

ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ



– Πτυχιακή Εργασία –

Πλαίσιο συλλογής πληροφορίας **playlist** ραδιοφωνικών σταθμών με
διαδικτυακή παρουσία και δημιουργία **playlist** αυτόματα με
machine-learning αλγόριθμο

Διακουμάκος Βασίλειος

Επιβλέπων: – Ιωάννης Καρύδης –

15 Σεπτεμβρίου 2022

Επιβλέπων

Ιωάννης Καρύδης, *Επίκουρος Καθηγητής*,
Ιόνιο Πανεπιστήμιο

Τριμελής Επιτροπή

Ιωάννης Καρύδης, *Επίκουρος Καθηγητής*,
Ιόνιο Πανεπιστήμιο

Κάτια-Λήδα Κερμανίδου, *Αναπληρώτρια Καθηγήτρια*,
Ιόνιο Πανεπιστήμιο

Μάρκος Αυλωνίτης, *Αναπληρωτής Καθηγητής*,
Ιόνιο Πανεπιστήμιο

Περίληψη

Η συγκεκριμένη εργασία ασχολήθηκε με τη συλλογή μουσικής πληροφορίας από διαδικτυακούς ραδιοφωνικούς σταθμούς και στη συνέχεια με την αυτόματη δημιουργία λιστών μουσικής (playlist). Δημιουργήθηκε μια διαδικτυακή εφαρμογή με τη βοήθεια ενός από τα πιο δημοφιλή πλαίσια λογισμικού (στην Αγγλική κι εφεξής στο κείμενο για συντομία, framework) της γλώσσας PHP το Laravel. Σχεδιάστηκε με τέτοιο τρόπο ώστε να μπορεί μελλοντικά να υποστηρίξει νέες λειτουργίες και τη βελτίωση των υπαρχόντων. Η εφαρμογή δημιουργήθηκε με σκοπό οι χρήστες της να μπορούν να δημιουργήσουν playlist με την επιλογή ενός μόνο τραγουδιού. Τα δεδομένα της εφαρμογής αντλήθηκαν από τις ιστοσελίδες πέντε ραδιοφωνικών σταθμών μέσα από τα API που διαθέτουν. Τα δεδομένα επεξεργάστηκαν με στόχο την αποθήκευση των σωστών πληροφοριών. Για την αποθήκευση των δεδομένων δημιουργήθηκε βάση δεδομένων με τρεις βασικούς πίνακες, τον πίνακα των τραγουδιών (songs), τον πίνακα των καλλιτεχνών (artists) και τον πίνακα των ραδιοφωνικών σταθμών (stations). Η βάση ενημερώνεται σε πραγματικό χρόνο αφού γίνεται έλεγχος σε κάθε σταθμό για κάθε τραγούδι που μεταδίδεται την συγκεκριμένη χρονική στιγμή. Για τον συγκεκριμένο έλεγχο γράφτηκε συγκεκριμένο script που κάνει κλήση στα API των σταθμών, το οποίο εκτελείται μέσω cron-job κάθε ενάμιση λεπτό. Τα τραγούδια αποθηκεύονται με τη σειρά που έχουν μεταδοθεί σε κάθε σταθμό. Τα δεδομένα χρησιμοποιούνται για τη δημιουργία playlist μέσα από αλγόριθμο που αναπτύχθηκε για την αυτοματοποίηση της διαδικασίας. Ο αλγόριθμος σχεδιάστηκε βασισμένος σε άλλες προσεγγίσεις και αποτελέσματα ακαδημαϊκών ερευνών. Η συγκεκριμένη προσέγγιση βασίζεται στο χρόνο μετάδοσης των τραγουδιών και στην αναζήτηση του πόσες φορές ένα τραγούδι επιλέγεται έπειτα από ένα συγκεκριμένο. Για την καλύτερη λειτουργία και οργάνωση του αλγορίθμου έγινε διαχωρισμός σε διαδοχικά βήματα. Για την αξιολόγηση του πειράματος δημιουργήθηκαν ερωτηματολόγια, που απαντήθηκαν από χρήστες της διαδικτυακής εφαρμογής με ερωτήσεις που θα βοηθήσουν στο να εξαχθούν σχετικά συμπεράσματα. Τέλος η εφαρμογή έχει εγκατασταθεί σε πραγματικό περιβάλλον και είναι διαθέσιμη στο διαδίκτυο

με αποτέλεσμα να μπορούν να την επισκεφτούν χρήστες καθημερινά.

