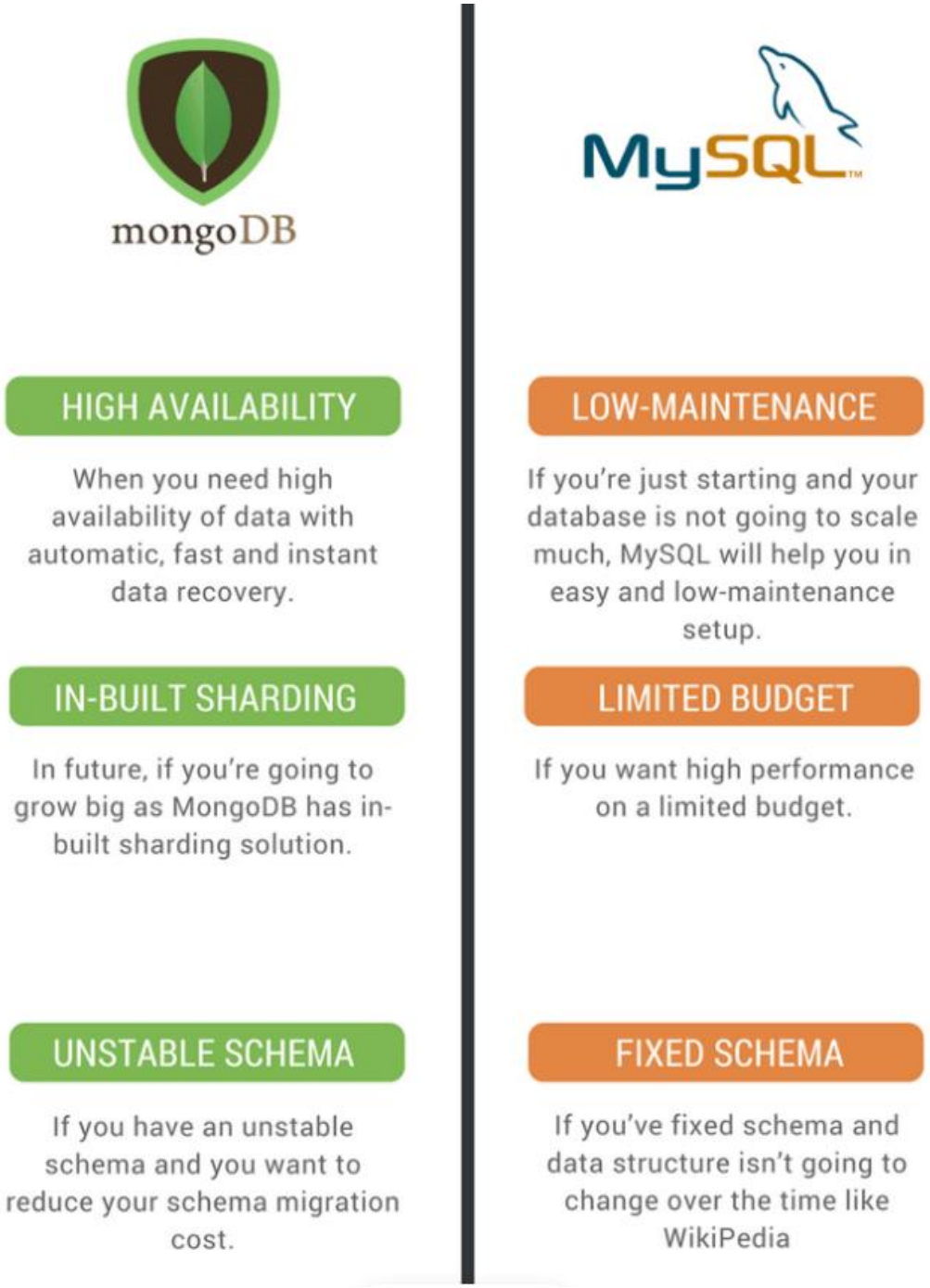
Το συγκεκριμένο χαρακτηριστικό μπορούμε να αναφέρουμε με σιγουριά ότι πηγαίνει στη MongoDB. Ο σχεδιασμός χωρίς σχήμα (schema-less) των documents της MongoDB καθιστά εξαιρετικά εύκολη τη δημιουργία και τη βελτίωση εφαρμογών με την πάροδο του χρόνου, χωρίς να χρειάζεται να εκτελούμε πολύπλοκες και δαπανηρές διαδικασίες μετεγκατάστασης (migration) σχήματος όπως θα κάνατε με μια σχεσιακή βάση δεδομένων.

Με την MongoDB, υπάρχουν πιο δυναμικές επιλογές για την ενημέρωση του σχήματος ενός collection, όπως η δημιουργία νέων πεδίων. Το όφελος είναι ιδιαίτερα σημαντικό καθώς οι βάσεις δεδομένων μεγαλώνουν σε μέγεθος. Αντίθετα, οι μεγαλύτερες βάσεις δεδομένων MySQL είναι πιο αργές στη μετεγκατάσταση (migration) σχημάτων και αποθηκευμένων διαδικασιών.

Ασφάλεια

Η MongoDB αξιοποιεί το δημοφιλές μοντέλο ελέγχου πρόσβασης (role-based access control model) που βασίζεται σε ρόλους με ένα ευέλικτο σύνολο δικαιωμάτων. Οι χρήστες εκχωρούνται σε έναν ρόλο και αυτός ο ρόλος τους παρέχει συγκεκριμένα δικαιώματα για σύνολα δεδομένων και λειτουργίες βάσης δεδομένων. Όλη η επικοινωνία είναι κρυπτογραφημένη με TLS.

Η MySQL υποστηρίζει τις ίδιες δυνατότητες κρυπτογράφησης με την MongoDB. Το μοντέλο ελέγχου ταυτότητας είναι επίσης παρόμοιο. Στους χρήστες μπορούν να παραχωρηθούν ρόλοι αλλά και προνόμια, δίνοντάς τους δικαιώματα για συγκεκριμένες λειτουργίες βάσης δεδομένων και έναντι συγκεκριμένων συνόλων δεδομένων.



Σχήμα 1.8 Ενδεικτικοί λόγοι χρήσης της κάθε βάσης

1.8 Pusher

Το Pusher λειτουργεί ως επίπεδο επικοινωνίας σε πραγματικό χρόνο μεταξύ του διακομιστή και του πελάτη. Διατηρεί μόνιμες συνδέσεις στον πελάτη χρησιμοποιώντας WebSockets, καθώς και όταν προστίθενται νέα δεδομένα στον διακομιστή σας. Εάν ένας διακομιστής θέλει να προωθήσει νέα δεδομένα στους πελάτες, μπορεί να το κάνει αμέσως χρησιμοποιώντας το Pusher. Είναι εξαιρετικά ευέλικτο, επεκτάσιμο και εύκολο να ενσωματωθεί. Στο πλαίσιο της παροχής δεδομένων σε πραγματικό χρόνο, υπάρχουν διαθέσιμες και άλλες υπηρεσίες φιλοξενίας. Εξαρτάται από την περίπτωση χρήσης του τι ακριβώς χρειάζεται κάποιος, όπως εάν χρειάζεται να μεταδώσει δεδομένα σε όλους τους χρήστες ή κάτι πιο περίπλοκο με συγκεκριμένες ομάδες-στόχους. Στην περίπτωση της

εφαρμογής της πτυχιακής αυτής το Pusher ήταν κατάλληλο, καθώς είναι εύκολο στη χρήση του και εύκολα επεκτάσιμο.

Το Pusher είναι κατάλληλο για λειτουργίες επικοινωνίας και συνεργασίας χρησιμοποιώντας WebSockets. Η βασική διαφορά με το Pusher είναι ότι πρόκειται μια φιλοξενούμενη υπηρεσία/API. Χρειάζεται λιγότερη δουλειά για να ξεκινήσουμε, σε σύγκριση με άλλα που πρέπει το η ανάπτυξη να γίνει από την πλευρά μας. Μόλις κάνουμε την αρχική ρύθμιση έχουμε ουσιαστικά τελειώσει χωρίς να απαιτείται μελλοντική εργασία.

Περιπτώσεις χρήσης

1. Ειδοποιήσεις : Το Pusher μπορεί να ενημερώσει τους χρήστες εάν υπάρχει οποιαδήποτε σχετική αλλαγή. Οι ειδοποιήσεις αυτές εννοείται δεν εμφανίζονται στο UI αλλά μέσα στην εφαρμογή μας.
2. Activity streams: Όταν κάτι αλλάζει στον διακομιστή ή κάποιος δημοσιεύει οτιδήποτε σε όλα τα κανάλια.
3. Εμφάνιση live data: Το Pusher μας επιτρέπει να μεταδίδουμε συνεχώς μεταβαλλόμενα δεδομένα όταν χρειάζεται.
4. Chat : Μπορούμε να χρησιμοποιήσετε το Pusher για peer to peer επικοινωνία.

Τύποι καναλιών (channels)

- Public κανάλια: Αυτά τα κανάλια είναι δημόσια, επομένως όποιος γνωρίζει το όνομα του καναλιού μπορεί να εγγραφεί στο κανάλι και να αρχίσει να λαμβάνει μηνύματα από το κανάλι. Τα δημόσια κανάλια χρησιμοποιούνται συνήθως για τη μετάδοση γενικών/δημόσιων πληροφοριών, οι οποίες δεν περιέχουν ασφαλείς πληροφορίες ή δεδομένα ειδικά για τον χρήστη.
- Private κανάλια: Αυτά τα κανάλια διαθέτουν μηχανισμό ελέγχου πρόσβασης που επιτρέπει στον διακομιστή να ελέγχει ποιος μπορεί να εγγραφεί στο κανάλι και να λαμβάνει δεδομένα από το κανάλι. Όλα τα ιδιωτικά κανάλια θα πρέπει να έχουν ένα ιδιωτικό πρόθεμα στο όνομα. Χρησιμοποιούνται συνήθως όταν ο διακομιστής πρέπει να γνωρίζει ποιος μπορεί να εγγραφεί στο κανάλι και να επικυρώσει τους συνδρομητές.
- Presence κανάλια: Είναι μια επέκταση των private καναλιών. Εκτός από τις ιδιότητες που έχουν τα ιδιωτικά κανάλια, επιτρέπει στον διακομιστή να «εγγράφει» τις πληροφορίες των χρηστών σχετικά με τη συνδρομή/εγγραφή τους στο κανάλι. Επιτρέπει επίσης σε άλλα μέλη να αναγνωρίσουν ποιος είναι συνδεδεμένος.

1.9 Github

Πρόκειται για μία εταιρεία παγκόσμιου βελγικού που βοηθάει σημαντικά την φιλοξενία και την ανάπτυξη κώδικα. Τα παραπάνω επιτυγχάνονται με την χρήση του Git. Ενδείκνυται για μεγάλα project αλλά και μικρότερης κλίμακας καθώς σου παρέχει πρόσβαση στον κώδικα σου από οποιοδήποτε σημείο και να βρίσκεσαι και το πιο σημαντικό είναι ότι σου δίνει την δυνατότητα rollback σε περίπτωση κάποιου λάθους.

1.10 Επίλογος

Είναι πολύ σημαντικό πριν την δημιουργία μια εφαρμογής, να έχουμε καλά δομημένο τον τρόπο με τον οποίο θα υλοποιηθεί, καθώς και τις ανάγκες που μπορεί να προκύψουν μελλοντικά. Για την συγκεκριμένη εφαρμογή θεώρησα ως καλύτερη επιλογή την χρήση της MongoDB, λόγω της φύσης ενός Media Application το οποίο διαρκώς χρειάζεται προσθήκες, είτε σε feautures είτε σε περιεχόμενα. Στη συγκεκριμένη περίπτωση καθώς χρησιμοποιείται για προβολή ταινιών, θα υπάρχει η ανάγκη με τον καιρό να προστίθενται όλο και περισσότερες με ότι συνέπειες έχει αυτό σχετικά με την μεταβολή της βάσης. Τέλος, πριν κλείσει το κεφάλαιο αυτό, θα χρειαστεί να αναφέρω ότι η οργάνωση όλων των παραπάνω (framework, βιβλιοθηκών κτλ) έγινε μέσω του Visual Studio Code που είναι ένας editor βελτιστοποιημένος για την δημιουργία και το debugging των σύγχρονων web εφαρμογών.

Κεφάλαιο 2ο: Ανάλυση δικτυακών εφαρμογών πολυμέσων, streaming και δίκτυα διανομής περιεχομένου (CDN)

2.1 Εισαγωγή

Στο συγκεκριμένο κεφάλαιο θα γίνει μια αναλυτική περιγραφή των πιο διαδεδομένων πρωτοκόλλων που σχετίζονται με το streaming καθώς και για την χρήση και την λειτουργία των CDN στις μέρες μας. Επίσης θα γίνει αναφορά για τις δικτυακές εφαρμογές πολυμέσων αλλά και για τις ιδιότητες τους.

2.2 Δικτυακές εφαρμογές πολυμέσων

Ορίζουμε μία δικτυακή εφαρμογή πολυμέσων ως οποιαδήποτε δικτυακή εφαρμογή που χρησιμοποιεί ήχο και βίντεο. Παρακάτω θα αναλυθούν ορισμένες ιδιότητες του βίντεο και του ήχου.

2.2.1 Ιδιότητες Βίντεο

Ίσως το πλέον περίοπτο χαρακτηριστικό του βίντεο είναι ο υψηλός ρυθμός bit (high bit rate). Τα βίντεο που διανέμονται στο διαδίκτυο ποικίλουν από βίντεο διάσκεψης χαμηλής ποιότητας 100 Kbps, μέχρι βίντεο 3 Mbps για συνεχή ροή ταινιών υψηλής ευκρίνειας. Ένα άλλο σημαντικό χαρακτηριστικό του βίντεο είναι ότι μπορεί να συμπιέζεται και έτσι να διαπραγματεύεστε ανάμεσα στην ποιότητα του βίντεο και τον ρυθμό bit. Ένα βίντεο είναι μία ακολουθία εικόνων, που τυπικά εμφανίζονται σε σταθερό ρυθμό, π.χ. σε 24 ή 30 εικόνες ανά δευτερόλεπτο. Μία ασυμπίεστη, ψηφιακά κωδικοποιημένη εικόνα αποτελείται από έναν πίνακα pixels, όπου κάθε pixel κωδικοποιείται σε έναν αριθμό bit, για να παρασταθεί η φωτεινότητα και το χρώμα του. Υπάρχουν δύο τύποι πλεονασμού στα βίντεο, με και τους δύο να μπορούν να χρησιμοποιηθούν για συμπίεση βίντεο (video compression).

Ο χωρικός πλεονασμός είναι ο πλεονασμός μέσα σε μία δεδομένη εικόνα. Διορατικά, μια εικόνα που αποτελείται από κυρίως λευκό χώρο έχει υψηλό βαθμό πλεονασμού και μπορεί να συμπιεσθεί αποδοτικά, χωρίς να υπάρχει σημαντική θυσία στην ποιότητα της εικόνας. Ο χρονικός πλεονασμός αφορά την επανάληψη μίας εικόνας στην επόμενη εικόνα. Εάν για παράδειγμα, μία εικόνα και η επόμενη της είναι ακριβώς ίδιες, δεν υπάρχει λόγος να κωδικοποιήσουμε εκ νέου την επόμενη εικόνα. Είναι αντιθέτως πιο αποδοτικό να δηλώσουμε απλώς κατά την διάρκεια της κωδικοποίησης ότι η επόμενη εικόνα είναι ακριβώς ίδια. Οι σημερινοί εμπορικοί αλγόριθμοι συμπίεσης μπορούν να συμπιέζουν ένα βίντεο σε οποιονδήποτε επιθυμητό ρυθμό bit. Φυσικά, όσο υψηλότερος είναι ο ρυθμός bit, τόσο καλύτερη θα είναι η ποιότητα της εικόνας και τόσο καλύτερη θα είναι η εμπειρία της προβολής της από τον χρήστη.

Μπορούμε επίσης να χρησιμοποιήσουμε συμπίεση για να δημιουργήσουμε πολλαπλές εκδόσεις του ίδιου βίντεο, κάθε μίας σε διαφορετικό επίπεδο ποιότητας. Για παράδειγμα μπορούμε να χρησιμοποιήσουμε συμπίεση για να δημιουργήσουμε, π.χ. τρεις εκδόσεις του ίδιου βίντεο με

διαφορετικούς ρυθμούς (300 kbps, 1 mbps κτλ). Ο τελικός χρήστης μπορεί κατόπιν να αποφασίσει ποια έκδοση θέλει να δει ως μία συνάρτηση του τρέχοντος διαθέσιμου εύρους ζώνης του. Χρήστης με σύνδεση υψηλής ταχύτητας στο διαδίκτυο μπορεί να επιλέξει την μεγαλύτερη δυνατή έκδοση, ενώ ένας χρήστης με περιορισμένη σύνδεση μπορεί να επιλέξει πιο χαμηλές εκδόσεις. Παρόμοια, το βίντεο σε μία εφαρμογή εικονοδιάσκεψης μπορεί να συμπιεσθεί "στα γρήγορα" για να παρέχει την καλύτερη ποιότητα βίντεο, δεδομένου του διαθέσιμου εύρους ζώνης από άκρο σε άκρο, ανάμεσα σε συνομιλούντες χρήστες.

2.2.2 Ιδιότητες Ήχου

Ο ψηφιακός ήχος (που περιλαμβάνει ψηφιακή φωνή και μουσική) έχει σημαντικότερα μικρότερες απαιτήσεις εύρους ζώνης από το βίντεο. Μία δημοφιλής τεχνική συμπίεσης για τον ήχο και μία από τις πιο γνωστές είναι η MPEG 1 layer3 ή MP3, όπως είναι ευρύτερα γνωστή. Οι κωδικοποιητές MP3 μπορούν να συμπιέσουν σε πολλούς διαφορετικούς ρυθμούς. Ο ρυθμός 128 Kbps είναι ο συνηθέστερος ρυθμός κωδικοποίησης και παράγει πολύ μικρή υποβάθμιση του ήχου. Ένα σχετικό πρότυπο είναι ο Advanced Audio Coding (AAC), το οποίο έχει γίνει δημοφιλές από την Apple. Όπως και με το βίντεο, μπορούν να παραχθούν πολλαπλές εκδόσεις ενός ρεύματος ήχου, με την κάθε μία σε διαφορετικό ρυθμό. Αν και οι ρυθμοί bit του ήχου είναι γενικά πολύ μικρότεροι από τους ρυθμούς bit του βίντεο, οι χρήστες σε γενικό πλαίσιο είναι περισσότερο ευαίσθητοι σε διακοπές ήχου, παρά σε διακοπές βίντεο. Εάν, από καιρό σε καιρό, χάνεται το σήμερα το βίντεο για λίγα δευτερόλεπτα, μία εικονοδιάσκεψη μπορεί πιθανώς να συνεχίσει χωρίς ιδιαίτερη δυσφορία από τους χρήστες της. Εάν όμως, χάνεται συχνά το ηχητικό σήμα, οι χρήστες ίσως να τερματίσουν την σύνοδο.

2.3 Κατηγορίες πολυμεσικών δικτυακών εφαρμογών

Το διαδίκτυο υποστηρίζει μεγάλη ποικιλία χρήσιμων και συναρπαστικών πολυμεσικών εφαμοργών. Αυτές οι πολυμεσικές εφαρμογές μπορούν να ταξινομηθούν σε τρεις κύριες κατηγορίες.

- Εφαρμογές συνεχούς ροής αποθηκευμένου ήχου/βίντεο (streaming stored audio/video)
- Εφαρμογές συνομιλίας ήχου/βίντεο επί IP (conversational voice/video-over-IP)
- Εφαρμογές συνεχούς ροής ζωντανού ήχου/βίντεο (streaming live audio/video)

2.3.1 Συνεχής Ροή Αποθηκευμένου Ήχου και Βίντεο

Το αποθηκευμένου βίντεο συνεχούς ροής συνδυάζει συστατικά ήχου και βίντεο. Ο αποθηκευμένος ήχος συνεχούς ροής είναι πολύ όμοια με το βίντεο συνεχούς ροής, αν και οι ρυθμοί bit είναι τυπικά πολύ μικρότεροι. Σε αυτήν την κατηγορία εφαρμογών, το υποκείμενο μέσο είναι εκ τω προτέρων εγγεγραμμένο βίντεο, όπως μία ταινία, ένα τηλεοπτικό πρόγραμμα, ένα εγγεγραμμένο αθλητικό γεγονός ή ένα εγγεγραμμένο βίντεο από χρήστη. Αυτά τα εκ τω προτέρων εγγεγραμμένα βίντεο τοποθετούνται σε εξυπηρέτες και οι χρήστες στέλνουν αιτήσεις στους εξυπηρέτες για να δουν τα βίντεο κατ΄απαίτηση (on demand). Τα βίντεο συνεχούς ροής έχει τρία διακριτά χαρακτηριστικά.

- Συνεχής ροή (streaming). Σε μία εφαρμογή βίντεο συνεχούς ροής, ο πελάτης τυπικά αρχίζει την αναπαραγωγή του βίντεο λίγα δευτερόλεπτα, αφού αρχίσει να λαμβάνει το αρχείο απ' τον εξυπηρέτη. Αυτό σημαίνει ότι ο πελάτης θα αναπαράγει από μία θέση μέσα στο βίντεο, ενώ την ίδια ώρα θα λαμβάνει επόμενα κομμάτια του βίντεο από τον εξυπηρέτη. Αυτή η τεχνική, γνωστή ως συνεχής ροή (streaming) αποφεύγει το να χρειάζεται να κατεβάσετε όλο το αρχείο του βίντεο (και έτσι να έχετε μεγάλη καθυστέρηση), πριν αρχίσει η αναπαραγωγή.
- Διαδραστικότητα (interactivity). Επειδή το μέσον είναι εγγεγραμμένο εκ των προτέρων, ο χρήστης μπορεί να κάνει παύση, να αλλάξει θέση προς τα εμπρός η προς τα πίσω, να κάνει γρήγορη προώθηση κ.ο.κ., μέσα στο περιεχόμενο του βίντεο. Ο χρόνος από την στιγμή που ο χρήστης κάνει μία τέτοια αίτηση, μέχρι να εκδηλωθεί η ενέργεια στον πελάτη πρέπει να είναι λιγότερος από μερικά δευτερόλεπτα, για αποδεκτό επίπεδο απόκρισης.
- Συνεχής αναπαραγωγή (continuous playout). Μόλις αρχίσει η αναπαραγωγή του βίντεο, αυτή πρέπει να προχωρεί σύμφωνα με τον αρχικό χρονισμό της εγγραφής. Γι' αυτό τα δεδομένα πρέπει να λαμβάνονται απ' τον εξυπηρέτη έγκαιρα ώστε να αναπαραχθούν στο πελάτη, διαφορετικά οι χρήστες θα αντιλαμβάνονται πάγωμα των καρέ του βίντεο (όταν ο πελάτης περιμένει για τα καθυστερημένα καρέ) ή πήδημα καρέ (όταν ο πελάτης πηδάει τα καθυστερημένα καρέ).

Μέχρι τώρα, ο σημαντικότερος δείκτης μέτρησης της απόδοσης για βίντεο συνεχούς ροής είναι η μέση διεκπεραιωτική ικανότητα. Για να παρέχει συνεχή αναπαραγωγή, το δίκτυο πρέπει να παρέχει μία μέση διεκπεραιωτική ικανότητα σε όλα την εφαρμογή, η οποία να είναι τουλάχιστον τόσο μεγάλη όρο ο ρυθμός bit του ίδιο του βίντεο. Χρησιμοποιώντας ενταμίευση και εκ των προτέρων φόρτωση (prefetching) είμαστε ικανοί να παρέχουμε συνεχή αναπαραγωγή, ακόμα και όταν η διεκπεραιωτική ικανότητα έχει διακυμάνσεις, αρκεί η μέση διεκπεραιωτική ικανότητα να παραμένει μεγαλύτερη του ρυθμού του βίντεο.

Για πολλές εφαρμογές βίντεο συνεχούς ροής, το εγγεγραμμένο εκ των προτέρων βίντεο αποθηκεύεται και ρέει από ένα CDN παρά από ένα μοναδικό κέντρο δεδομένων. Υπάρχουν όμως πολλές εφαρμογές βίντεο συνεχούς ροής P2P, για τις οποίες το βίντεο αποθηκεύεται στους υπολογιστές των χρηστών και διαφορετικά κομμάτια του βίντεο φθάνουν από διαφορετικούς ομότιμους, που μπορεί να είναι διασκορπισμένοι σε όλον τον κόσμο.

2.3.2 Ήχος και Βίντεο Συζήτησης επάνω από IP

Η συνομιλία πραγματικού χρόνου με φωνή στο διαδίκτυο, συχνά αναφέρεται ως διαδικτυακή τηλεφωνία (Internet telephony), αφού από την σκοπιά του χρήστη, είναι παρόμοια με την παραδοσιακή υπηρεσία τηλεφωνικής μεταγωγής κυκλώματος. Συνήθως αναφέρεται ως Φωνή επάνω από IP (voice over IP). Η συνομιλία με βίντεο είναι παρόμοια, εκτός του ότι περιλαμβάνει το βίντεο των συμμετεχόντων, καθώς και τις φωνές τους. Τα περισσότερα από τα σημερινά συστήματα συνομιλίας με ήχο και βίντεο επιτρέπουν στους χρήστες να δημιουργούν διασκέψεις με πολλούς συμμετέχοντες. Συνομιλία ήχου/βίντεο χρησιμοποιείται σήμερα παγκοσμίως από μεγάλες εταιρίες (π.χ. Skype, Googletalk κτλ) οι οποίες φιλοξενούν καθημερινά εκατομμύρια χρήστες.

Όσον αφορά τις εφαρμογές συνομιλίας ήχου/βίντεο, υπάρχουν δύο πολύ σημαντικού άξονες σε θέματα χρονισμού και ανοχής σε απώλεια δεδομένων. Τα θέματα χρονισμού είναι σημαντικά, επειδή οι εφαρμογές συνομιλίας ήχου/βίντεο είναι άκρως ευαίσθητες στην καθυστέρηση (delay-sensitive). Για μία συνομιλία με δύο η περισσότερους ομιλητές, η καθυστέρηση από την στιγμή που ένας χρήστης μιλά η κινείται, μέχρι αυτή η ενέργεια να εκδηλωθεί στο άλλο άκρο, πρέπει να είναι

μικρότερος των μερικών εκατοντάδων χιλιοστών του δευτερολέπτου. Για την φωνή, καθυστερήσεις μικρότερες των 150 msec δεν γίνονται αντιληπτές από έναν ανθρώπινο ακροατή. Καθυστερήσεις ανάμεσα στα 150 και 400 msec μπορεί να είναι αποδεκτές και καθυστερήσεις που υπερβαίνουν τα όρια αυτά μπορούν να προκαλέσουν απογοήτευση, αν όχι τελείως ακατάληπτες συνομιλίες φωνής.

Από την άλλη, οι πολυμεσικές εφαρμογές συνομιλίας είναι ανθεκτικές σε απώλειες (loss-tolerant). Περιστασιακή απώλειας δεδομένων προκαλεί μικρά, σποραδικά προβλήματα στην αναπαραγωγή ήχου/βίντεο, ενώ συνήθως οι απώλειες αυτές μπορούν να αποκρύπτονται, είτε πλήρως, είτε εν μέρει. Αυτά τα ευαίσθητα στην καθυστέρηση, αλλά ανθεκτικά στ απώλειες χαρακτηριστικά είναι σαφώς διαφορετικά από τα χαρακτηριστικά των ελαστικών εφαρμογών, όπως είναι τα emails ή τα κοινωνικά δίκτυα. Για ελαστικές εφαρμογές οι μεγάλες καθυστερήσεις είναι μεν ενοχλητικές αλλά όχι ιδιαίτερα καταστροφικές. Καταλαβαίνουμε λοιπόν ότι η ακεραιότητα των μεταφερόμενων δεδομένων είναι ζωτικής σημασίας.

2.3.3 Ζωντανός ήχος και βίντεο συνεχούς ροής

Η Τρίτη κατηγορία εφαρμογών είναι παρόμοια με τις παραδοσιακές ραδιοφωνικές και τηλεοπτικές εκπομπές, εκτός από το γεγονός ότι η μετάδοση γίνεται μέσα του διαδικτύου. Αυτές οι εφαρμογές δίνουν στον χρήστη την δυνατότητα να λαμβάνει ζωντανή ραδιοφωνική ή τηλεοπτική μετάδοση από οποιαδήποτε γωνία του κόσμου. Οι εφαρμογές ζωντανής εκπομπής συνήθως έχουν πολλούς χρήστες οι οποίοι λαμβάνουν το ίδιο πρόγραμμα ήχου/βίντεο ταυτόχρονα. Αν και η διανομή ζωντανού ήχου/βίντεο σε πολλούς παραλήπτες μπορεί να επιτευχθεί αποδοτικά κάνοντας χρήση τεχνικών πολυεκπομπής IP (IP multicasting). Σήμερα επιτυγχάνεται μέσω πολυεκπομπής επιπέδου εφαρμογής χρησιμοποιώντας P2P η CDN δίκτυα ή μέσω πολλαπλών, ανεξάρτητων ροών δεδομένων (unicast streams). Όπως και με την συνεχή ροή αποθηκευμένου περιεχομένου πολυμέσων, το δίκτυο πρέπει να παρέχει σε κάθε ζωντανή ροή πολυμέσων μία μέση διεκπεραιωτική ικανότητα, η οποία να είναι μεγαλύτερη από τον ρυθμό κατανάλωσης του βίντεο. Επειδή το γεγονός είναι ζωντανό, η καθυστέρηση μπορεί επίσης να αποτελεί πρόβλημα, αν και οι περιορισμοί χρονισμού είναι λιγότερο αυστηροί από τους περιορισμούς της συνομιλίας φωνής. Καθυστερήσεις έως δέκα δευτερόλεπτα περίπου από την στιγμή που ο χρήστης επιλέγει να δει μία ζωντανή μετάδοση μέχρι την στιγμή που ξεκινά η αναπαραγωγή μπορούν να είναι ανεκτές.

2.4 Πρωτόκολλα streaming

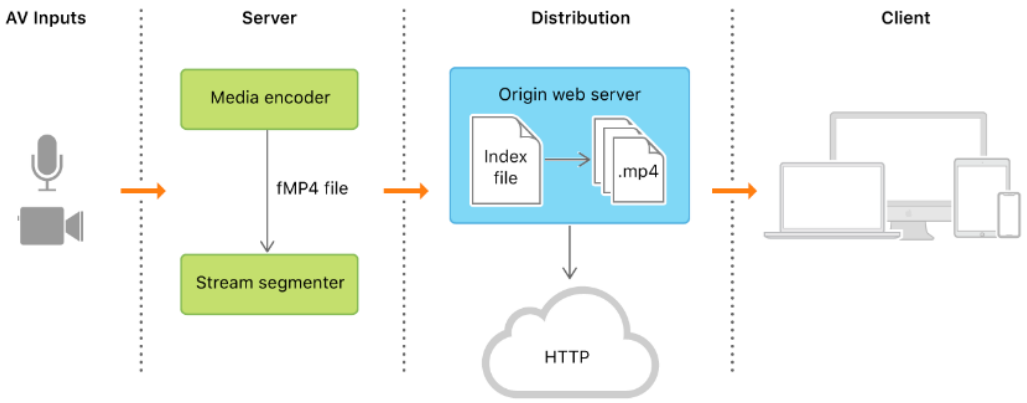
Αρχικά, πριν αναφέρουμε τα πρωτόκολλα, θα χρειαστεί να εξηγήσουμε τι ακριβώς είναι το streaming και πως λειτουργεί. Οι πρώτοι ιστότοποι ήταν απλές σελίδες κειμένου με ίσως μια ή δύο εικόνες. Σήμερα, ωστόσο, όποιος έχει αρκετά γρήγορη σύνδεση στο Διαδίκτυο μπορεί να παρακολουθήσει ταινίες υψηλής ευκρίνειας ή να πραγματοποιήσει βιντεοκλήση μέσω Διαδικτύου. Αυτό είναι δυνατό λόγω μιας τεχνολογίας που ονομάζεται ροή. Η ροή είναι η συνεχής μετάδοση αρχείων ήχου ή βίντεο από έναν διακομιστή σε έναν πελάτη. Με απλούστερους όρους, η ροή είναι αυτό που συμβαίνει όταν οι καταναλωτές παρακολουθούν τηλεόραση ή ακούν podcast σε συσκευές συνδεδεμένες στο Διαδίκτυο. Με τη ροή, το αρχείο πολυμέσων που αναπαράγεται στη συσκευή-πελάτη αποθηκεύεται εξ αποστάσεως και μεταδίδεται λίγα δευτερόλεπτα τη φορά μέσω του Διαδικτύου.

Σχετικά με την λειτουργία του, όπως και άλλα δεδομένα που αποστέλλονται μέσω του Διαδικτύου, τα δεδομένα ήχου και βίντεο αναλύονται σε πακέτα δεδομένων. Κάθε πακέτο περιέχει ένα μικρό κομμάτι του αρχείου και ένα πρόγραμμα αναπαραγωγής ήχου ή βίντεο στο πρόγραμμα περιήγησης στη συσκευή-πελάτη παίρνει τη ροή των πακέτων δεδομένων και τα ερμηνεύει ως βίντεο ή ήχο.

Για εφαρμογές βίντεο συνεχούς ροής, υπάρχουν βίντεο τα οποία τοποθετούνται εκ των προτέρων σε εξυπηρετητές και οι διάφοροι χρήστες στέλνουν αιτήματα ώστε να μπορούν να δουν τα βίντεο κατ΄απαίτηση. Ο χρήστης έχει την δυνατότητα να παρακολουθήσει το βίντεο από την αρχή μέχρι το τέλος, χωρίς να υπάρξει κάποια διακοπή στο ενδιάμεσο, να σταματήσει την παρακολούθηση του βίντεο πριν τελειώσει ή τέλος να αλληλεπιδράσει με το βίντεο, είτε κάνοντας παύσης είτε ανατοποθετώντας το σημείο αναπαραγωγής σε κάποια μελλοντική ή παρελθοντική σκηνή. Τα συστήματα βίντεο συνεχούς ροής μπορούν να διαχωριστούν σε έξι κατηγορίες: HTTP συνεχούς ροής (HTTP streaming),HTTP προσαρμόσιμης συνεχούς ροής(adaptive HTTP streaming),WebRTC, Secure Reliable Transport (SRT), Real-TimeMessagingProtocol (RTMP) και Real-Time Streaming Protocol (RTSP).

2.4.1 HTTP συνεχούς ροής (HTTP streaming)

Στο HTTP συνεχούς ροής, το βίντεο απλώς αποθηκεύεται σ΄έναν εξυπηρετητή HTTP ως ένα συνηθισμένο αρχείο, με ένα συγκεκριμένο URL. Όταν κάποιος χρήστης θέλει να παρακολουθήσει το βίντεο, δημιουργείται μια TCP σύνδεση με τον εξυπηρετητή και εκδίδει μία αίτηση (request) HTTPGET για το URL αυτό. Κατόπιν, ο εξυπηρετητής στέλνει το αρχείο του βίντεο, μέσα σε ένα μήνυμα απόκρισης HTTP, όσο το δυνατόν γρηγορότερα (δηλαδή όσο πιο γρήγορα επιτρέπουν ο έλεγχος συμφόρησης και ο έλεγχος ροής). Από την πλευρά του πελάτη, τα bytes συλλέγονται σε έναν ενταμιευτή εφαρμογής πελάτη. Μόλις ο αριθμός των bytes μέσα στον ενταμιευτή υπερβούν ένα προκαθορισμένο κατώφλι, η εφαρμογή πελάτη ξεκινάει την αναπαραγωγή παίρνοντας περιοδικά καρέ του βίντεο απ΄τον ενταμιευτή της εφαρμογής πελάτη, αποσυμπιέζει τα καρέ και τα εμφανίζει στην οθόνη του χρήστη. Η χρήση του HTTP επάνω από TCP επιτρέπει επίσης το βίντεο να διασχίζει firewalls και NAT πιο εύκολα (τα οποία υπάρχουν ώστε να μπλοκάρουν το μεγαλύτερο μέρος της κίνησης UDP, αλλά να επιτρέπουν την περισσότερη κίνηση HTTP. Η συνεχής ροή επάνω από HTTP αποφεύγει επίσης την ανάγκη για κάποιο εξυπηρετητή ελέγχου μέσων, π.χ. για έναν εξυπηρετητή RTSP, μειώνοντας το κόστος μίας υλοποίησης μεγάλης κλίμακας επί του Internet.



Σχήμα 2.9 Αρχιτεκτονική HTTP stream

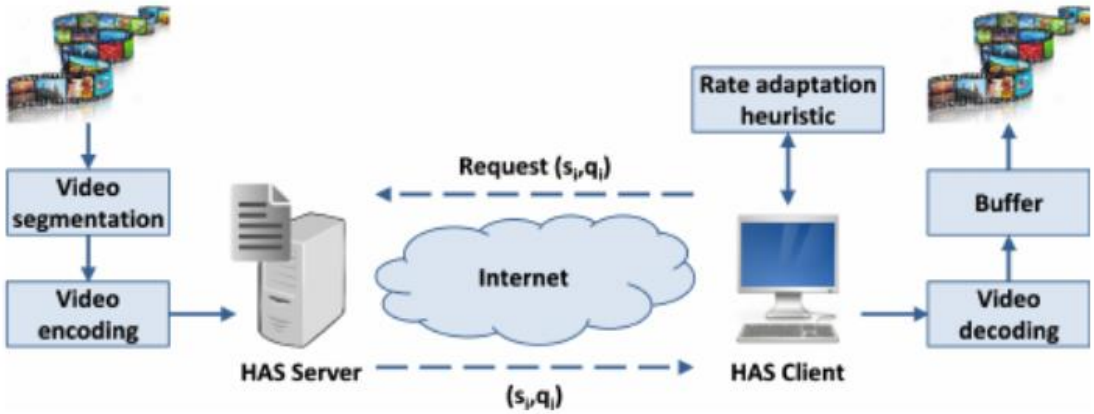
2.4.2 HTTP προσαρμόσιμης συνεχούς ροής (adaptive HTTP streaming) και DASH

Αν και το HTTP streaming, όπως το περιγράφτηκε νωρίτερα, χρησιμοποιείται εκτεταμένα, από την στιγμή της έναρξης του έχει μεγάλο μειονέκτημα. Όλοι οι πελάτες λαμβάνουν την ίδια κωδικοποίηση του βίντεο, παρά τις μεγάλες μεταβολές στο διαθέσιμο εύρος ζώνης που διατίθεται σε κάποιο πελάτη, τόσο σε διαφορετικούς πελάτες, όσο και μέσα στον χρόνο στον ίδιο πελάτη. Το συγκεκριμένο, οδήγησε στην ανάπτυξη ενός άλλου τύπου συνεχούς ροής που βασίζεται σε HTTP, που ονομάζεται Δυναμική Προσαρμόσιμη Συνεχής Ροή επάνω από HTTP(Dynamic Adaptive Streaming over HTTP,DASH). Στο DASH, το βίντεο κωδικοποιείται σε αρκετές διαφορετικές εκδόσεις και κάθε έκδοση έχει έναν διαφορετικό αριθμό bit, συνεπώς ένα διαφορετικό επίπεδο ποιότητας. Ο πελάτης ζητά δυναμικά κομμάτια τμημάτων βίντεο μερικών δευτερολέπτων από τις διαφορετικές εκδόσεις. Όταν η ποσότητα του διαθέσιμου εύρους ζώνης είναι μεγάλη, ο πελάτης επιλέγει κομμάτια από μία έκδοσης υψηλότερου ρυθμού bit και όταν το διαθέσιμο εύρος ζώνης είναι μικρό, επιλέγει από μία έκδοση χαμηλού ρυθμού bit. Ο πελάτης επιλέγει διαφορετικά κομμάτια ένα προς ένα με μηνύματα HTTPGET.