Extended synopsis

Current project deals with collecting music information from web radio stations and then producing playlists. Results are presented on a web application created using Laravel, one of the most widespread PHP frameworks available, using MySQL as the database backend. It has been designed with scalability as a prime concern so that it will be feasible to expand it by adding new features and functionality in the future. The web app has been developed in order to make it possible for its users to create a playlist by selecting just one single song. Music data that the web application uses, have been drawn using the APIs provided by five web radio stations. Data has been processed so that only the necessary information will be extracted and stored in the database. In order to store those information there are 3 main database tables; these are: Songs -the database table holding the songs Artists -the database table holding the artists Stations -the database table holding the stations. The database gets updated in real time, as each station is being monitored (using crone-jobs) with regards to which song is being played at any given time. For this task, a script has been created which calls each web radio station's API in order to retrieve information about the song currently being played. The script is triggered (polls) every 90 seconds. Songs are then stored in the same order as the have been broadcasted by each radio station (thanks to the song broadcasted time information provided by data returned from the API call). Those songs are then used by the playlist creation algorithm in order to actually produce a playlist. The playlist creation algorithm has been developed based on other common playlist creation algorithm approaches as well as using academic research. Current approach is heavily based on two factors:

1. the songs broadcast time and
2. How many times a song has been played after a particular other song

For the best algorithm operation and organisation, the whole process has been dived into

steps. For the purpose of evaluating the current project a questionnaire has been created. These have been answered by some of the web application users. Questionnaire contains questions which will help to estimate the accuracy -thus the level of success- of the playlists produced by the algorithm. Furthermore, the web application has been hosted on a web server so that it can be access at any given time by anyone.

Πρόλογος και Ευχαριστίες

Ολοκληρώνοντας αυτήν την πτυχιακή εργασία θα ήθελα να ευχαριστήσω θερμά τον καθηγητή μου κ. Ιωάννη Καρύδη για τη σημαντική βοήθεια του και καθοδήγηση κατά τη διάρκεια της εκπόνησης της εργασίας, καθώς και για τη στήριξη που μου προσέφερε. Επιπλέον θα ήθελα να ευχαριστήσω την οικογένειά μου που ήταν δίπλα μου σε κάθε μου βήμα κατά τη διάρκεια της φοίτησής μου στο τμήμα Πληροφορικής του Ιονίου Πανεπιστημίου. Η βοήθεια και η στήριξη όλων ήταν πολύτιμη.

Αθήνα, 15 Σεπτεμβρίου 2022

Βασίλειος Διακουμάκος

Περιεχόμενα

A'	Εισαγωγή	1
A'.1	Ο ΣΚΟΠΟΣ ΚΑΙ ΟΙ ΣΤΟΧΟΙ ΤΗΣ ΕΦΑΡΜΟΓΗΣ	1
A'.2	Η ΣΥΝΕΙΣΦΟΡΑ	2
A'.3	ΠΕΡΙΛΗΨΗ ΚΕΦΑΛΑΙΩΝ	3
B'	ΣΧΕΤΙΚΕΣ ΕΡΕΥΝΕΣ	4
B'.1	ΠΡΟΪΠΑΡΧΟΥΣΕΣ ΕΡΕΥΝΕΣ	4
B'.2	ΣΥΓΚΡΙΣΗ ΕΡΓΑΣΙΑΣ ΜΕ ΔΙΑΦΟΡΕΣ ΕΡΕΥΝΕΣ	9
Γ'	Ανάπτυξη εφαρμογής	11
Γ'.1	ΕΡΓΑΛΕΙΑ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ	12
Γ'.1.1	Apache	12
Γ'.1.2	MySQL	13
Γ'.1.3	Η γλώσσα PHP	13
Γ'.1.4	Τεχνολογίες για τη Δημιουργία του UI.	14
Γ'.1.5	Το laravel	14
Γ'.2	ΠΗΓΗ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΔΙΑΧΕΙΡΙΣΗ	16
Γ'.3	ΑΝΑΛΥΣΗ ΚΑΙ ΣΧΕΔΙΑΣΜΟΣ ΕΦΑΡΜΟΓΗΣ	19
Γ'.4	Η ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ	25
Γ'.4.1	Σχεδιασμός και σχήμα	25
Γ'.4.2	Πίνακες	27
Γ'.5	ΤΕΧΝΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ	29
Γ'.5.1	Ανάλυση τεχνικών ανάπτυξης κώδικα	31