Από την μία, το DASH επιτρέπει σε πελάτες με διαφορετικούς ρυθμούς προσπέλασης του διαδικτύου να λαμβάνουν βίντεο σε συνεχόμενη ροή με διαφορετικούς ρυθμούς κωδικοποίησης. Οι πελάτες με συνδέσεις χαμηλής ταχύτητας μπορούν να λαμβάνουν μία έκδοσης χαμηλού ρυθμού bit, άρα και χαμηλές ποιότητας και με πελάτες π.χ. με συνδέσεις οπτικών ινών μπορούν να λαμβάνουν μία έκδοσης υψηλής ποιότητας. Από την άλλη, το DASH επιτρέπει σ΄έναν πελάτη να προσαρμοσθεί στο διαθέσιμο εύρος ζώνης, εάν το εύρος ζώνης από άκρο σε άκρο αλλάζει κατά την διάρκεια της συνόδου. Αυτό το χαρακτηριστικό είναι ιδιαίτερα σημαντικό για κινητούς χρήστες, οι οποίοι τυπικά βλέπουν την διαθεσιμότητα του εύρος ζώνης τους να κυμαίνεται, καθώς μετακινούνται σε σχέσης με τους σταθμούς βάσης. Για παράδειγμα, η Comcast έχει υλοποιήσει ένα σύστημα προσαρμόσιμης συνεχούς ροής, με βάση το οποίο κάθε αρχείο προέλευσης βίντεο κωδικοποιείται σε 8 ως 10 διαφορετικές μορφοποιήσεις, επιτρέποντας στο βίντεο με την μορφοποίησης υψηλότερης ποιότητας να ρέει στον πελάτη, με την προσαρμογή να γίνεται ως απόκριση σε μεταβαλλόμενες συνθήκες δικτύου και συσκευών.

Με το DASH, κάθε έκδοση του βίντεο αποθηκεύεται μέσα στον εξυπηρετητή HTTP και η κάθε μία έχει διαφορετικό URL. Ο εξυπηρετητής HTTP έχει επίσης ένα αρχείο manifest, το οποίο παρέχει ένα URL για κάθε έκδοση, μαζί με τον bit ρυθμό. Ο πελάτης, ζητά πρώτα το αρχείο manifest και μαθαίνει για τις διάφορες εκδόσεις. Κατόπιν, ο πελάτης επιλέγει ένα κομμάτι κάθε φορά, καθορίζοντας ένα URL και μία περιοχή byte μέσα σ’ ένα μήνυμα αίτησης HTTPGET για κάθε κομμάτι. Ενώ κατεβάζει κομμάτια, ο πελάτης μετρά επίσης το εύρος ζώνης λήψης και εκτελεί έναν αλγόριθμο καθορισμού ρυθμού, ώστε να επιλέξει το κομμάτι, που θα αιτηθεί στην συνέχεια. Εννοείται ότι, εάν ένας πελάτης έχει πολύ βίντεο ενταμιευμένο και εάν το μετρούμενο εύρος ζώνης λήψης είναι μεγάλο, θα επιλέξει ένα κομμάτι από μία έκδοση υψηλού ρυθμού. Το DASH επιτρέπει λοιπόν στους πελάτες να επιλέξουν ελεύθερα ανάμεσα στα διαφορετικά επίπεδα ποιότητας. Αφού μία ξαφνική πρώτη στον ρυθμό bit και η αλλαγή έκδοσης μπορεί να έχει ως αποτέλεσμα μια παρατηρήσιμη υποβάθμιση της ποιότητας, η μείωση του ρυθμού bit μπορεί να επιτευχθεί χρησιμοποιώντας ενδιάμεσες εκδόσεις, ώστε να γίνει ομαλή αλλαγή σε έναν ρυθμό, όταν ο ρυθμός κατανάλωσης του πελάτη πέφτει κάτω από το διαθέσιμο εύρος ζώνης λήψης του. Όταν βελτιωθούν οι συνθήκες του δικτύου, ο πελάτης μπορεί να επιλέξει κομμάτια από εκδόσεις με υψηλότερο ρυθμό bit. Παρακολουθώντας δυναμικά το διαθέσιμο εύρος ζώνης και το επίπεδο ενταμίευσης στον πελάτη και ρυθμίζοντας τον ρυθμό μετάδοσης με εναλλαγή εκδόσεων, το DASH μπορεί συχνά να επιτύχει συνεχή αναπαραγωγή με το καλύτερο δυνατό επίπεδο ποιότητας, χωρίς πάγωμα η παράκαμψη καρέ. Ακόμη αφού ο πελάτης έχει την ευφυΐα να καθορίσει

ποιο κομμάτι να στείλει στην συνέχεια, το σχήμα βελτιώσει επίσης της δυνατότητα κλιμάκωσης στην πλευρά του εξυπηρετητή. Ένα άλλο όφελος αυτής της προσέγγισης είναι ότι ο πελάτης μπορεί να χρησιμοποιήσει την αίτηση περιοχής-Byte HTTP για να ελέγξει με ακρίβεια την ποσότητα του εκ τω προτέρων φορτωμένου βίντεο, που ενταμιεύει τοπικά. Αξίζει να σημειωθεί ότι το συγκεκριμένο πρωτόκολλο το χρησιμοποιούν κορυφαίες εταιρίες παγκοσμίως όπως είναι η Netflix, η YouTube, η Microsoft, η Googlεκτλ.



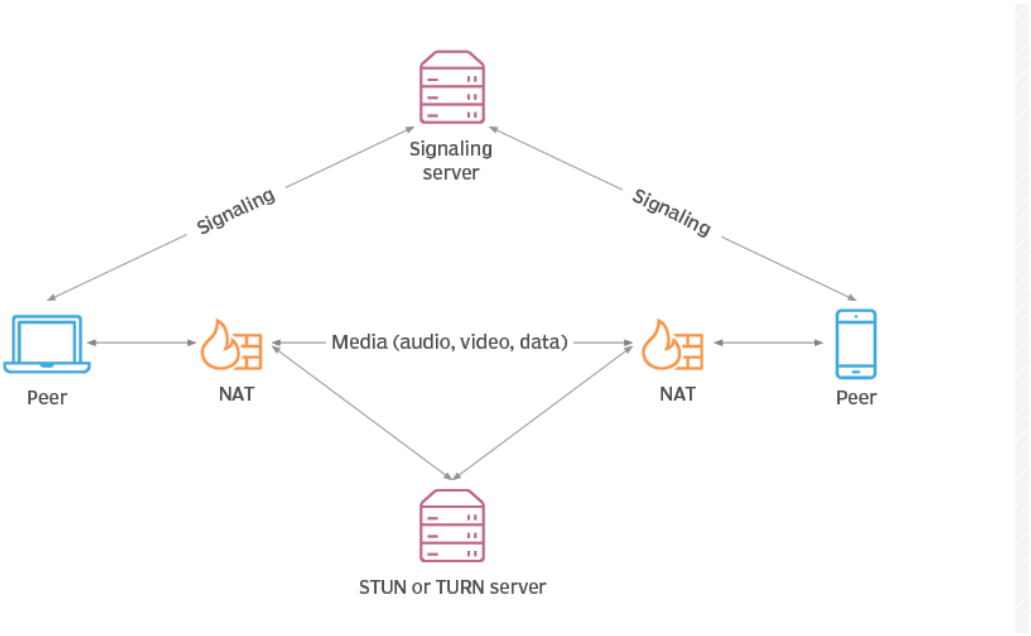
Σχήμα 2.2 Αρχιτεκτονική HTTP adaptive stream

2.4.3 WebRTC

Το WebRTC είναι ένα open-source project που στοχεύει στην παροχή ροής με real-time καθυστέρηση. Αρχικά αναπτύχθηκε για εφαρμογές που βασίζονται σε καθαρά chat και χρήση VoIP, έγινε γνωστό για χρήση σε εφαρμογές συνομιλίας μέσω βίντεο και συνεδριάσεων αφού αγοράστηκε από την Google. Μερικές από τις πιο κοινές εφαρμογές της εποχής που απευθύνονται σε καταναλωτές, όπως το GoogleMeet, το Discord, το Messenger χρησιμοποιούν όλες το WebRTC. Ως συνδυασμός προτύπων, πρωτοκόλλων και API JavaScript, το WebRTC αξιοποιεί συνδέσεις peer-to-peer μεταξύ προγραμμάτων περιήγησης για την υποστήριξη σχεδόν ταυτόχρονης ανταλλαγής δεδομένων χωρίς να απαιτείται λογισμικό ή πρόσθετα plugins. Επίσης, επειδή το WebRTC χρησιμοποιεί τρία API HTML5 που επιτρέπουν στα προγράμματα περιήγησης των χρηστών να καταγράφουν, να κωδικοποιούν και να μεταδίδουν ζωντανές ροές μεταξύ τους, επιτρέποντας αμφίδρομη επικοινωνία, είναι και ο λόγος για τον οποίο το WebRTC αναφέρεται ως τεχνολογία peer-to-peer, σύμφωνα με την οποία κάθε πρόγραμμα περιήγησης επικοινωνεί απευθείας μεταξύ τους.

Η ομορφιά του WebRTC βρίσκεται στο γεγονός ότι εξαλείφει την ανάγκη για ενδιάμεσους διακομιστές Ιστού κατά τη διάρκεια αυτών των ανταλλαγών, για να μην αναφέρουμε πρόσθετο εξοπλισμό ή λογισμικό. Οι online αίθουσες συσκέψεων είναι ένα εξαιρετικό παράδειγμα της ευκολίας και της επικοινωνίας σε πραγματικό χρόνο που παρέχεται από το WebRTC. Ενώ ορισμένες ροές εργασίας ροής απαιτούν κάμερα ζωντανής ροής, κωδικοποιητή και διακομιστή πολυμέσων, οι απλούστερες αναπτύξεις WebRTC μπορούν να επιτύχουν τα πάντα με μια συνδεδεμένη κάμερα web και πρόγραμμα περιήγησης. Και σε αντίθεση με το βίντεο που βασίζεται σε Flash, το WebRTC μπορεί να αναπαραχθεί σε οποιοδήποτε πρόγραμμα αναπαραγωγής HTML5 που υποστηρίζει WebRTC API.

Το WebRTC είναι συνυφασμένο με υψηλές ταχύτητες παράδοσης. Με καθυστέρηση κάτω από 500 χιλιοστά του δευτερολέπτου από προσφέρει την ταχύτερη μέθοδο για τη μεταφορά βίντεο στο διαδίκτυο. Όλα τα μεγάλα προγράμματα περιήγησης και συσκευές υποστηρίζουν το WebRTC, καθιστώντας εύκολη την ενσωμάτωση σε ένα ευρύ φάσμα εφαρμογών χωρίς αποκλειστική υποδομή. Εφόσον χρησιμοποιεί API HTML5, μπορεί και επιτρέπει στους προγραμματιστές να χρησιμοποιούν πολλές από τις δυνατότητες που είναι ενσωματωμένες στη γλώσσα προγραμματισμού HTML5 μέσω ενός ελαφρού, ενσωματωμένου framework. Βέβαια το συγκεκριμένο πρωτόκολλο δεν σχεδιάστηκε με γνώμονα την επεκτασιμότητα. Η ρύθμιση παραμέτρων έντασης εύρους ζώνης απαιτεί από κάθε συμμετέχοντα πρόγραμμα περιήγησης να συνδέεται μεταξύ τους μέσω μιας peer σύνδεσης με αποτέλεσμα να μην συνίσταται να μην υπάρχουν πολλές ταυτόχρονες peer συνδέσεις .



Σχήμα 2.3 Παράδειγμα επικοινωνίας με την χρήση WebRTC

2.4.4 Secure Reliable Transport (SRT)

Το SRT είναι ένα πρωτόκολλο μεταφοράς ροής βίντεο και μια στοίβα τεχνολογίας που έχει σχεδιαστεί για τη σύνδεση δύο τελικών σημείων με σκοπό την παροχή βίντεο χαμηλής καθυστέρησης και άλλων ροών πολυμέσων σε δίκτυα με απώλειες, όπως το δημόσιο Διαδίκτυο. Με λίγα λόγια, το SRT φέρνει την καλύτερη ποιότητα ζωντανού βίντεο στα χειρότερα δίκτυα συνδυάζοντας την αστραπιαία ταχύτητα του UDP με τις ιδιότητες διόρθωσης σφαλμάτων του TCP. Αντιπροσωπεύει την απώλεια πακέτων, το jitter και το κυμαινόμενο εύρος ζώνης, διατηρώντας ταυτόχρονα την ακεραιότητα και την ποιότητα του βίντεο. Με το SRT, μπορείτε να διατηρήσετε τις ροές σας ασφαλείς και να διασχίσετε εύκολα τείχη προστασίας. Το SRT χρησιμοποιεί μια προσωρινή μνήμη καθυστέρησης τόσο στην πλευρά του αποστολέα όσο και στην πλευρά του παραλήπτη κατά τη διάρκεια της μετάδοσης. Η προσωρινή μνήμη της πλευράς του δέκτη αναδομεί το πακέτο ροής του αποστολέα και τα ταξινομεί πριν τα στείλει στον αποκωδικοποιητή. Σε περίπτωση που συμβεί οποιαδήποτε απώλεια πακέτου μεταξύ της διαδικασίας, το buffer του δέκτη στέλνει μια αρνητική επιβεβαίωση στην πλευρά του αποστολέα. Σε απόκριση, το χαμένο πακέτο δεδομένων αποστέλλεται εκ νέου από το buffer αποστολέα. Για να διασφαλιστεί η ροή δεδομένων μεταξύ διαφορετικών δικτύων, το SRT

περιλαμβάνει λειτουργία ακροατή και καλούντος. Αποφασίζει ποιος θα ξεκινήσει τη σύνδεση και ποιος θα την αποδεχτεί. Με τη χρήση καλούντων και ακροατών, είναι δυνατή η χρήση οποιουδήποτε τείχους προστασίας για ροή εντός ενός τομέα. Η ασφαλής αξιόπιστη μεταφορά επιτρέπει στους χρήστες να μεταδίδουν διαφορετικούς τύπους περιεχομένου, συμπεριλαμβανομένων των MPEG-4/AVC και HEVC (κωδικοποίηση βίντεο υψηλής απόδοσης).

Χρησιμοποιώντας την ίδια κρυπτογράφηση AES 128/256-bit που εμπιστεύονται κυβερνήσεις και οργανισμούς σε όλο τον κόσμο, το SRT διασφαλίζει ότι το πολύτιμο περιεχόμενο προστατεύεται από άκρο σε άκρο από τη συνεισφορά στη διανομή, ώστε να μην μπορούν να ακούσουν μη εξουσιοδοτημένα μέρη. Ανεξάρτητα από το πόσο αναξιόπιστο είναι ένα δίκτυο σας, το SRT μπορεί να ανακάμψει από τη σοβαρή απώλεια πακέτων διασφαλίζοντας την ακεραιότητα και την ποιότητα των ροών βίντεο σας. Σε αντίθεση με ορισμένα άλλα πρωτόκολλα ροής που υποστηρίζουν μόνο συγκεκριμένες μορφές βίντεο και ήχου, το SRT είναι αγνωστικιστικό ωφέλιμου φορτίου. Επειδή το SRT λειτουργεί σε επίπεδο μεταφοράς δικτύου, ενεργώντας ως περιτύλιγμα γύρω από το περιεχόμενό σας, μπορεί να μεταφέρει οποιονδήποτε τύπο μορφής βίντεο, κωδικοποιητή, ανάλυσης ή ταχύτητας καρέ. Το SRT χρησιμοποιείται από χιλιάδες οργανισμούς παγκοσμίως για ένα ευρύ φάσμα εφαρμογών, από κάμερες IP, κωδικοποιητές βίντεο και αποκωδικοποιητές έως πύλες (gateways). Ορισμένες ενδεικτικές εταιρίες που χρησιμοποιούν το SRT είναι η AWS, η Microsoft και η NASA.

Σχήμα 2.4 Γενικό overview του SRT

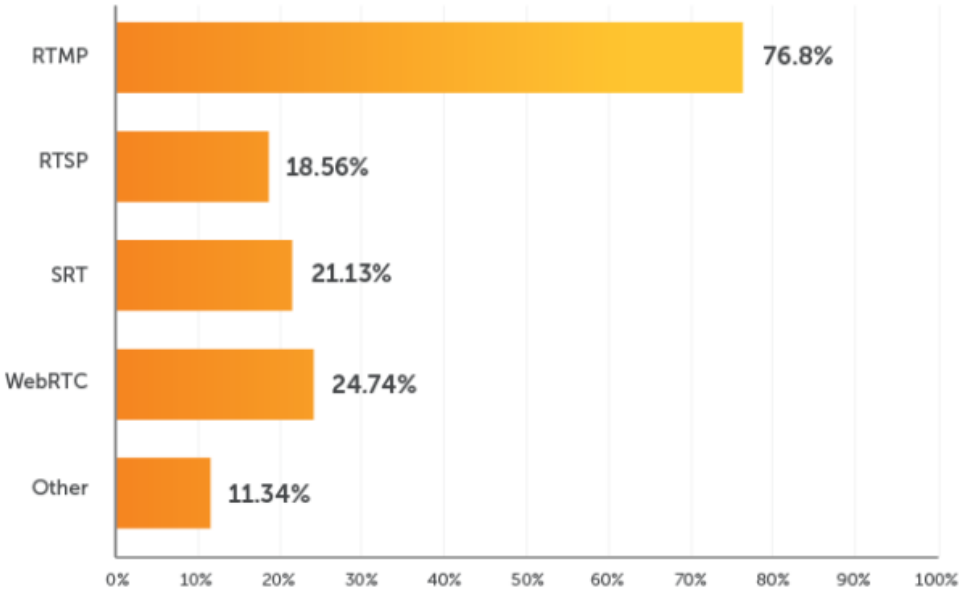
2.4.5Real-Time Messaging Protocol (RTMP)

Το RTMP πρόκειται για ένα πρωτόκολλο ροής που αρχικά σχεδιάστηκε για τη μετάδοση ήχου, βίντεο και άλλων δεδομένων μεταξύ ενός αποκλειστικού διακομιστή ροής και του Adobe Flash Player. Αν και κάποτε ήταν ιδιόκτητο, το RTMP είναι τώρα open-source. Το πρωτόκολλο ανταλλαγής μηνυμάτων σε πραγματικό χρόνο (RTMP) της Adobe παρέχει μια αμφίδρομη υπηρεσία πολλαπλών μηνυμάτων μέσω αξιόπιστης μεταφοράς ροής, όπως το TCP που προορίζεται να μεταφέρει παράλληλες ροές μηνυμάτων βίντεο, ήχου και δεδομένων, με σχετικές πληροφορίες χρονισμού, μεταξύ ενός ζεύγους της επικοινωνίας των peers. Η Macromedia (η οποία είναι σήμερα η Adobe Systems) ανέπτυξε την προδιαγραφή RTMP για μετάδοση δεδομένων ήχου και εικόνας υψηλής απόδοσης. Το RTMP διατηρεί μια σταθερή σύνδεση μεταξύ του προγράμματος-πελάτη του προγράμματος αναπαραγωγής και του διακομιστή, επιτρέποντας στο πρωτόκολλο να λειτουργεί ως σωλήνας (pipe) και να μεταφέρει γρήγορα δεδομένα βίντεο στον θεατή.

Επειδή το RTMP βρίσκεται πάνω από το Πρωτόκολλο Ελέγχου Μετάδοσης (TCP), χρησιμοποιεί μια τριπλή χειραψία κατά τη μεταφορά δεδομένων. Ο εκκινητής (πελάτης) ζητά από τον αποδέκτη

(διακομιστή) να ξεκινήσει μια σύνδεση. ο αποδέκτης απαντά. τότε ο εκκινητής αναγνωρίζει την απάντηση και διατηρεί μια συνεδρία μεταξύ των δύο άκρων. Για το λόγο αυτό, το RTMP είναι αρκετά αξιόπιστο. Το συγκεκριμένο πρωτόκολλο διατηρεί μόνιμες συνδέσεις και επιτρέπει επικοινωνία χαμηλής καθυστέρησης. Για την ομαλή παροχή ροών και τη μετάδοση όσο το δυνατόν περισσότερων πληροφοριών, χωρίζει τις ροές σε τμήματα και το μέγεθός τους διαπραγματεύεται δυναμικά μεταξύ του πελάτη και του διακομιστή. Μερικές φορές, διατηρείται αμετάβλητο. τα προεπιλεγμένα μεγέθη θραυσμάτων είναι 64 byte για δεδομένα ήχου και 128 byte για δεδομένα βίντεο και τους περισσότερους άλλους τύπους δεδομένων. Θραύσματα από διαφορετικά ρεύματα μπορούν στη συνέχεια να παρεμβληθούν και να πολυπλεχθούν σε μία μόνο σύνδεση. Με μεγαλύτερα κομμάτια δεδομένων, το πρωτόκολλο φέρει έτσι μόνο μια κεφαλίδα ενός byte ανά τμήμα, οπότε επιβαρύνεται με πολύ μικρό κόστος. Ωστόσο, στην πράξη, μεμονωμένα θραύσματα τυπικά δεν παρεμβάλλονται. Αντίθετα, η παρεμβολή και η πολυπλεξία γίνονται σε επίπεδο πακέτων, με πακέτα RTMP σε πολλά διαφορετικά ενεργά κανάλια να παρεμβάλλονται με τέτοιο τρόπο ώστε να διασφαλίζεται ότι κάθε κανάλι πληροί το εύρος ζώνης, την καθυστέρηση και άλλες απαιτήσεις ποιότητας υπηρεσίας. Τα πακέτα που παρεμβάλλονται με αυτόν τον τρόπο αντιμετωπίζονται ως αδιαίρετα και δεν παρεμβάλλονται σε επίπεδο τμήματος.