Περιεχόμενα	vii
Δ' ΑΥΤΟΜΑΤΗ ΔΗΜΙΟΥΡΓΙΑ PLAYLIST	34
Δ'.1 ΑΠΟΦΑΣΕΙΣ ΥΛΟΠΟΙΗΣΗΣ ΑΛΓΟΡΙΘΜΟΥ	34
Δ'.2 ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ	35
Δ'.3 ΠΕΙΡΑΜΑΤΙΣΜΟΣ ΑΛΓΟΡΙΘΜΟΥ	39
Ε' ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΗ ΑΝΑΠΤΥΞΗ	43
Ε'.1 ΣΥΜΠΕΡΑΣΜΑΤΑ	43
Ε'.2 ΕΠΕΚΤΑΣΗ ΕΦΑΡΜΟΓΗΣ ΚΑΙ ΜΕΛΛΟΝΤΙΚΑ FEATURES	44
ΧΡΗΣΙΜΟΙ ΣΥΝΔΕΣΜΟΙ	46
Βιβλιογραφία	47

Κατάλογος Σχημάτων

Γ'.1 Στο συγκεκριμένο σχήμα παρουσιάζεται η βασική λειτουργία της εφαρμογής. .	11
Γ'.2 Εικόνα η οποία παρουσιάζει την αρχική σελίδα της εφαρμογής	21
Γ'.3 Εικόνα η οποία παρουσιάζει την σελίδα stations της εφαρμογής	22
Γ'.4 Εικόνα η οποία παρουσιάζει τη σελίδα Now Playing της εφαρμογής	22
Γ'.5 Εικόνα η οποία παρουσιάζει τη διαδικασία δημιουργίας	23
Γ'.6 Εικόνα η οποία παρουσιάζει την playlist που δημιουργήθηκε.	23
Γ'.7 Εικόνα η οποία παρουσιάζει τη λίστα τραγουδιών με το μεγαλύτερο ποσοστό εμφάνισης	24
Γ'.8 Διάγραμμα οντοτήτων συσχετίσεων	26
Γ'.9 Το σχήμα της βάσης δεδομένων	27
Δ'.10 Διάγραμμα το οποίο παρουσιάζει τις ηλικίες των συμμετεχόντων.	39
Δ'.11 Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις για τη λίστα με μικρή συσχέτιση μεταξύ των τραγουδιών.	40
Δ'.12 Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις για τη λίστα με μεγάλη συσχέτιση μεταξύ των τραγουδιών.	41
Δ'.13 Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις για τη λίστα που δημιούργησε ο αλγόριθμος της εφαρμογής.	41
Δ'.14 Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις των χρηστών για τη συνοχή των λιστών που δημιούργησαν.	42

Κατάλογος Πινάκων

Κεφάλαιο Α΄

Εισαγωγή

Στο παρόν κεφάλαιο γίνεται μια γρήγορη ανάλυση του σκοπού της εργασίας, των στόχων αλλά και η παρουσίαση των κεφαλαίων συνοπτικά. Επίσης αναλύεται η συνεισφορά της εργασίας (διαδικτυακής εφαρμογής) προς την επιστήμη της πληροφορικής και η σημασία της για το χρήστη.