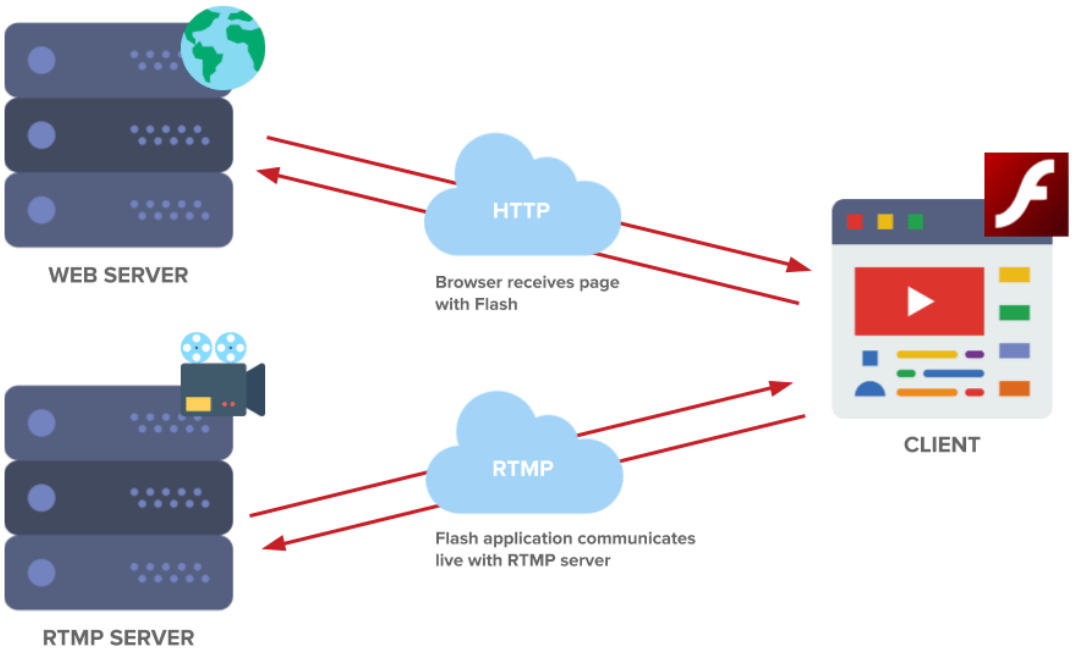
Αρχικά, το RTMP χρησιμοποιήθηκε για τη μετάδοση περιεχομένου μεταξύ ενός προγράμματος αναπαραγωγής βίντεο και ενός διακομιστή φιλοξενίας, ο οποίος αναφερόταν ως "RTMP delivery" . Σήμερα, ο σκοπός του είναι λίγο διαφορετικός. Όσον αφορά τις πιο σύγχρονες ρυθμίσεις ζωντανής ροής, ο πρωταρχικός ρόλος του RTMP είναι να παρέχει περιεχόμενο από έναν κωδικοποιητή σε έναν διαδικτυακό οικοδεσπότη βίντεο. Αυτή η λειτουργία RTMP αναφέρεται ως " RTMP ingest".



Σχήμα 2.5 Streaming μορφές που χρησιμοποιούνται σήμερα για ingest

Το συγκεκριμένο πρωτόκολλο διαθέτει μία προσαρμόσιμη ροή σημαίνει ότι οι θεατές σας δεν είναι κλειδωμένοι να παρακολουθούν τις ροές σας σε μία γραμμική κατεύθυνση. Η προσαρμοστικότητα τους επιτρέπει να παραλείπουν και να επαναφέρουν (rewind) τμήματα της ροής ή να συμμετέχουν σε μια ζωντανή ροή μετά την έναρξή της (είναι μία δυνατότητα που σχεδόν όλοι οι χρήστες επιθυμούν να

υπάρχει). Βέβαια, επειδή υποστηρίζεται από προγράμματα αναπαραγωγής Flash μπορούμε εύκολα να συμπεράνουμε ότι θα δημιουργηθούν ορισμένα θέματα συμβατότητας καθώς το Flash είναι εδώ και λίγο καιρό απαρχαιωμένο καθώς τα σύγχρονα πρότυπα βασίζονται σε HTML5 players και για την ορθή λειτουργία του το RTMP θα χρειαστεί κάποιο μετατροπέα. Επίσης, το RTMP δεν μπορεί να κάνει απευθείας stream πάνω από μία HTTP σύνδεση. Για να χρησιμοποιήσετε μια ροή RTMP στον εκάστοτε ιστότοπο, πρέπει να υπάρχει σύνδεση με έναν ειδικό διακομιστή, όπως ο Flash Media Server. Τέλος, τα RTMP stream μπορεί να είναι ιδιαίτερα ευάλωτα σε ζητήματα χαμηλού εύρους ζώνης βίντεο. Αυτό μπορεί να προκαλέσει συχνές, απογοητευτικές διακοπές στις ροές σας που καταστρέφουν την εμπειρία για τους θεατές σας.



Σχήμα 2.6 Ανταλλαγή πληροφοριών μεταξύ συσκευών με το RTMP

2.4.6 Real-Time Streaming Protocol (RTSP)

Το RTSP είναι ένα πρωτόκολλο επιπέδου εφαρμογής που χρησιμοποιείται για την εντολή διακομιστών πολυμέσων ροής μέσω δυνατοτήτων παύσης και αναπαραγωγής. Διευκολύνει έτσι τον έλεγχο σε πραγματικό χρόνο των μέσων ροής μέσω της επικοινωνίας με τον διακομιστή — χωρίς να μεταδίδει πραγματικά τα ίδια τα δεδομένα. Αντίθετα, οι διακομιστές RTSP χρησιμοποιούν συχνά το πρωτόκολλο μεταφοράς σε πραγματικό χρόνο (RTP) σε συνδυασμό με το πρωτόκολλο ελέγχου σε πραγματικό χρόνο (RTCP) για να μετακινήσουν τα πραγματικά δεδομένα ροής. Όταν ένας χρήστης ξεκινά μια ροή βίντεο από μια κάμερα IP χρησιμοποιώντας RTSP, η συσκευή στέλνει ένα αίτημα RTSP στον διακομιστή ροής. Αυτό ξεκινά τη διαδικασία εγκατάστασης. Στη συνέχεια, τα δεδομένα βίντεο και ήχου μπορούν στη συνέχεια να μεταδοθούν χρησιμοποιώντας RTP. Επομένως, μπορούμε να σκεφτούμε το RTSP ως ένα τηλεχειριστήριο τηλεόρασης για ροή πολυμέσων, με το RTP να λειτουργεί ως η ίδια η εκπομπή. Το συγκεκριμένο πρωτόκολλο, αντί να αναγκάζει τον θεατή να

κατεβάσει ένα ολόκληρο βίντεο πριν το παρακολουθήσει, του επιτρέπει να παρακολουθήσει το περιεχόμενό πριν ολοκληρωθεί η λήψη.

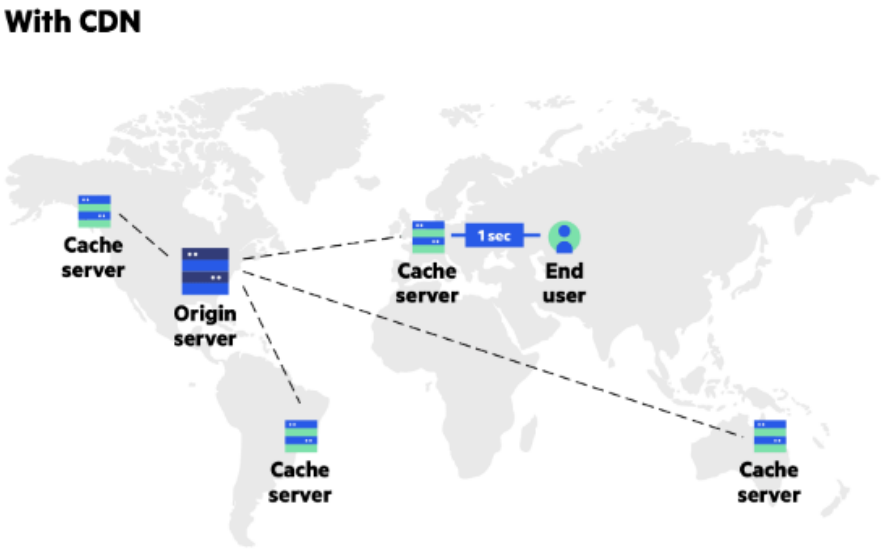
Σε σύγκριση με άλλα πρωτόκολλα ροής πολυμέσων, το RTSP είναι πολύ λιγότερο δημοφιλές. Τα περισσότερα προγράμματα αναπαραγωγής βίντεο και οι υπηρεσίες ροής δεν υποστηρίζουν ροή RTSP, γεγονός που καθιστά πιο δύσκολη τη μετάδοση της ροής στο πρόγραμμα περιήγησής του τελικού χρήστη. Για να μεταδοθεί μια ροή RTSP, πρέπει να χρησιμοποιηθεί μια ξεχωριστή υπηρεσία ζωντανής ροής RTSP. Τέλος, όπως το RTMP, δεν μπορεί να γίνει απευθείας ροή RTSP μέσω HTTP. Εξαιτίας αυτού, δεν υπάρχει εύκολος, άμεσος τρόπος ροής του RTSP σε ένα πρόγραμμα περιήγησης ιστού, καθώς το RTSP έχει σχεδιαστεί περισσότερο για ροή βίντεο σε ιδιωτικά δίκτυα, όπως συστήματα ασφαλείας σε μια επιχείρηση. Ωστόσο, μπορεί να γίνει ροή RTSP χρησιμοποιώντας πρόσθετο λογισμικό που είναι ενσωματωμένο στον ιστότοπο του χρήστη.

Σχήμα 2.7 Επικοινωνία με clients με την χρήση RTSP

2.5 Δίκτυα διανομής περιεχομένου (CDN)

Τα δίκτυα παράδοσης περιεχομένου (CDN) είναι η διαφανής ραχοκοκαλιά (backbone) του Διαδικτύου που είναι υπεύθυνη για την παράδοση περιεχομένου. Είτε το γνωρίζουμε είτε όχι, ο καθένας από εμάς αλληλεπιδρά με τα CDN σε καθημερινή βάση. κατά την ανάγνωση άρθρων σε ειδησεογραφικούς ιστότοπους, τις αγορές στο διαδίκτυο, την παρακολούθηση βίντεο YouTube ή την εξέταση των ροών κοινωνικής δικτύωσης. Ανεξάρτητα από το τι κάνετε ή τον τύπο περιεχομένου που καταναλώνετε, το πιθανότερο είναι ότι θα βρείτε CDN πίσω από κάθε χαρακτήρα κειμένου, κάθε pixel εικόνας και κάθε καρέ ταινίας που παραδίδεται στον υπολογιστή σας και στο πρόγραμμα περιήγησης για κινητά. Για να κατανοήσουμε γιατί τα CDN χρησιμοποιούνται τόσο ευρέως, πρέπει πρώτα να αναγνωρίσουμε το πρόβλημα που έχουν σχεδιαστεί να επιλύουν. Γνωστή ως λανθάνουσα κατάσταση (latency), είναι η ενοχλητική καθυστέρηση που συμβαίνει από τη στιγμή που ζητάτε να φορτώσετε μια ιστοσελίδα μέχρι τη στιγμή που το περιεχόμενό της εμφανίζεται πραγματικά στην οθόνη. Αυτό το διάστημα καθυστέρησης επηρεάζεται από διάφορους παράγοντες, πολλοί από τους οποίους είναι συγκεκριμένοι για μια δεδομένη ιστοσελίδα. Ωστόσο, σε όλες τις περιπτώσεις, η διάρκεια της καθυστέρησης επηρεάζεται από τη φυσική απόσταση μεταξύ εσάς και του διακομιστή φιλοξενίας αυτού του ιστότοπου.Η αποστολή ενός CDN είναι να μειώσει ουσιαστικά αυτή τη φυσική απόσταση, με στόχο τη βελτίωση της ταχύτητας και της απόδοσης απόδοσης ιστότοπου.

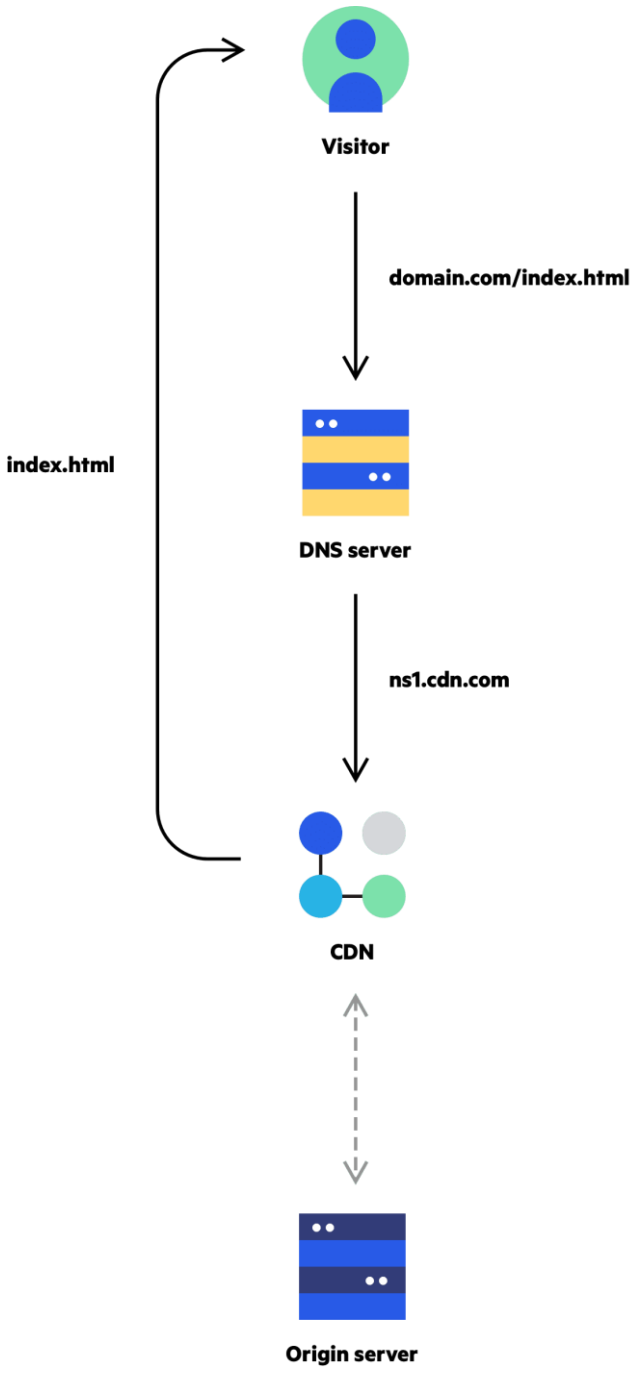
Σχήμα 2.8 Χρόνος απόκρισης αιτήματος μεταξύ χρήστη και server χωρίς CDN



Σχήμα 2.9 Χρόνος απόκρισης αιτήματος μεταξύ χρήστη και server με την χρήση CDN

Στο συγκεκριμένο σημείο θα χρειαστεί να αναφέρουμε στο πως λειτουργεί ένα CDN. Για να ελαχιστοποιηθεί η απόσταση μεταξύ των επισκεπτών και του διακομιστή του ιστότοπου σας, ένα CDN αποθηκεύει μια κρυφή έκδοση του περιεχομένου του σε πολλές γεωγραφικές τοποθεσίες (γνωστός και ως σημεία παρουσίας ή PoP). Κάθε PoP περιέχει έναν αριθμό διακομιστών προσωρινής αποθήκευσης που είναι υπεύθυνοι για την παράδοση περιεχομένου στους επισκέπτες που βρίσκονται κοντά του. Στην ουσία, το CDN τοποθετεί το περιεχόμενό σας σε πολλά σημεία ταυτόχρονα, παρέχοντας ανώτερη κάλυψη στους χρήστες σας. Για παράδειγμα, όταν κάποιος στο Λονδίνο έχει πρόσβαση στον ιστότοπο σας που φιλοξενείται στις ΗΠΑ, αυτό γίνεται μέσω ενός τοπικού PoP του Ηνωμένου Βασιλείου. Αυτό είναι πολύ πιο γρήγορο από το να ταξιδεύουν τα αιτήματα των επισκεπτών και οι απαντήσεις σας σε όλο το πλάτος του Ατλαντικού και πίσω. Αυτός ενδεικτικά είναι

ο τρόπος με τον οποίο δουλεύουν τα CDN. Σήμερα, πάνω από το ήμισυ της επισκεψιμότητας εξυπηρετείται ήδη από CDN. Αυτοί οι αριθμοί έχουν ραγδαία ανοδική τάση κάθε χρόνο που περνάει.



Σχήμα 2.10 Διάγραμμα χρήσης CDN

2.5.1 Τύποι CDN

Γενικά υπάρχουν τέσσερις τύποι CDN

- Δημόσια CDN που προσφέρονται ως υπηρεσία διανομής περιεχομένου μπορούν να χωριστεί σε δύο υποομάδες
    1. Βαθειά Εισχώρηση (Enter-Deep Architecture CDN)

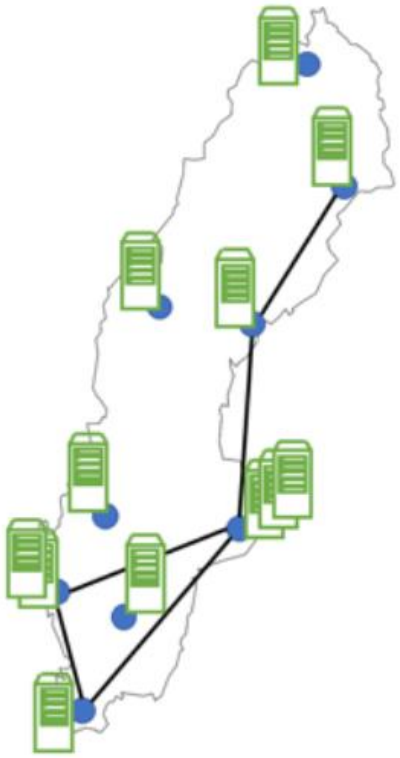
2. Super-POP Architecture CDN

- Ιδιόκτητα CDN (Proprietary CDN)
- Υβριδικά CDN (Hybrid CDN)

Βαθειά Εισχώρηση (Enter-Deep Architecture CDN)

Η αρχιτεκτονική Enter-Deep βασίζεται στη διανομή των CDN pops όσο το δυνατόν πιο κοντά στους πελάτες. Οι POP συνήθως τοποθετούνται πιο έξω στα δίκτυα πρόσβασης, επιτρέποντας πολύ χαμηλό χρόνο μετ' επιστροφής στους πελάτες. Η απόδοση είναι συνήθως πολύ καλή και η ανθεκτικότητα στις επιθέσεις D-DOS είναι μεγάλη με πολλά POP. Από επιχειρηματική άποψη, αυτή η προσέγγιση απαιτεί καλή συνεργασία μεταξύ του CDN και των παρόχων παροχής υπηρεσιών Internet που κατέχουν τα δίκτυα πρόσβασης και επιτρέπουν την τοποθέτηση υλικού CDNPOP σε εγκαταστάσεις ISP.