Α΄.1 Ο σκοπός και οι στόχοι της εφαρμογής

Κάνοντας μια αναζήτηση στο διαδίκτυο θα συναντήσουμε αρκετές πλατφόρμες αναπαραγωγής μουσικής. Πολλές από αυτές διαθέτουν μουσική δωρεάν, άλλες με πληρωμή, έχοντας όμως το στόχο να διασκεδάσουν το χρήστη παρέχοντας ένα μεγάλο πλήθος τραγουδιών. Το πάθος των χρηστών για τη μουσική αλλά και την ανακάλυψη της, γέννησε την ανάγκη δημιουργίας λιστών μουσικής (στην Αγγλική, κι εφεξής στο κείμενο για συντομία, playlist). Ερευνητικές ομάδες από όλο τον κόσμο καθημερινά χρησιμοποιούν και δουλεύουν στην ανάπτυξη αλγορίθμων μηχανικής μάθησης με στόχο να καταφέρουν να παρέχουν στον κόσμο μουσική που ίσως θα τους ενδιέφερε, αναλύοντας έτσι καθημερινά δεδομένα από εκατομμύρια χρήστες με στόχο τα καλύτερα αποτελέσματα.

Η συγκεκριμένη έρευνα έχει ως σκοπό να παρέχει μουσικό περιεχόμενο στους χρήστες που αγαπούν τη μουσική και τους αρέσει να την εξερευνούν διαρκώς. Να βοηθήσει και αυτή με τη σειρά της στην έρευνα και στην επιστήμη της πληροφορικής. Να μπορέσει να δώσει ένα εργαλείο στον χρήστη που θα μπορεί να το χρησιμοποιήσει καθημερινά, συνδυάζοντας την ευκολία και την αμεσότητα που παρέχει μια διαδικτυακή εφαρμογή με την αυτοματοποίηση

διαδικασιών μέσω αλγόριθμου.

Ένας από τους στόχους της εφαρμογής είναι ο χρήστης να μπορεί να περιηγηθεί εύκολα στην εφαρμογή και να έχει άμεσα αποτελέσματα στην οθόνη του. Δηλαδή μέσα από ένα φιλικό περιβάλλον προς αυτόν και με μόλις δύο βήματα, να μπορεί να δημιουργήσει μουσικές λίστες έχοντας στα χέρια του μια μεγάλη ποικιλία τραγουδιών. Να ανακαλύψει μουσική μέσα από την εφαρμογή, τραγούδια που ίσως είναι της αρεσκείας του και δεν είχε την ευκαιρία να τα ακούσει μέχρι σήμερα. Να μπορεί να δημιουργεί όσες playlist θέλει καθημερινά χωρίς κανένα κόστος. Από τους πιο βασικούς στόχους της εφαρμογής επίσης είναι η ανάλυση των μουσικών δεδομένων αλλά και η αποθήκευση τους σε μια βάση δεδομένων η οποία κρατάει το ιστορικό αναπαραγωγής. Στη συγκεκριμένη εργασία προτείνεται ένας διαφορετικός τρόπος προσέγγισης δημιουργίας playlist μουσικού περιεχομένου που αναλύεται στο επόμενο κεφάλαιο.

Α΄.2 Η Συνεισφορά

Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο ένας από τους βασικούς σκοπούς της εργασίας ήταν και η συνεισφορά προς την επιστήμη. Μέσα από τη συγκεκριμένη εργασία πολλά πράγματα θα μπορούσαν να χρησιμοποιηθούν από συναδέλφους - φοιτητές του τμήματος και ίσως τους βοηθήσουν και στην ακαδημαϊκή τους καριέρα. Έτσι και αλλιώς η γνώση θα πρέπει να μοιράζεται με αποτέλεσμα γίνουν μαζικά βήματα προς την εξέλιξη όσον αφορά τη πληροφορική. Πιο συγκεκριμένα μέσα από την παρούσα εργασία προσφέρονται τα εξής:

- Δημιουργία διαδικτυακής εφαρμογής (Στην Αγγλική, web - application) αναλύοντας όλη τη διαδικασία έτσι ώστε να μπορούν να κατανοηθούν όλες οι τεχνικές ανάπτυξης αλλά και τα βασικά εργαλεία δημιουργίας ιστοσελίδων και διαδικτυακών εφαρμογών. Ο κώδικας της εφαρμογής είναι διαθέσιμος στους χρήσιμους συνδέσμους παρακάτω.
- Δημιουργία προγραμματισμένης εργασίας (Στην Αγγλική και εφεξής στο κείμενο για συντομία, cron job) μια λειτουργία που τη συναντάμε σε διάφορους πίνακες ελέγχου. Στη συγκεκριμένη περίπτωση είναι ένα script το οποίο τρέχει μέσα από μια linux εντολή για τον προγραμματισμό διεργασιών μελλοντικά - περιοδικά. Πρόκειται στην ουσία για το μηχανισμό που μας επιτρέπει να γίνονται διάφορες λειτουργίες αυτόματα, σε συγκεκριμένο χρόνο χωρίς να χρειάζεται να παρεμβάλλεται κάποιος χρήστης (What is Cron Job?, 2022).

- Δημιουργία συγκεκριμένου τρόπου επεξεργασίας δεδομένων που αντλούνται από τους ραδιοφωνικούς σταθμούς. Δίνοντας δυναμικό χαρακτήρα στην επεξεργασία (Στην Αγγλική, processing) των δεδομένων καλύπτοντας ένα μεγάλο μέρος σταθμών που δίνουν δεδομένα μέσα από διεπαφή προγραμματισμού εφαρμογών (Στην Αγγλική και εφεξής στο κείμενο για συντομία, API).
- Δημιουργία αλγόριθμου ο οποίος αυτοματοποιεί τη διαδικασία δημιουργίας playlist που είναι μια ευρύτερη προσφορά προς την επιστήμη. Ο αλγόριθμος αναλύεται πλήρως στα επόμενα κεφάλαια και είναι γραμμένος με τρόπο που μπορεί να συντηρηθεί αφού κατανοηθεί σωστά.
- Δημιουργία βάσης δεδομένων με περίπου 6.500 τραγούδια. Η συγκεκριμένη συλλογή έχει αξία καθώς έκτος από ότι αναπτύχθηκαν αρκετές τεχνικές για τη γρηγορότερη ανταπόκριση, επίσης καταγράφει και την ιστορικότητα αναπαραγωγής των τραγουδιών στους σταθμούς.

Τέλος σημαντικό ρόλο έχει και η συνεισφορά ως προς το χρήστη, αφού δημιουργήθηκε μια εφαρμογή με σκοπό να χρησιμοποιηθεί σαν ένα εργαλείο το οποίο επεξεργάζεται μουσικά δεδομένα, δίνοντας στο χρήστη τη δυνατότητα να μπορέσει να δημιουργήσει μουσικές λίστες αλλά και να περιηγηθεί σε μια διαδικτυακή εφαρμογή.

Α΄.3 Περίληψη Κεφαλαίων

Η παρούσα εργασία έχει χωριστεί σε τέσσερα βασικά μέρη. Στη συγκεκριμένη ενότητα γίνεται μια συνοπτική παρουσίαση των βασικών αυτών μερών. Αναλύει τις αποφάσεις που πάρθηκαν για τη δόμηση του αλγορίθμου και επεξηγεί ένα ένα τα βήματα του αλγορίθμου. Εξηγεί το λόγο που πάρθηκαν οι συγκεκριμένες αποφάσεις και παρουσιάζει τις τεχνικές που χρησιμοποιήθηκαν για τη δημιουργία του. Ακόμα αναφέρεται και στον πειραματισμό του αλγορίθμου, αφού αναλύει τις απαντήσεις που μας έδωσαν οι χρήστες του συστήματος μέσω ερωτηματολογίου. Το τελευταίο μέρος αναφέρεται στα συμπεράσματα που βγήκαν μέσα από τη συγκεκριμένη εργασία αλλά και στις δυνατότητες εξέλιξης της υπάρχουσας εφαρμογής, αναφέροντας καινούργιες ιδέες προς υλοποίηση - χαρακτηριστικά (Στην Αγγλική και εφεξής στο κείμενο για συντομία, features) στο μέλλον.