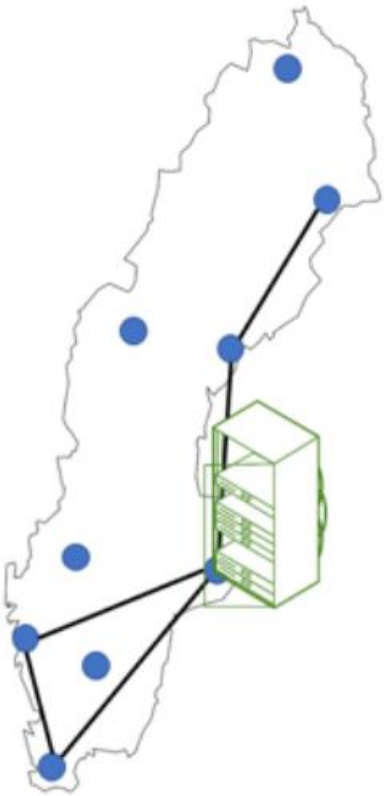
Τα πλεονεκτήματα με χαμηλότερο λανθάνοντα χρόνο και απόδοση συνήθως αντισταθμίζονται όταν διατηρείται ο υψηλός αριθμός POP. Η διάδοση της διαμόρφωσης και των αλλαγών μπορεί να είναι δύσκολη. Επίσης, λόγω του μεγάλου αριθμού POP, είναι πολύ ακριβό για κάθε POP να διατηρεί μια μεγάλη κρυφή μνήμη ή ακόμα και όλες τις λειτουργίες. Οι μικρότερες κρυφές μνήμες δημιουργούν μεγαλύτερο αριθμό παραλείψεων προσωρινής μνήμης (cache) και έτσι δημιουργούνται λήψεις από άλλα POP στο δίκτυο CDN.



Σχήμα 2.11 Αρχιτεκτονική Enter-deep με πολλαπλά μικρότερα POP κατανεμημένα σε μια γεωγραφική περιοχή και δίκτυα ISP

Super-POP Architecture CDN

Η αρχιτεκτονική super-POP στοχεύει στην έξυπνη τοποθέτηση λιγότερων αλλά πολύ μεγαλύτερων POP σε στρατηγικές τοποθεσίες, όπως σημεία ανταλλαγής ή μεγάλα σημεία peering. Αυτά τα POP υψηλής απόδοσης είναι πολύ καλά συνδεδεμένα με συνδέσμους υψηλού εύρους ζώνης σε πολλαπλούς παρόχους υπηρεσιών Διαδικτύου και ως εκ τούτου μπορούν να επιτύχουν καλή απόδοση έναντι των πελατών. Με λίγα POP, οι διαδόσεις συντήρησης και διαμόρφωσης είναι μάλλον αποτελεσματικές. Επίσης, τα μεγαλύτερα POP διασφαλίζουν ότι οι ελλείψεις της προσωρινής μνήμης διατηρούνται σε χαμηλά επίπεδα. Ωστόσο, λιγότερα POP απαιτούν προσεκτικά σχεδιασμένες τοποθεσίες POP για βέλτιστη απόδοση σύνδεσης και συμφωνίες ομότιμης επικοινωνίας (peering agreements) που φθάνουν αποτελεσματικά σε οποιονδήποτε πελάτη στην περιοχή. Και αν κάποιο από τα λίγα POP αποτύχει, οι πελάτες θα κατευθυνθούν σε άλλους POP συνήθως πολύ πιο μακριά από τον αποτυχημένο, δημιουργώντας πιθανώς υψηλότερο λανθάνοντα χρόνο



Σχήμα 2.12 Αρχιτεκτονική Super POP με ένα μεγάλο POP καλά συνδεδεμένο σε ολόκληρη την περιοχή

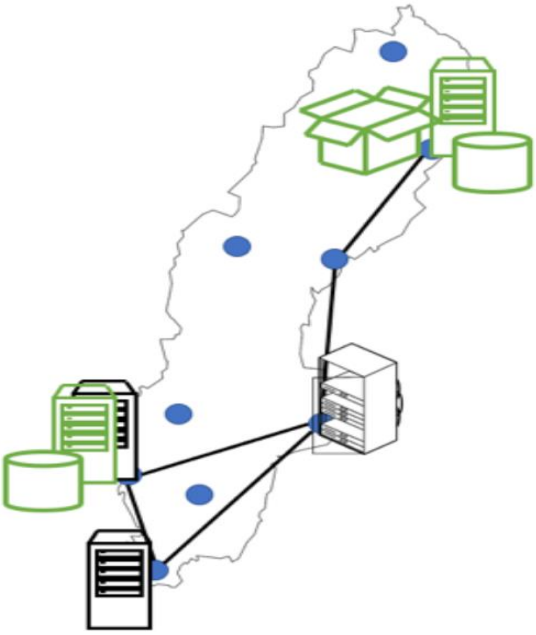
Ιδιόκτητα CDN (Proprietary CDN)

Τα ιδιόκτητα CDN δημιουργούνται αποκλειστικά από και για ιδιοκτήτες δικτύων, όπως χειριστές, παρόχους υπηρεσιών Internet, κ.λπ., και διαχειρίζονται μόνο τη δική τους κίνηση. Οι POP συνήθως βρίσκονται στα δίκτυα του ιδιοκτήτη ή στις νοικιασμένες τοποθεσίες POP. Χτίζοντας δικά του CDN, ο ιδιοκτήτης είναι ελεύθερος να επιλέξει οποιεσδήποτε τεχνολογίες και εργαλεία για την αντιμετώπιση τυχόν προϋποθέσεων δικτύου ή επιχειρηματικών προκλήσεων. Το ιδιόκτητο CDN μπορεί να έχει τέλειες διαστάσεις και να κατασκευαστεί για την ομάδα-στόχο. Ο συνολικός έλεγχος CDN, η πλήρης πρόσβαση σε όλα τα στατιστικά στοιχεία και τα δεδομένα χρήσης και ο κίνδυνος συμφόρησης CDN από άλλους χρήστες, είναι μερικά από τα πλεονεκτήματα των ιδιόκτητων CDN.

Η δημιουργία ιδιωτικών CDN απαιτεί πολλές τεχνικές γνώσεις και συνεχή εργασία για να διασφαλιστεί ότι προετοιμάζονται τυχόν διακυμάνσεις της κυκλοφορίας. Και οι κορυφές κυκλοφορίας μπορεί να γίνουν εφιάλτης επεκτασιμότητας, καθώς έρχονται σπάνια, αλλά απαιτούν εκ των προτέρων επενδύσεις και προετοιμασία.

Υβριδικά CDN (Hybrid CDN)

Η βιομηχανία ροής πολυμέσων βασίζεται συχνά σε πολλαπλά CDN (βλ. τελευταίο άρθρο) ή ακόμα και σε υβριδικά CDN. Ένα υβριδικό CDN είναι απλώς ένας συνδυασμός δημοσίων CDN και ενός self-built CDN και επιτρέπει έναν τέλειο συνδυασμό ελέγχου, κόστους και απόδοσης. Τα υβριδικά CDN μπορούν να χρησιμοποιηθούν για διάφορες βελτιώσεις στη διανομή περιεχομένου



Σχήμα 2.13 Hybrid CDN — Δημόσιο CDN ενισχυμένο με πρόσθετες τοποθεσίες και χώρο αποθήκευσης

2.5.2 Στρατηγικές Επιλογής Συνεργατικής Ομάδας (cluster selection strategy)

Στον πυρήνα της υλοποίησης οποιουδήποτε CDN βρίσκεται μία στρατηγική βρίσκεται μια στρατηγική επιλογής συνεργατικής ομάδας (cluster selection strategy), δηλαδή ένας μηχανισμός για την δυναμική

κατεύθυνση πελατών προς μια συνεργατικά ομάδα εξυπηρετών ή ένα κέντρο δεδομένων, μέσα στο CDN. Το CDN μαθαίνει την διεύθυνση IP του εξυπηρέτη LDNS του πελάτη, μέσω της αναζήτησης του DNS του πελάτη. Μετά την εκμάθηση της διεύθυνσης IP, το CDN πρέπει να επιλέξει μία κατάλληλη συνεργατική ομάδα με βάση αυτήν την διεύθυνση IP. Τα CDN γενικά χρησιμοποιούν ιδιοταγείς στρατηγικές επιλογής συμπλέγματος.

Μία απλή στρατηγική είναι να εκχωρήσετε τον πελάτη στην συνεργατική ομάδα, η οποία είναι γεωγραφικά πλησιέστερη. Χρησιμοποιώντας εμπορικά διαθέσιμες βάσεις δεδομένων γεωγραφικού εντοπισμού κάθε διεύθυνση IPLDNS αντιστοιχίζεται σε μια γεωγραφική τοποθεσία. Όταν λαμβάνεται μία αίτηση DNS από ένα συγκεκριμένο LDNS, το CDN επιλέγει την γεωγραφικά πλησιέστερη συνεργατική ομάδα, δηλαδή την ομάδα που απέχει τα λιγότερα χιλιόμετρα από τον LDNS σε "ευθεία γραμμή". Μία τέτοια λύση μπορεί να εργασθεί αρκετά καλά για ένα μεγάλο τμήμα πελατών. Ωστόσο, για ορισμένους πελάτες η λύση μπορεί να μην εργασθεί καλά, επειδή η γεωγραφικά πλησιέστερη συνεργατική ομάδα μπορεί να μην είναι η πλησιέστερη συνεργατική ομάδα κατά μήκος της διαδρομής του δικτύου. Ακόμη, ένα πρόβλημα που είναι εγγενές σε όλες τις προσεγγίσεις, που βασίζονται στο DNS είναι ότι ορισμένοι τελικού χρήστες είναι διαμορφωμένοι έτσι, ώστε να χρησιμοποιούν απομακρυσμένα τοποθετημένους LDNS, οπότε η θέση του LDNS μπορεί να βρίσκεται μακριά από την θέση του πελάτη. Επιπλέον, αυτή η απλή στρατηγική αγνοεί τις μεταβολές στην καθυστέρηση και στο εύρος ζώνης στις διαδρομές του διαδικτύου με το πέρασμα του χρόνου, εκχωρώντας πάντα την ίδια συνεργατική ομάδα σε έναν συγκεκριμένο πελάτη.

Για να προσδιορίσουν την βέλτιστη συνεργατική ομάδα για ένα πελάτη, με βάση τις τρέχουσες συνθήκες κίνησης, τα CDN μπορούν, αντί της προηγούμενης στρατηγικής, να εκτελούν περιοδικές μετρήσεις σε πραγματικό χρόνο της καθυστέρησης και της απώλειας απόδοσης ανάμεσα στις συνεργατικές ομάδες και στους πελάτες τους. Για παράδειγμα ένα CDN μπορεί να κάνει κάθε συνεργατική ομάδα του να στέλνει περιοδικά βολιδοσκοπήσεις (π.χ. με ping ή DNS requests) σε όλους τους LDNS σε όλο τον κόσμο. Ένα μειονέκτημα αυτής της προσέγγισης είναι ότι πολλοί LDNS διαμορφώνονται ώστε να μην αποκρίνονται σε τέτοιες βολιδοσκοπήσεις.

Μία εναλλακτική λύση, αντί της αποστολής πρόσθετης κίνησης για την μέτρησης των ιδιοτήτων διαδρομών είναι να χρησιμοποιήσουμε τα χαρακτηριστικά της πρόσφατης κίνησης και της κίνησης σε εξέλιξη, ανάμεσα στους πελάτες και στους εξυπηρέτες του CDN. Τέτοιες λύσεις βέβαια απαιτούν μια ανακατεύθυνση των πελατών σε πιθανώς υπό-βέλτιστες λύσεις από καιρό σε καιρό για να μετρηθούν οι ιδιότητες των διαδρομών προς αυτές τις συνεργατικές ομάδες. Αν και χρειάζεται μόνο ένας μικρός αριθμός αιτήσεων, που θα λειτουργήσουν ως βολιδοσκοπήσεις, οι επιλεγμένοι πελάτες μπορούν να υποστούν σημαντική υποβάθμιση της ποιότητας όταν λαμβάνουν περιεχόμενο (βίντεο ή άλλο). Μία άλλη εναλλακτική προσέγγιση για βολιδοσκόπηση της διαδρομής από μία συνεργατική ομάδα προς έναν πελάτη είναι να χρησιμοποιηθεί κίνηση ερωτημάτων DNS ώστε να μετρηθεί η καθυστέρηση ανάμεσα σε πελάτες και συνεργατικές ομάδες. Συγκεκριμένα, κατά την διάρκεια της φάσης DNS ο LDNS του πελάτη μπορεί περιστασιακά να κατευθύνεται σε διαφορετικούς αυθεντικούς εξυπηρέτες DNS, που είναι εγκατεστημένη σε διάφορες τοποθεσίες συνεργατικών ομάδων, παράγοντας κίνησης DNS, που μπορεί κατόπιν να μετρηθεί ανάμεσα στον LDNS και σε αυτές τις τοποθεσίες συνεργατικών ομάδων.

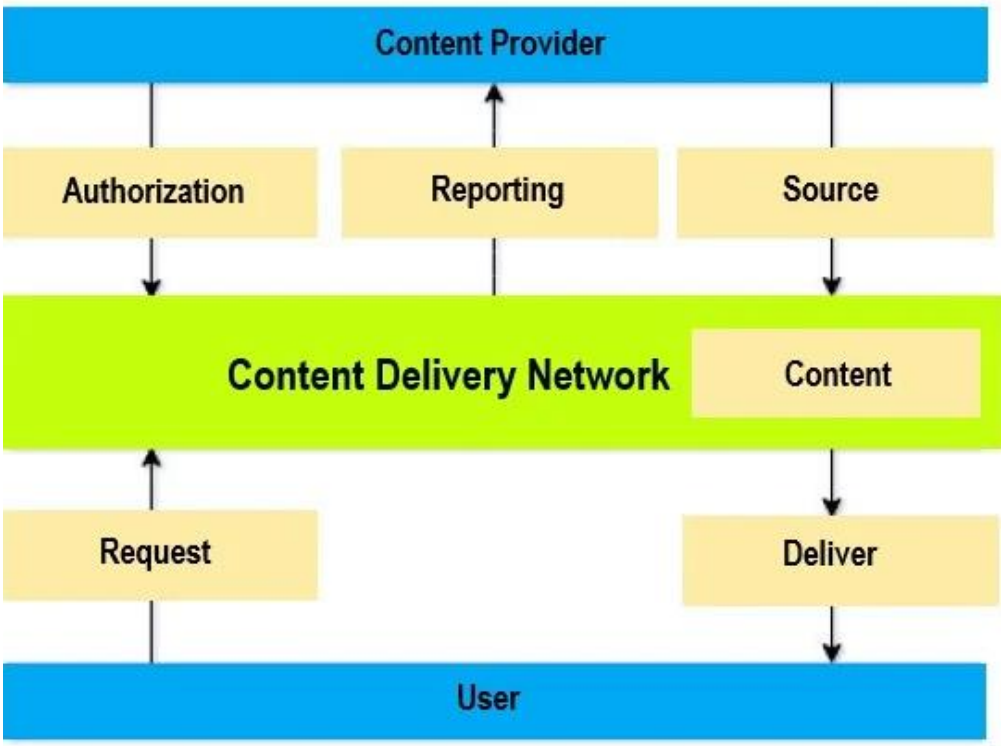
Μία τελείως διαφορετική προσέγγιση για το ταίριασμα πελατών με εξυπηρέτες CDN είναι να χρησιμοποιηθεί IP anycast. Η ιδέα πίσω από το IP anycast είναι να κάνει τους δρομολογητές μέσα στο διαδίκτυο να δρομολογούν τα πακέτα του πελάτη στην πλησιέστερη συνεργατική ομάδα. Όταν ένας πελάτης θέλει να δει ένα βίντεο, το DNS του CDN επιστρέφει την διεύθυνση anycast, ανεξάρτητα από το που βρίσκεται ο πελάτης. Αυτή προσέγγιση έχει το πλεονέκτημα της εύρεσης της συνεργατικής

ομάδας που είναι πλησιέστερα προς τον πελάτη και όχι της συνεργατικής ομάδας που είναι πλησιέστερα προς τον LDNS του πελάτη.

Πετά των δικτυακών θεμάτων, όπως είναι η καθυστέρηση , η απώλειας και η απόδοση εύρους ζώνης υπάρχουν πολλοί άλλοι πρόσθετοι σημαντικοί παράγοντες που έρχονται στην σχεδίασης μία στρατηγικής επιλογής συνεργατικής ομάδας. Ο φόρτος στης συνεργατικές ομάδες είναι ένας τέτοιος παράγοντας. Οι πελάτες δεν πρέπει να κατευθύνονται προς υπερφορτωμένες συνεργατικές ομάδες.

2.6 Επίλογος

Όπως μπορούμε να αντιληφθούμε η λειτουργία του streaming είναι άρρητα συνδεδεμένη με τα προαναφερθέντα πρωτόκολλα. Επίσης χρήση CDN στην σημερινή εποχή για πλατφόρμες και εφαρμογές που σχετίζονται με το streaming είναι κάτι παραπάνω από αναγκαία. Για αυτό τον λόγο έχει υιοθετηθεί από όλες τις μεγάλες εταιρίες που παρέχουν streaming υπηρεσίες (π.χ. Netflix, Amazon, YouTube κτλ).



Σχήμα 2.14 Αρχιτεκτονική CDN και βασικά στοιχεία

Κεφάλαιο 3ο:Ανάλυσηαρχιτεκτονικής για την δημιουργία της εφαρμογής και αρχές σχεδιασμού software

3.1 Πρόλογος

Στο συγκεκριμένο κεφάλαιο θα γίνει μια αναφορά σχετικά με την μεθοδολογία και τις αρχιτεκτονικές που βασίστηκε η δημιουργία και σχεδίαση της πτυχιακής εργασίας. Πιο συγκεκριμένα θα γίνει ανάλυση της αρχιτεκτονικής MVC, του μοτίβου Lazy Loading, των Dependency Injections καθώς και των Decorator Patterns.

3.2Αρχιτεκτονική Model–view–controller (MVC)

Το MVC είναι γνωστό σαν ένα αρχιτεκτονικό μοτίβο, το οποίο ενσωματώνει τρία μέρη Model, View και Controller, ή για να είμαστε πιο ακριβείς χωρίζει την εφαρμογή σε τρία λογικά μέρη: το Model, το View και τον Controller. Χρησιμοποιήθηκε στην αρχή για graphical user interfaces σε desktop, αλλά σήμερα χρησιμοποιείται στη σχεδίαση εφαρμογών για κινητά και διαδικτυακών εφαρμογών.

Πριν αναλύσουμε όμως την συγκεκριμένη αρχιτεκτονική, αξίζει να αναφέρουμε κάποια ιστορικά γεγονότα σχετικά με την προέλευση της και τα προβλήματα που προσπάθησε να επιλύσει. Ο Trygve Reenskaug ήταν ο εφευρέτης του MVC. Οι πρώτες αναφορές για το MVC γράφτηκαν όταν επισκεπτόταν έναν επιστήμονα στο Xerox Palo Alto Research Laboratory (PARC) το 1978/79. Στην αρχή, το MVC ονομαζόταν "Thing Model View Editor" αλλά το άλλαξε γρήγορα σε "Model View Controller". Ο στόχος του Tygrve ήταν να λύσει το πρόβλημα των χρηστών που ελέγχουν ένα μεγάλο και πολύπλοκο σύνολο δεδομένων. Η πρακτική του MVC έχει αλλάξει με τα χρόνια. Δεδομένου ότι το μοτίβο MVC επινοήθηκε πριν από τα προγράμματα περιήγησης Ιστού, αρχικά χρησιμοποιήθηκε ως αρχιτεκτονικό μοτίβο για γραφικές διεπαφές χρήστη (GUI). Στην σημερινή εποχή, το MVC χρησιμοποιείται για το σχεδιασμό εφαρμογών Ιστού. Ορισμένα πλαίσια (framework) ιστού που χρησιμοποιούν την έννοια MVC: Ruby on Rails, Laravel, Zend framework, CherryPy, Symphony κ.λπ.

Τα τρία μέρη της αρχιτεκτονικής MVC μπορούν να περιγραφούν ως εξής:

- Model: Διαχειρίζεται δεδομένα και επιχειρηματική λογική.
- View: Χειρίζεται τη διάταξη και την εμφάνιση.
- Controller: Δρομολογεί εντολές στο model και στα views

Model

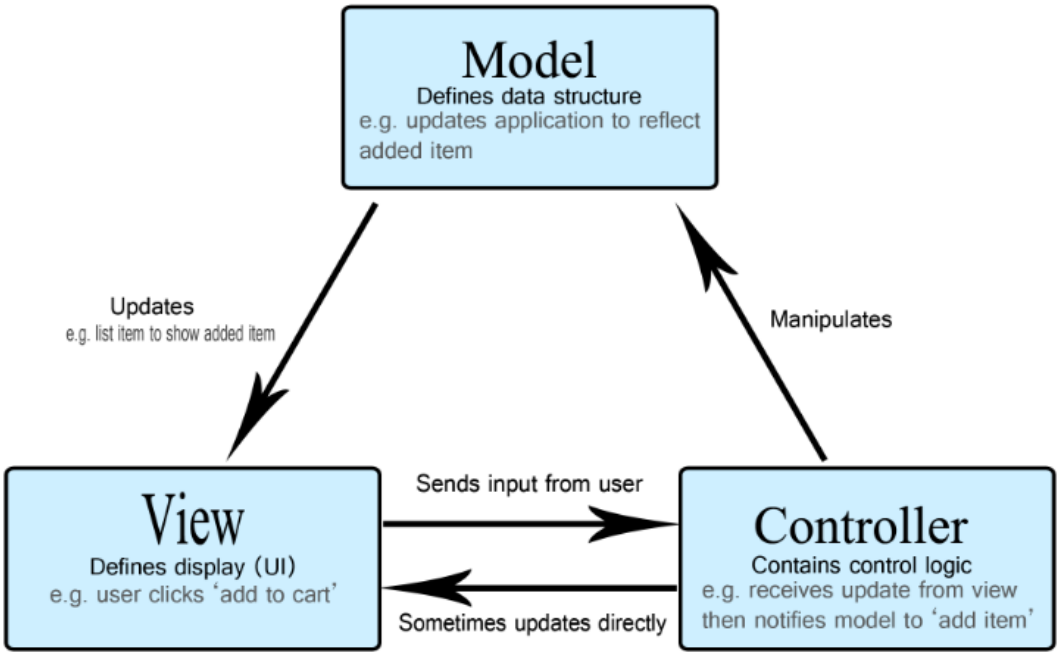
Είναι γνωστό ως το χαμηλότερο επίπεδο που σημαίνει ότι είναι υπεύθυνο για τη διατήρηση δεδομένων. Χειρίζεται τα δεδομένα με λογική, οπότε ασχολείται βασικά με δεδομένα. Το μοντέλο είναι στην πραγματικότητα συνδεδεμένο με τη βάση δεδομένων, για οτιδήποτε έχει σχέση με τα δεδομένα. Η προσθήκη ή η ανάκτηση δεδομένων πραγματοποιείται από το model. Απαντά στα αιτήματα του controller επειδή ο controller δεν μιλά ποτέ από μόνος του στη βάση δεδομένων. Το model συνομιλεί με τη βάση δεδομένων εμπρός και πίσω και στη συνέχεια δίνει τα απαραίτητα δεδομένα στον controller. Σημείωση: το model δεν επικοινωνεί ποτέ απευθείας με τα views.

View

Η αναπαράσταση δεδομένων γίνεται από toview. Στην πραγματικότητα δημιουργεί UI ή διεπαφή χρήστη για τον χρήστη. Έτσι, σε εφαρμογές web, όταν σκεφτόμαστε το στοιχείο view, αναφερόμαστε απλώς το τμήμα Html/CSS. Τα views δημιουργούνται από τα δεδομένα που συλλέγονται από το model, αλλά αυτά τα δεδομένα δεν λαμβάνονται απευθείας αλλά μέσω του controller, επομένως το view μιλάει μόνο στον controller.

Controller

Είναι γνωστό ως ο κύριος πρωταγωνιστής επειδή ο controller είναι το στοιχείο που επιτρέπει τη διασύνδεση μεταξύ των views και των models, επομένως λειτουργεί ως ενδιάμεσος. Ο controller δεν χρειάζεται να ανησυχεί για τον χειρισμό της λογικής δεδομένων, απλώς λέει στο model τι να κάνει. Αφού λάβει δεδομένα από το model, τα επεξεργάζεται και στη συνέχεια παίρνει όλες αυτές τις πληροφορίες που τις στέλνει στο view και εξηγεί πώς να τις αναπαραστήσει στον χρήστη. Σημείωση: Τα views και τα models δεν μπορούν να μιλήσουν απευθείας.



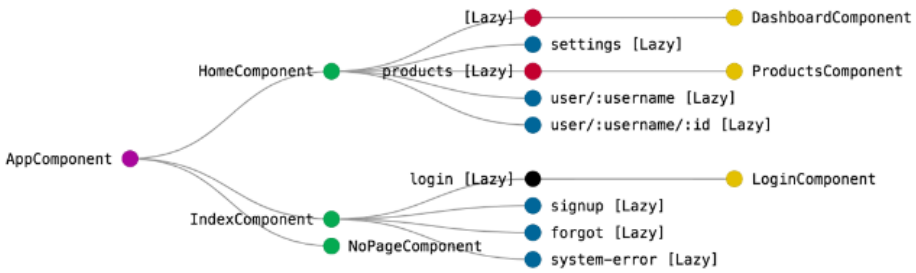
Σχήμα 3.1 Παράδειγμα αρχιτεκτονικής MVC

3.3 Lazy loading

Το Lazy loading είναι ένα μοτίβο σχεδιασμού για την υλοποίηση έργων Angular μεγάλης κλίμακας. Χρησιμοποιώντας αυτό το σχέδιο συστήματος μπορούμε να φορτώσουμε τα στοιχεία με βάση τις ανάγκες της εφαρμογής. Με αυτόν τον τρόπο μπορούμε να αυξήσουμε την απόδοση της εφαρμογής. Είναι μια εξαιρετική στρατηγική για να μειώσουμε το χρόνο στη διάδραση (TTI) μιας εφαρμογής μιας σελίδας (SPA) και έτσι να παρέχουμε καλύτερη εμπειρία χρήστη. Το TTI είναι μια μέτρηση που αναπτύχθηκε από την Google και μετρά το χρονικό διάστημα μεταξύ του σημείου στο οποίο έχουν φορτωθεί τα στοιχεία της σελίδας και του σημείου στο οποίο αυτά τα στοιχεία γίνονται αλληλεπιδραστικά (χρήσιμα).

Σχετικά με τον τρόπο λειτουργίας του, όταν ένας χρήστης πλοηγείται σε μια ιστοσελίδα που εφαρμόζει lazy loading, υποβάλλεται ένα αίτημα στον κεντρικό διακομιστή και όλα τα πακέτα, ή κομμάτια, αποστέλλονται ως απόκριση και αποθηκεύονται στην κρυφή μνήμη του προγράμματος περιήγησης, συμπεριλαμβανομένων των lazy loaded modules. Έπειτα, με εξαίρεση τα lazy loaded modules, τα πακέτα συναρμολογούνται, αναλύονται και φορτώνονται, αποδίδοντας τη σελίδα. Όταν απαιτείται περιεχόμενο με αργή φόρτωση για την απόδοση της σελίδας, το πρόγραμμα περιήγησης υποβάλλει ένα αίτημα στην προσωρινή μνήμη του προγράμματος περιήγησης για την απαραίτητη ενότητα και τη φορτώνει.

Αναφορικά με την εφαρμογή του, σε υψηλό επίπεδο, διαμορφώνετε μια γονική διαδρομή που δείχνει σε κάθε λειτουργική μονάδα στη ρίζα σας. Στη συνέχεια, μέσα σε κάθε λειτουργική λειτουργική μονάδα που έχει φορτωθεί αργά, συσχετίζετε θυγατρικές διαδρομές με στοιχεία που θέλετε να αποδώσετε κατά τη φόρτωση της λειτουργικής μονάδας. Όταν χρειάζεται ένα από αυτά, ο δρομολογητής (router) Angular το φορτώνει.



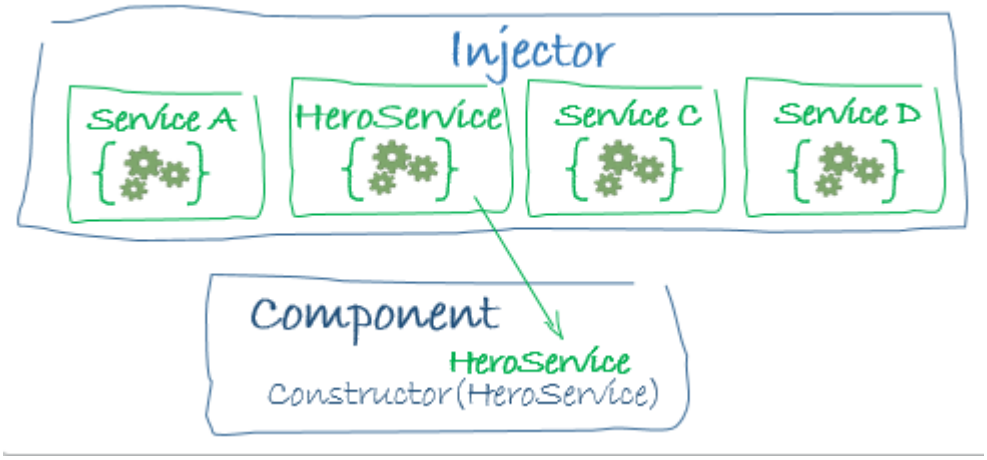
Σχήμα 3.2 Παράδειγμα χρήσης Lazy Loading όπου φορτώνονται κάθε φορά μόνο τα αναγκαία components

3.4 Dependency injection

Οι εξαρτήσεις (depedencies) στην Angular είναι υπηρεσίες (services) ή αντικείμενα που χρειάζεται μια κλάση για να εκτελέσει τη λειτουργία της. Το Dependency injection ή DI, είναι ένα μοτίβο σχεδίασης στο οποίο μια κλάση ζητά εξαρτήσεις από εξωτερικές πηγές αντί να τις δημιουργεί. Το

πλαίσιο DI της Angular παρέχει εξαρτήσεις σε μια κλάση κατά την εγκατάσταση. Χρησιμοποιούμε το AngularDI για να αυξήσουμε την ευελιξία και την αρθρώτητα στις εφαρμογές σας. Ας υποθέσουμε δύο κλάσεις, την A και τη B. Ας υποθέσουμε ότι η κλάση A χρησιμοποιεί τα αντικείμενα της κλάσης B. Κανονικά δημιουργείται μια παρουσία της κλάσης B έτσι ώστε η κλάση A να έχει πρόσβαση στα αντικείμενα. Χρησιμοποιώντας DI, μετακινούμε τη δημιουργία και τη σύνδεση των εξαρτημένων αντικειμένων εκτός της κλάσης που εξαρτώνται από αυτά.

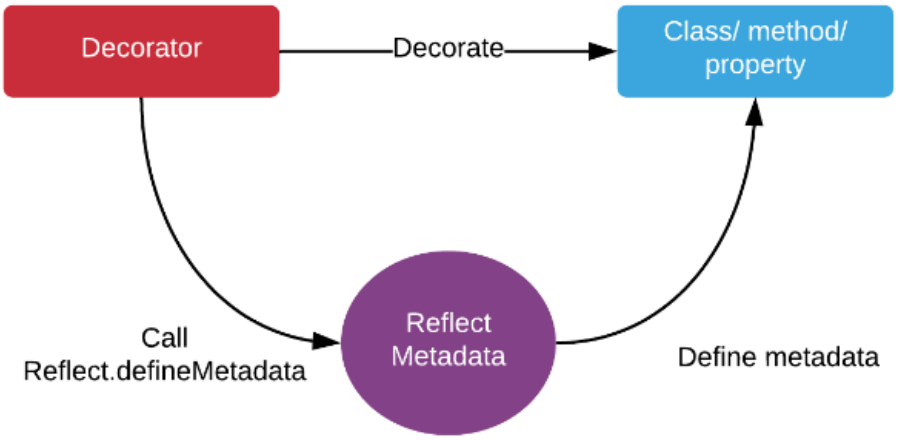
Το service είναι μια ευρεία κατηγορία που περιλαμβάνει οποιαδήποτε τιμή, λειτουργία ή χαρακτηριστικό χρειάζεται μια εφαρμογή. Ένα service είναι συνήθως μια κλάση με έναν στενό, καλά καθορισμένο σκοπό. Θα πρέπει να κάνει κάτι συγκεκριμένο και να το κάνει καλά. Η Angular διακρίνει τα components από τα services για να αυξήσει την αρθρώτητα και την επαναχρησιμοποίηση τους. Διαχωρίζοντας τη λειτουργικότητα που σχετίζεται με την προβολή ενός component από άλλα είδη επεξεργασίας, μπορούμε να κάνουμε τα component classes πιο λιτά και αποτελεσματικά Η Angular δεν επιβάλλει αυτά που αναλύσαμε παραπάνω. Η Angular μας βοηθά να ακολουθήσουμε αυτές τις αυτές τις αρχές κάνοντας εύκολο να συνυπολογίσουμε τη λογική της εφαρμογής μας σε services και να καταστήσουμε αυτά τα services διαθέσιμα στα components μέσω του Dependency injection.



Σχήμα 3.3 Η Angular καλεί τον constructor του component με αυτά τα services (π.χ. HeroService) ως ορίσματα

3.5 Decorator pattern

Οι decorators είναι ένα σχέδιο σχεδίασης που χρησιμοποιείται για να διαχωρίσει την τροποποίηση ή τη διακόσμηση μιας κλάσης χωρίς να τροποποιήσει τον αρχικό πηγαίο κώδικα. Για να το θέσουμε με απλά λόγια, με τους decorators, μπορούμε να διαμορφώσουμε και να προσαρμόσουμε τις κλάσεις μας την ώρα του σχεδιασμού. Είναι απλώς συναρτήσεις που μπορούν να χρησιμοποιηθούν για την προσθήκη μεταδεδομένων, ιδιοτήτων ή συναρτήσεων στο αντικείμενο με το οποίο συνδέονται. Οι συναρτήσεις των decorators καλούνται με ένα σύμβολο @ με πρόθεμα και ακολουθούνται αμέσως από μια κλάση, μια παράμετρο, μια μέθοδο ή μια ιδιότητα. Η συνάρτηση διακοσμητής παρέχει πληροφορίες σχετικά με την κλάση, την παράμετρο ή τη μέθοδο και η συνάρτηση διακοσμητής επιστρέφει κάτι στη θέση της ή χειρίζεται τον στόχο της με κάποιο τρόπο.



Σχήμα 3.4 Δομή ενός decorator

Τύποι Decorators

Στην Angular υπάρχουν διάφοροι τύποι decoretators.

- Class decorators, π.χ. @Componentκαι @NgModule
- Property decorators για properties εντόςκλάσεων, π.χ. @Input και @Output
- Method decorators για μεθόδους εντός κλάσεων, π.χ. @HostListener
- Parameter decorators για παραμέτρους εντός κλάσεων constructors, π.χ. @Inject

Class decorators

Οι Class decorators, γνωστοί και ως decorators ανώτατου επιπέδου, χρησιμοποιούνται για να ενημερώσουν την Angular ότι μια συγκεκριμένη κατηγορία (class) είναι ένα component ή ένα module. Οι decorators ανώτατου επιπέδου περιλαμβάνουν τα @Component και @NgModule.

Property decorators

Η χρήση Property decorators προκύπτει όταν απαιτείται επικοινωνία components. Οι Property decorators διαχειρίζονται την επικοινωνία μεταξύ γονικών και παιδιών component. Υπάρχουν διάφοροι τύποι Property decorators. Μερικά από τα πιο συχνά χρησιμοποιούμενα είναι:

1. @Input
2. @Output
3. @ViewChild

Method decorators

Οι Method decorators επιτρέπουν τη χρήση ειδικών λειτουργιών decorator. Για παράδειγμα, ο decorator @hostListener παρατηρεί συμβάντα και μας επιτρέπει να πούμε στην Angular να καλέσει τη μέθοδο decorator όταν ενεργοποιείται.

Parameter decorators

Οι Parameter decorators χρησιμοποιούνται όταν πρέπει να πούμε στην Angular να κάνει inject σε έναν συγκεκριμένο πάροχο σε έναν constructor.

3.6 Αρχές σχεδιασμού

SOLID

Το SOLID είναι ένα δημοφιλές σύνολο αρχών σχεδιασμού που χρησιμοποιούνται στην ανάπτυξη αντικειμενοστρεφούς λογισμικού. Το SOLID είναι ένα ακρωνύμιο που αντιπροσωπεύει πέντε βασικές αρχές σχεδιασμού: single responsibility, open-closed, Liskov substitution, interface segregation and dependency inversion.Οι αρχές SOLID αναπτύχθηκαν από τον RobertC. Martin το 2000.Ο γενικός στόχος των αρχών SOLID είναι να μειώσουν τις εξαρτήσεις έτσι ώστε οι μηχανικοί να μπορούν να αλλάξουν ένα κομμάτι λογισμικού χωρίς να επηρεάσουν άλλο. Επιπλέον, προορίζονται να κάνουν τα σχέδια πιο κατανοητά, διατηρημένα και επεκτατικά. Τελικά, η χρήση αυτών των αρχών σχεδιασμού διευκολύνει τους μηχανικούς λογισμικού να αποφύγουν προβλήματα και να δημιουργήσουν προσαρμοστικό, αποτελεσματικό και ευέλικτο λογισμικό. Ενώ οι αρχές έχουν πολλά πλεονεκτήματα, η τήρηση των αρχών οδηγεί γενικά στη σύνταξη μεγαλύτερου και πιο περίπλοκου κώδικα. Αυτό σημαίνει ότι μπορεί να επεκτείνει τη διαδικασία σχεδιασμού και να κάνει την ανάπτυξη λίγο πιο δύσκολη. Ωστόσο, αυτός ο επιπλέον χρόνος και προσπάθεια αξίζει τον κόπο γιατί κάνει το λογισμικό πολύ πιο εύκολο στη συντήρηση, τη δοκιμή και την επέκταση.Οι αρχές έχουν γίνει δημοφιλείς επειδή όταν ακολουθούνται σωστά, οδηγούν σε καλύτερο κώδικα για αναγνωσιμότητα, δυνατότητα συντήρησης, σχέδια σχεδίασης (design patternns) και δοκιμές.

Single Responsibility

Αυτή η αρχή σημαίνει ότι κάθε κλάση κάνει μόνο ένα πράγμα και κάθε κλάση ή module έχει ευθύνη μόνο για ένα μέρος της λειτουργικότητας του λογισμικού. Πιο απλά, κάθε κλάση θα πρέπει να λύνει μόνο ένα πρόβλημα. Η χρήση αυτής της αρχής καθιστά ευκολότερο τον έλεγχο και τη συντήρηση του κώδικα, διευκολύνει την εφαρμογή του λογισμικού και βοηθά στην αποφυγή απρόβλεπτων παρενεργειών μελλοντικών αλλαγών.

Open-Closed

Η ιδέα της αρχής Open-Closed είναι ότι οι υπάρχουσες, καλά δοκιμασμένες κλάσεις θα πρέπει να τροποποιηθούν όταν χρειάζεται να προστεθεί κάτι. Ωστόσο, η αλλαγή κλάσεων μπορεί να οδηγήσει σε προβλήματα ή σφάλματα. Αντί να αλλάξετε την κλάση, απλά θέλουμε να την επεκτείνετε. Η χρήση

κληρονομικότητας ή διεπαφών (interfaces) είναι ένας κοινός τρόπος συμμόρφωσης με αυτήν την αρχή. Ανεξάρτητα από τη μέθοδο που χρησιμοποιείται, είναι σημαντικό να ακολουθήσουμε αυτήν την αρχή για να γράψουμε κώδικα που να μπορεί να διατηρηθεί και να αναθεωρηθεί.

Liskov Substitution

Από τις πέντε solid αρχές, η Liskov Substitution είναι ίσως η πιο δύσκολη στην κατανόηση. Σε γενικές γραμμές, αυτή η αρχή απαιτεί απλώς ότι κάθε παράγωγη κλάση πρέπει να είναι αντικαταστάσιμη για τη μητρική κλάση της. Αν και αυτή μπορεί να είναι μια δύσκολη αρχή να εσωτερικευτεί, με πολλούς τρόπους είναι απλώς μια επέκταση της open-closed, καθώς είναι ένας τρόπος να διασφαλιστεί ότι οι παραγόμενες κλάσεις επεκτείνουν τη βασική κλάση χωρίς να αλλάξουν συμπεριφορά. Η τήρηση αυτής της αρχής βοηθά στην αποφυγή απροσδόκητων συνεπειών αλλαγών και αποφεύγει να χρειάζεται να ανοίξετε μια κλειστή κλάση για να κάνουμε αλλαγές. Οδηγεί σε εύκολες επεκτάσεις λογισμικού και, ενώ μπορεί να επιβραδύνει τη διαδικασία ανάπτυξης, η τήρηση αυτής της αρχής κατά την ανάπτυξη μπορεί να αποφύγει πολλά προβλήματα κατά τις ενημερώσεις και τις επεκτάσεις.

Interface Segregation

Η γενική ιδέα της αρχής Interface Segregation είναι ότι είναι καλύτερο να έχουμε πολλές μικρότερες διεπαφές (interfaces) από μερικές μεγαλύτερες. Αυτό σημαίνει ότι δεν θέλουμε να ξεκινήσουμε απλώς με μια υπάρχουσα διεπαφή και να προσθέσουμε νέες μεθόδους. Αντίθετα, ξεκινήστε δημιουργώντας μια νέα διεπαφή και στη συνέχεια, αφήστε την κλάση σας να εφαρμόσει πολλαπλές διεπαφές όπως απαιτείται. Σύμφωνα με αυτήν την αρχή, οι μηχανικοί πρέπει να εργάζονται για να έχουν πολλές διεπαφές, αποφεύγοντας τον πειρασμό να έχουν μια μεγάλη διεπαφή γενικής χρήσης.

Dependency Inversion

Αυτή η αρχή προσφέρει έναν τρόπο για να επιτυγχάνεται το decouple των modules του λογισμικού. Με απλά λόγια, η αρχή της αντιστροφής της εξάρτησης σημαίνει ότι οι προγραμματιστές θα πρέπει να «εξαρτώνται από abstractions, όχι από concretions». Ένας δημοφιλής τρόπος συμμόρφωσης με αυτήν την αρχή είναι μέσω της χρήσης ενός μοτίβου αντιστροφής εξάρτησης, αν και αυτή η μέθοδος δεν είναι ο μόνος τρόπος για να γίνει αυτό. Όποια μέθοδο κι αν επιλέξετε να χρησιμοποιήσετε, η εύρεση τρόπου χρήσης αυτής της αρχής θα κάνει τον κώδικά σας πιο ευέλικτο, ευέλικτο και επαναχρησιμοποιήσιμο.

3.7 Επίλογος

Μπορούμε να καταλάβουμε λοιπόν ότι το MVC έχει μια πολυπλοκότητα στην κατανόηση του αλλά κάθε προγραμματιστής πρέπει να το έχει υπόψη του όταν αναπτύσσει μια εφαρμογή. Αυτό που χρειάζεται να καταλάβουμε για το MVC είναι ότι πρόκειται για μια αρχιτεκτονική που χωρίζει το λογισμικό μας σε μικρότερα στοιχεία. Το model ασχολείται με τα δεδομένα και τη λογική του συστήματός σας. Τα views εμφανίζουν μόνο δεδομένα και ο controllerδιατηρεί τη σύνδεση μεταξύ του model και του view. Αυτή η «διαίρεση» καθιστά δυνατή την αναγνωσιμότητα και την αρθρωτή δομή, καθώς και διευκολύνει το τις διαδικασίες δοκιμών. Επίσης, η lazy loading είναι ένας τρόπος για τους προγραμματιστές να μειώσουν το TTI, να βελτιώσουν την εμπειρία του χρήστη και να μειώσουν τον χρόνο απόκρισης των εφαρμογών τους. Επίσης, οι Decorators της Angular μας παρέχουν τη δυνατότητα τροποποίησης των κλάσεων κάνοντας τον σχεδιασμό τους πιο συνοπτικό και αποτελεσματικό. Τέλος, είναι πολύ σημαντικός ο τρόπος με τον οποίο πρέπει να οργανώσουμε την συγγραφή του κώδικα μου ώστε να μην αντιμετωπίσουμε δυσκολίες για τυχόν αλλαγές/επεκτάσεις που μπορεί να προκύψουν είτε άμεσα είτε μελλοντικά.

Κεφάλαιο 4ο: Περιγραφή και ανάλυση της εφαρμογής σε επίπεδο χρήσης

4.1 Πρόλογος

Στο συγκεκριμένο κεφάλαιο θα δούμε την ροή της εφαρμογής μας, καθώς και τις διάφορες λειτουργίες που υποστηρίζει είτε για κάποιον χρήστη που είναι συνδεδεμένος είτε για κάποιον που δεν διαθέτει ενεργό κάποιο λογαριασμό και απλά επιθυμεί να περιηγηθεί.

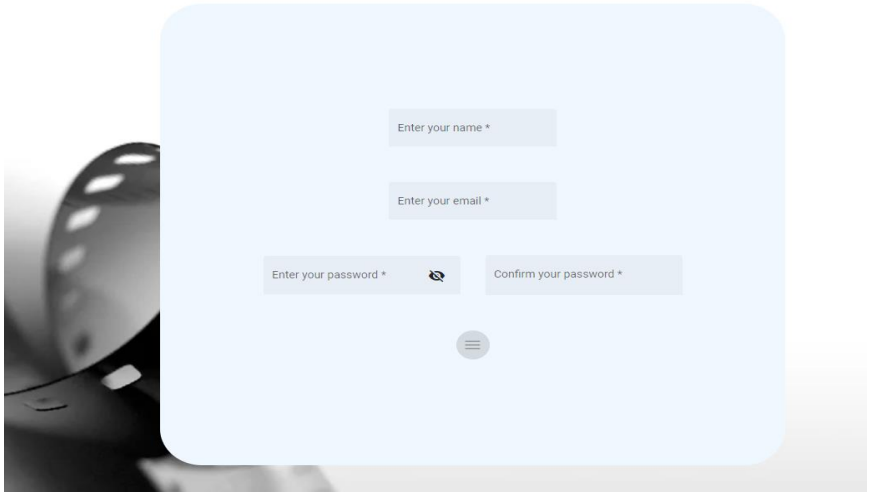
4.2 Λειτουργίες

Αρχικό Menu

Η εφαρμογή ξεκινάει αρχικά φορτώνοντας ένα component της Angular το οποίο εμπεριέχει τρεις βασικές λειτουργίες.

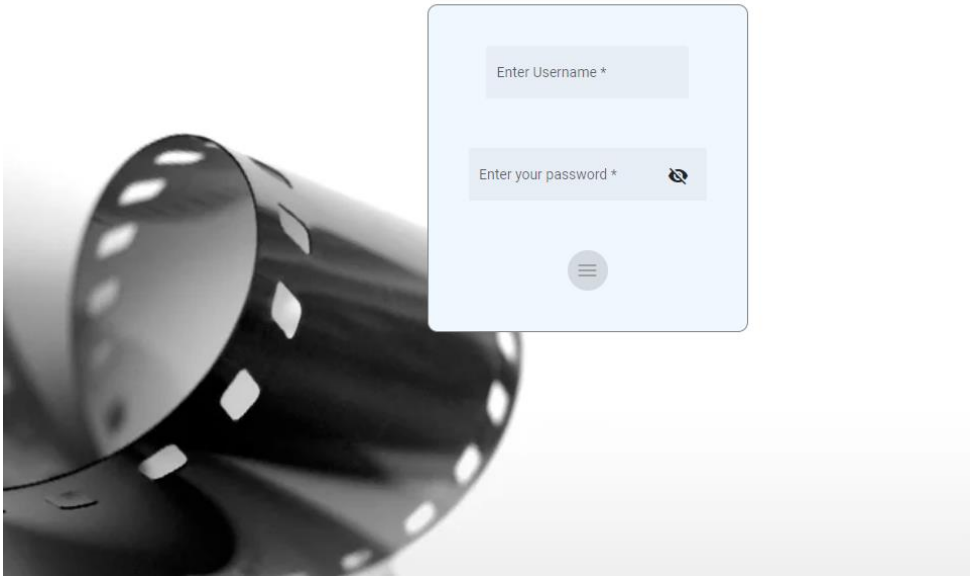
Register

Πρόκειται για μία συνηθισμένη διαδικασία εγγραφής χρήστη, όπου πρέπει να συμπληρωθούν κατάλληλα τα πεδία ονόματος χρήστη, διεύθυνση mail καθώς και ένας κωδικός ο οποίος απαιτεί να διαθέτει ένα τουλάχιστον κεφαλαίο γράμμα, ένα σύμβολο και έναν αριθμό για λόγους ασφαλείας.

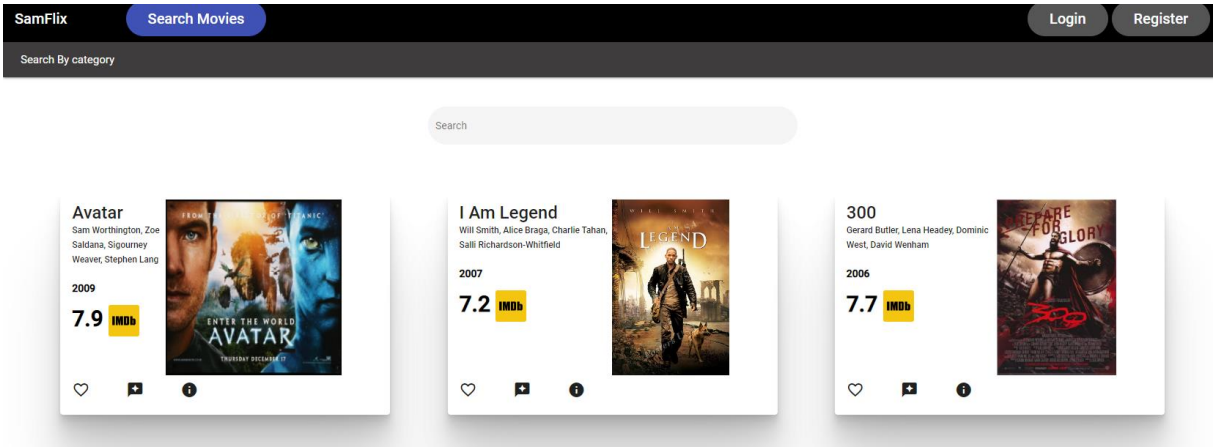


Login

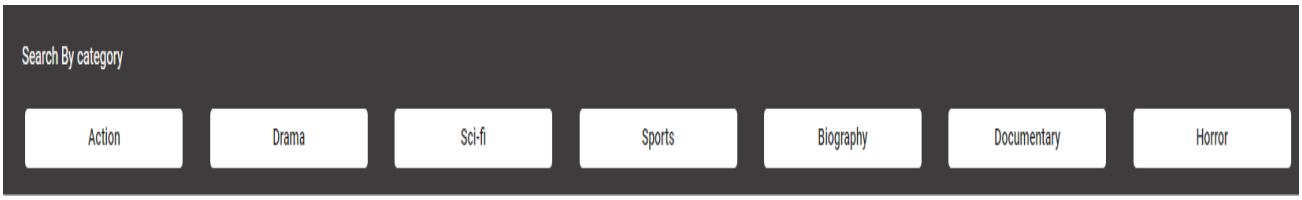
Σε περίπτωση που έχει ολοκληρωθεί επιτυχώς το register, η εφαρμογή κατευθύνει άμεσα τον χρήστη στην λειτουργία login ώστε να συμπληρώσει τα στοιχεία του και να συνδεθεί στον λογαριασμό του.



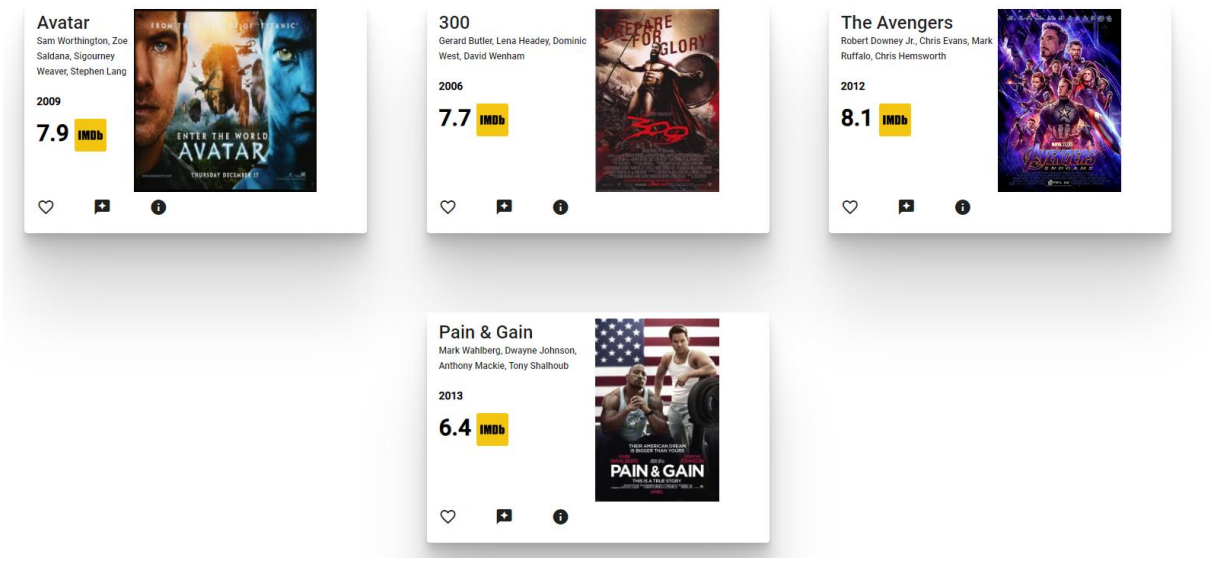
Search Movies



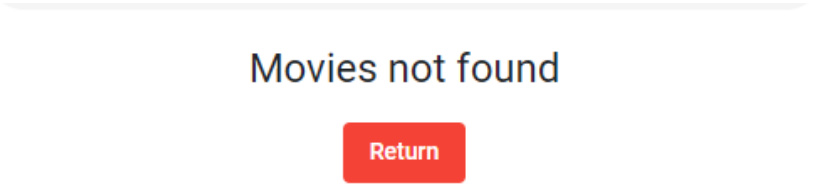
Ένας χρήστης, ακόμα και χωρίς να έχει συνδεθεί έχει κάποιες δυνατότητες που παρέχονται ελεύθερα από την εφαρμογή. Πατώντας το κουμπί Search Movies του παρουσιάζεται μία λίστα απ' όλες τις διαθέσιμες ταινίες που υπάρχουν. Από το menu αυτό έχει την δυνατότητα κατηγοριοποίησης των ταινιών μέσω του Search By category (π.χ. μπορεί να τον ενδιαφέρουν μόνο οι Action τύπου ταινίες).



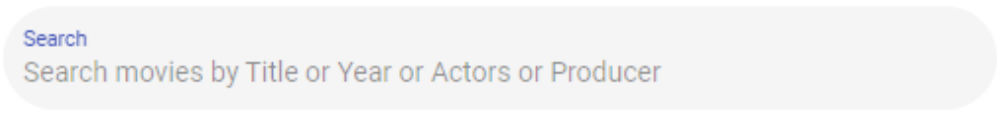
Επιλέγοντας π.χ. την κατηγορία Action του εμφανίζεται ένα διαφορετικό set ταινιών που ανήκουν στην συγκεκριμένη κατηγορία.



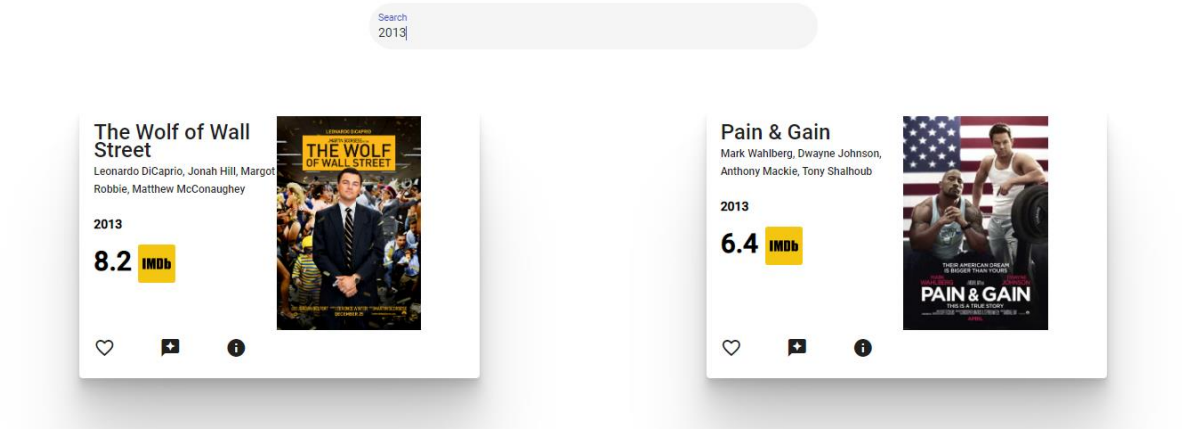
Βεβαίως στην περίπτωση που κάποια από τις κατηγορίες αυτές δεν διαθέτει κάποια ταινία (π.χ. Documentary) του εμφανίζει αντίστοιχο μήνυμα



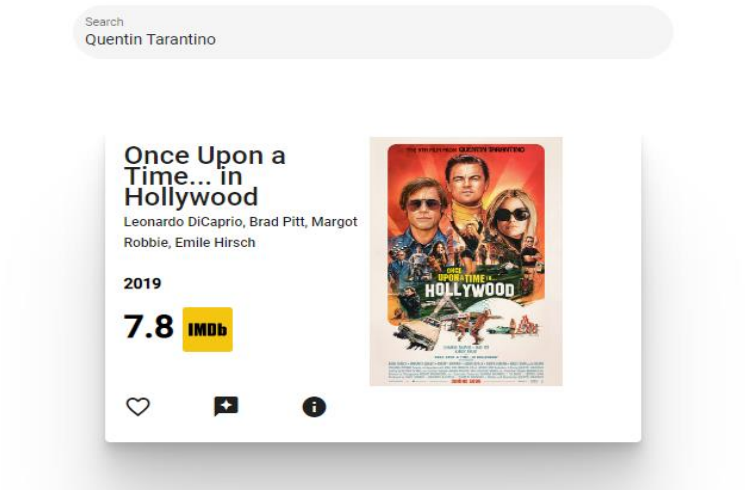
Στο σημείο αυτό του δίνεται και η επιλογή για επιστροφή (return) όπου θα του εμφανιστούν ξανά όλες οι διαθέσιμες ταινίες. Επιπλέον πέρα από το search ταινιών βάσει κατηγορίας παρέχεται η δυνατότητα αναζήτησης μέσω τίτλου ταινίας, χρονιάς παραγωγής, ηθοποιών αλλά και σκηνοθετών.



Παράδειγμα αναζήτησης ταινιών που “γυρίστηκαν” το 2013

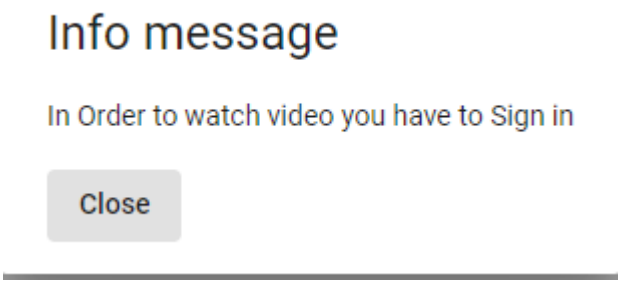


Παράδειγμα αναζήτησης ταινιών σύμφωνα με κάποιο σκηνοθέτη (Quentin Tarantino) για “πιο εξειδικευμένους” χρήστες.

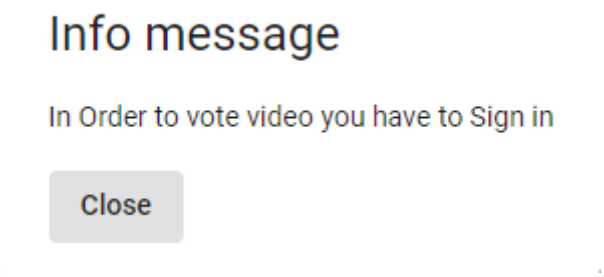


Κάθε ταινία πέρα από την βασική δυνατότητα να την παρακολουθήσει κάποιο χρήστης (πατώντας πάνω στην φωτογραφία της εκάστοτε ταινίας) διαθέτει και κάποιες άλλες χρήσιμες λειτουργίες (προσθήκη στα αγαπημένα, βαθμολόγηση ταινίας, διάφορες πληροφορίες για την ταινία). Φυσικά όλα τα παραπάνω δεν είναι διαθέσιμα για χρήστες που δεν έχουν συνδεθεί. Στην προσπάθεια τους να χρησιμοποιήσουν οποιαδήποτε από τις παραπάνω λειτουργίες τους εμφανίζονται αντίστοιχα μηνύματα.

Π.χ. προσπάθεια για παρακολούθηση ταινίας



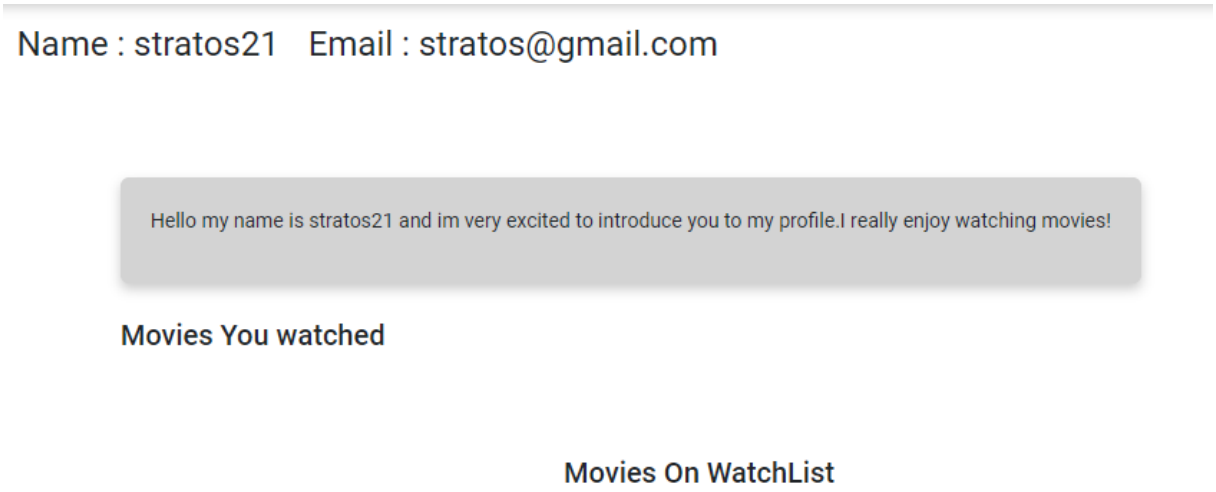
Π.χ. προσπάθεια για βαθμολόγηση ταινίας



Αφού πραγματοποιηθεί login από κάποιο χρήστη το αρχικό menu αλλάζει καθώς ενεργοποιούνται κάποια επιπλέον components όπως είναι το live chat, η προβολή προφίλ, καθώς και η συνολική βαθμολόγηση των ταινιών. Στο σημείο αυτό ο χρήστης μπορεί να χρησιμοποιήσει τις λειτουργίες που αναφέραμε προηγουμένως.

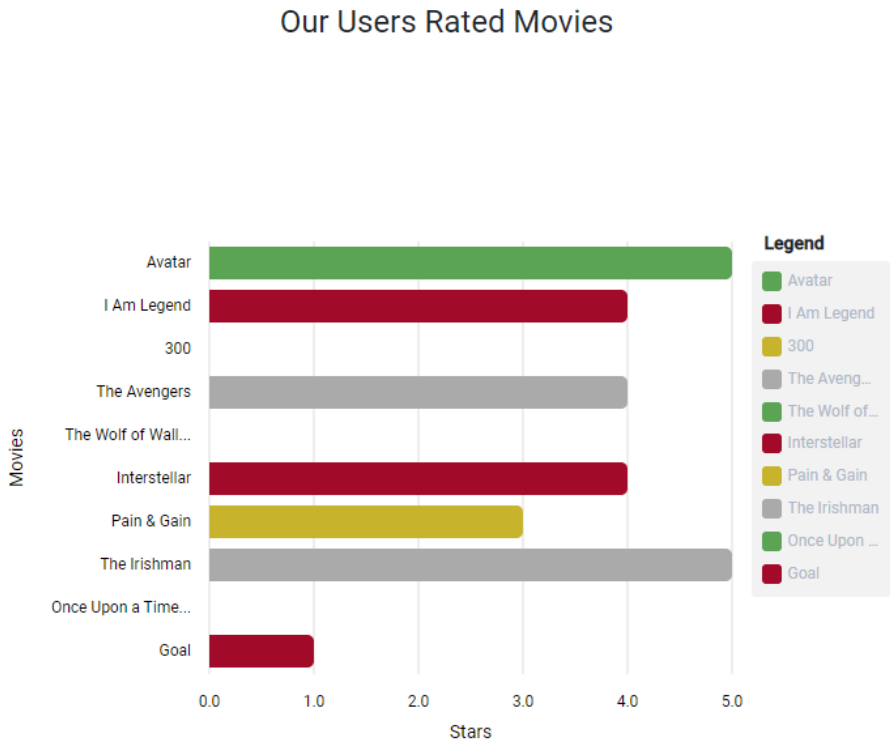
Profile

Σχετικά με το προφίλ του χρήστη εμφανίζονται κάποιες γενικές πληροφορίες για τον καθένα (όνομα χρήστη, mail κτλ). Στο προφίλ του κάθε χρήστη εμφανίζονται επίσης οι ταινίες τις οποίες έχει κάνει favorite (στο συγκεκριμένο παράδειγμα δεν έχει γίνει ακόμα κάποια ταινία favorite και το watchlist είναι κενό).



Rated movies

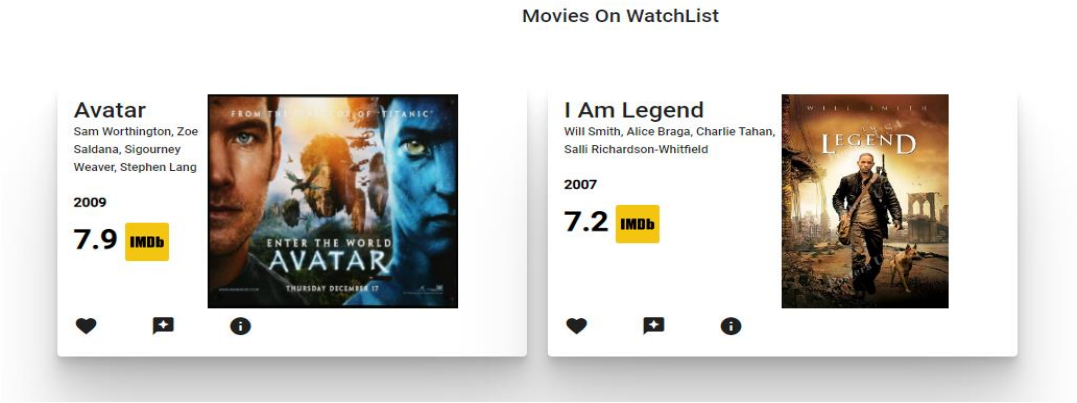
Στο συγκεκριμένο menu εμφανίζονται όλες οι διαθέσιμες ταινίες και η βαθμολογία τους (αν έχουν) όπως έχει συμψηφιστεί από το σύνολο των χρηστών.



Favorite

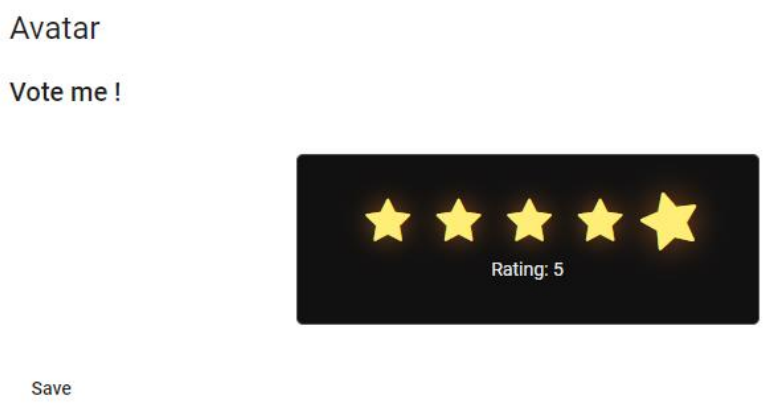
Από το Search Movies μενού μόλις ο χρήστης κάνει favorite όποια ταινία επιθυμεί (στη συγκεκριμένη περίπτωση έχουν γίνει οι ταινίες Avatar και I Am Legend).

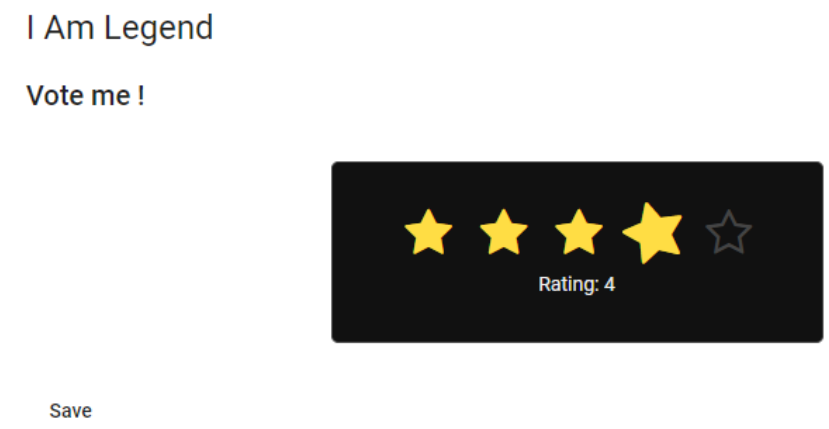
Στη συνέχεια η ταινία ή οι ταινίες που έκανε favorite περνάνε αυτόματα και στο προφίλ του.



Rate

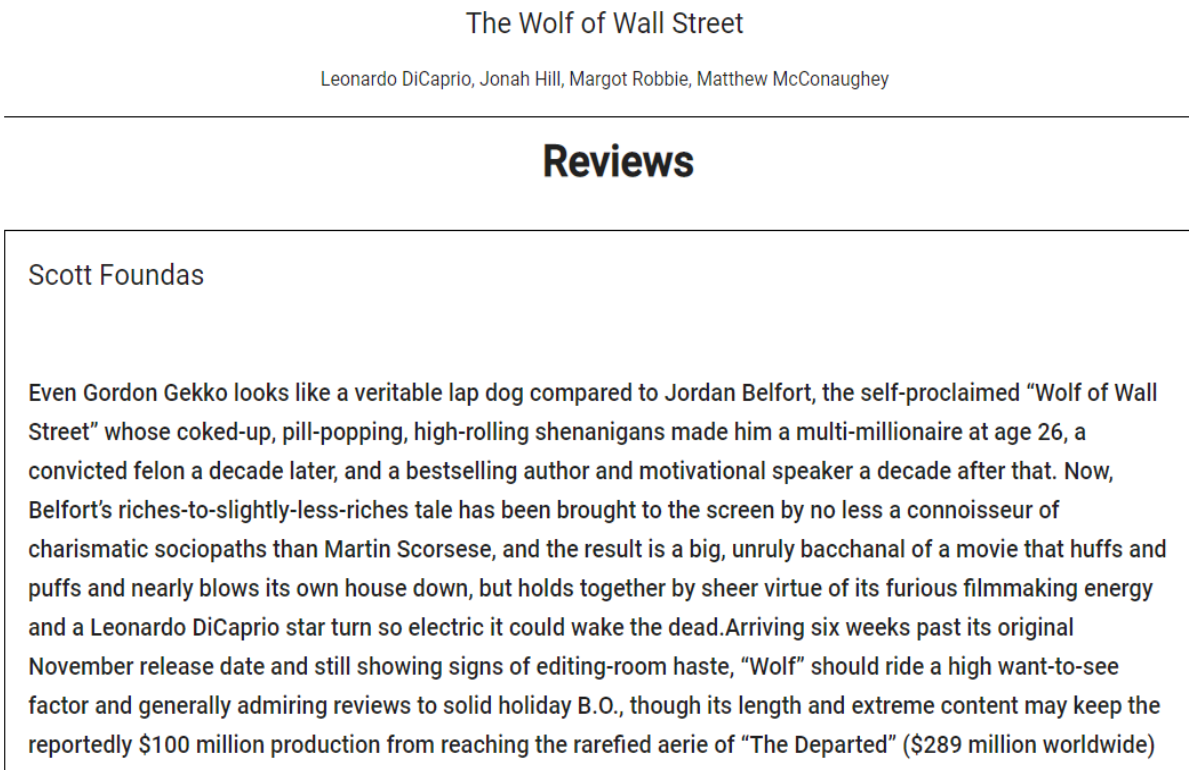
Φορτώνεται ένα διαφορετικό component όπως φαίνεται και μέσα από την εφαρμογή και ο χρήστης μπορεί να ψηφίσει κατά πόσο του άρεσε μία ταινία με κλίμακα από το ένα έως το πέντε πατώντας το κουμπί save για να κατοχυρωθεί η ψήφος του.





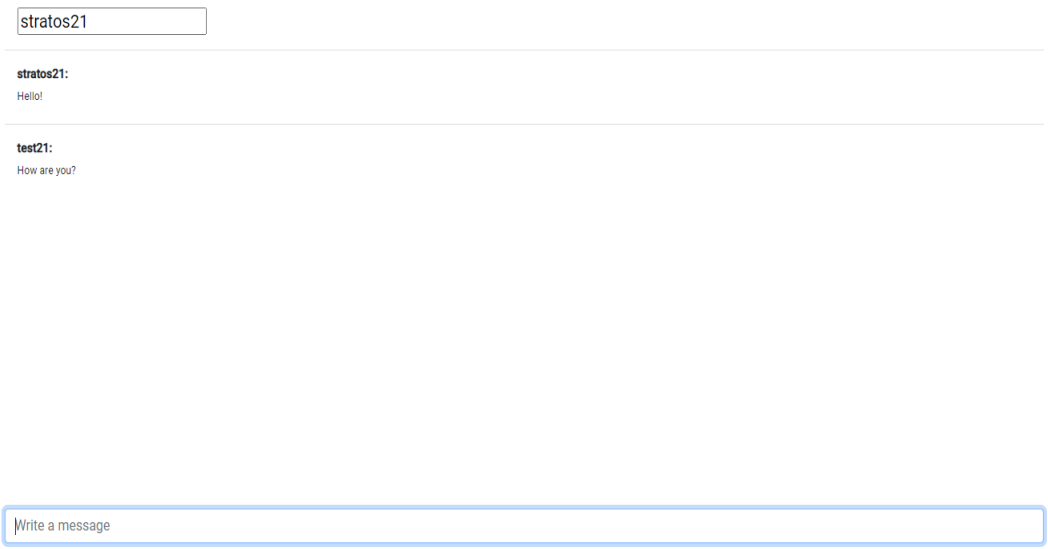
Info

Η συγκεκριμένη λειτουργία παρέχει στον χρήστη κριτικές για την ταινία που θέλει να δει, από επίσημους κριτές ταινιών. Μία ταινία εννοείται ότι μπορεί να εμπεριέχει παραπάνω από μια κριτικές. Παρακάτω ακολουθούν παραδείγματα από κριτικές ορισμένες ταινιών.



Live chat

Μία λειτουργία που δεν συνηθίζεται είναι η αλήθεια σε τέτοιους είδους εφαρμογές παρακολούθησης ταινιών. Ωστόσο έχει προστεθεί ως μία λειτουργία που έχει ως στόχο τον live σχολιασμό ταινιών ανάμεσα στους χρήστες. Ακολουθεί ενδεικτικό παράδειγμα επικοινωνίας δύο χρηστών.



Στο σημείο αυτό, αφού αναλύσαμε τις λειτουργίες της εφαρμογής, αξίζει να αναφέρουμε και τον τρόπο με τον οποίο με τον οποίο τα δεδομένα για τις ταινίες έχουν περαστεί στην βάση μας. Αυτό επιτυγχάνεται μέσω ενός Post request που στέλνει τα data μας σε μορφή JSON στο backend, πιο συγκεκριμένα στο http://localhost:4040/api/product.

```
{
    "Title": "Avatar",
    "Year": "2009",
    "Rated": "PG-13",
    "Released": "18 Dec 2009",
    "Runtime": "162 min",
    "Genre": "Action, Adventure, Fantasy",
    "Director": "James Cameron",
    "Writer": "James Cameron",
    "Actors": "Sam Worthington, Zoe Saldana, Sigourney Weaver, Stephen Lang",
    "Plot": "A paraplegic marine dispatched to the moon Pandora on a unique mission becomes torn between following his orders and protecting the world he feels
        is his home.",
    "Language": "English, Spanish",
    "Country": "USA, UK",
    "Video": "/assets/video/Avatar",
    "Poster": "/assets/img/Avatar.jpg",
    "Critics": [
        {
            "CriticName": "Richard Propes",
            "Critic": "Having gleefully projected writer/director James Cameron's \"Avatar\" as one of the \"10 Fall Films That Are Really Gonna' Suck,\" I must
```


Το συγκεκριμένο request αποθηκεύεται μέσα στη βάση μας με την παρακάτω μορφή.


```
_id: ObjectId('62aeee6113dba12f343651b1')
Title: "Avatar"
Year: "2009"
Released: "18 Dec 2009"
Actors: "Sam Worthington, Zoe Saldana, Sigourney Weaver, Stephen Lang"
Director: "James Cameron"
Poster: "/assets/img/Avatar.jpg"
Video: "/assets/video/Avatar"
imdbRating: "7.9"
Votes: 4
Rating: 19
› Critics: Array
› Genre: Array
__v: 0
```

Κεφάλαιο 5ο: Συμπεράσματα και προτάσεις βελτίωσης

Συμπεράσματα

Με την δημιουργία αυτής της εφαρμογής, ήρθα σε επαφή με ενδιαφέροντα frameworks (είτε Backend είτε Front-end) που χρησιμοποιούνται σήμερα από εταιρίες παγκόσμιου βεληνεκούς για την συγγραφή κώδικα και εφαρμογών. Επίσης κατάλαβα πόσο σημαντικό είναι ο κώδικας που γράφουμε να είναι αποδοτικός (δεν χρειάζεται να δημιουργούμε μεγάλες κλάσεις και components με πολλές λειτουργίες) αλλά και επαναχρησιμοποιήσιμος. Τέλος θα ήθελα να επισημάνω την σημαντικότητα της σχεδίασης μίας εφαρμογής πριν ξεκινήσουμε την υλοποίησης ώστε να μην συναντήσουμε μεγάλες δυσκολίες κατά την διάρκεια του project.

Προτάσεις Βελτίωσης

Αρχικά, κάτι που το θεωρώ και το πιο σημαντικό στο κομμάτι της βελτίωσης της εφαρμογής, είναι δημιουργία και ανάθεση ρόλων στους χρήστες (Administrator-User). Αυτή τη στιγμή όλοι οι χρήστες έχουν τον ίδιο ρόλο. Η προσθήκη αυτή σίγουρα θα βοηθήσει στην καλύτερη διαχείριση της εφαρμογής, καθώς δεν θα έχουν όλοι οι χρήστες τις ίδιες λειτουργίες και δικαιώματα. Χρήστες administrator θα μπορούν να πραγματοποιούν αλλαγές μέσα από την εφαρμογή (π.χ. διαγραφή μίας ταινίας ή διαγραφή κάποιο σχόλιου από το live chat). Μία ακόμα βελτίωση που έχει σχέση με το live chat θα μπορούσε να είναι η καταμέτρηση των συνολικών χρηστών που βρίσκονται online. Τέλος, σημαντική προσθήκη στην εφαρμογή θα ήταν και η προσθήκη μίας εντελώς καινούριας καρτέλας η οποία θα είχε αποκλειστική χρήση για παρακολούθηση σειρών αλλά και γενικώς όλων των διαθέσιμων λειτουργιών που διαθέτουν και οι ταινίες.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Βιβλία

[1] Jim Kurose and Keith Ross. Δικτύωση Υπολογιστών Προσέγγιση από Πάνω προς τα Κάτω

[2] Γεώργιος Β. Ξυλωμένος, Γεώργιος Κ. Πολύζος Τεχνολογία πολυμέσων και πολυμεσικές επικοινωνίες

Internet Site

[3] Angular documentation Available: https://angular.io/docs

[4] MVC documentation Available https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

[5] CDN documentation Availablehttps://www.cloudflare.com/learning/cdn/what-is-a-cdn/

[6] MongoDB documentation Availablehttps://www.mongodb.com/what-is-mongodb/features

[7] Typscript documentation Available https://www.typescriptlang.org/docs/

[8] Express documentation Available https://expressjs.com/

[9] Streaming protocols Available https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5424723/

[10] CDN cluster selection strategies Available https://www.ic.unicamp.br/~nfonseca/arquivos/CDN-kurose.pdf

[11] Streaming protocols Available https://www.cdnetworks.com/media-delivery-blog/video-streaming-protocols/

[12] Node.js documentation Available https://nodejs.org/en/docs/

[13] Solid documentation Available https://www.digitalocean.com/community/conceptual_articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design

Papers

[14] Adaptive streaming Available https://www.nctatechnicalpapers.com/Paper/2017/2017-addressing-ip-video-adaptive-stream-latency-and-video-player-synchronization pp. 3