Κεφάλαιο Β΄

ΣΧΕΤΙΚΕΣ ΕΡΕΥΝΕΣ

Όσον αφορά τη μουσική και ειδικότερα τη δημιουργία μουσικών λιστών, εδώ και πολλά χρόνια γίνονται έρευνες και υπάρχουν αρκετές διαφορετικές προσεγγίσεις. Οι αυτοματοποιημένες playlist είναι ένα χαρακτηριστικό που συναντάμε στις περισσότερες εφαρμογές παροχής μουσικής σήμερα. Εδώ και αρκετά χρόνια υπάρχουν πολλές ακαδημαϊκές έρευνες που στοχεύουν στη δημιουργία καλύτερων αυτοματοποιημένων λιστών μουσικής. Η χρήση αλγορίθμων μηχανικής μάθησης, η μελέτη των χαρακτηριστικών αλλά και η χρήση διαφορετικών παραμέτρων είναι κάποια από τα συστατικά αυτών των ερευνών. Η παρούσα εργασία βασίστηκε σε συγκεκριμένες έρευνες, προσπαθώντας να δώσει μια διαφορετική προσέγγιση.

Β΄.1 Προϋπάρχουσες Έρευνες

Σύμφωνα με τους Jannach, Kamehkhosh & Bonnin, (2016) ακόμα και στις μέρες μας οι αλγόριθμοι που υπάρχουν για τη δημιουργία playlist δεν έχουν τελειοποιηθεί, γιατί η έρευνα στο συγκεκριμένο κομμάτι δεν είναι εύκολη. Σύμφωνα με τους συγκεκριμένους είναι απαραίτητο οι συμμετέχοντες στην έρευνά να ακούνε την ίδια τη μουσική για να εξασφαλίσουν καλύτερα αποτελέσματα. Θεωρούν ότι οι playlist που δημιουργούνται από χρήστες, δηλαδή χωρίς την ύπαρξη κάποιου αλγορίθμου, είναι οι δέουσες για το πείραμα τους. Στόχος τους είναι να συγκρίνουν τεχνικές δημιουργίας είτε ακαδημαϊκές είτε εμπορικές (continuations playlists) με λίστες που δημιούργησαν οι ίδιοι οι χρήστες. Ένα από τα πιο συνήθη προβλήματα στην αυτοματοποιημένη παραγωγή λιστών αναπαραγωγής που βασίζονται στα επόμενα τραγούδια

είναι ότι λειτουργούν με βάση το ιστορικό του χρήστη χρονολογικά. Για να ξεχωρίσουν ποιοι αλγόριθμοι φαίνονται να είναι προκατειλημμένοι χρησιμοποιούν συγκεκριμένα μέτρα υπολογισμού (Στην Αγγλική και εφεξής στο κείμενο *measures*) όπως η ακρίβεια (Στην Αγγλική και εφεξής στο κείμενο για συντομία *accuracy*), η συνοχή (Στην Αγγλική εφεξής στο κείμενο για συντομία *coherence*) και η ποικιλομορφία (Στην Αγγλική και εφεξής στο κείμενο για συντομία *diversity*) όπως χρησιμοποίησαν και σε προηγούμενη έρευνα οι Jannach, Lerche, Kamehkhosh και Jugovac, (2015). Η έρευνά τους βασίζεται σε ακαδημαϊκές τεχνικές διαφορικών τύπων, όπως επίσης και σε εμπορικές τεχνικές παραγωγής μουσικών λιστών όπως για παράδειγμα η τεχνική που χρησιμοποιεί η The echo Nest ¹ θυγατρική του Spotify². Αρχικά ξεκίνησαν με αλγόριθμους που βασίζονται στο περιεχόμενο των τραγουδιών (*content similarity*) όπως στο Collaborative Filtering ως μία τεχνική μηχανικής μάθησης που χρησιμοποιείται για να ανιχνεύσει τις σχέσεις μεταξύ διαφορετικών δεδομένων, μια τεχνική που βασίζεται στο περιεχόμενο των κοινωνικών ετικετών (*social tags*), μία υβριδική μέθοδο και μία εμπορική υπηρεσία *playlist* (Στην Αγγλική, *commercial service*). Όλες οι τεχνικές έχουν σαν κύριο σκοπό να προσδιορίσουν τη σχετικότητα του υποψήφιου επόμενου τραγουδιού με το παρόν κομμάτι. Πιο συγκεκριμένα οι τεχνικές είναι :

1. CollocatedArtists- Greatest Hits (CAGH), μια προσέγγιση που βασίζεται στους καλλιτέχνες. Η συνύπαρξη καλλιτεχνών στα δεδομένα εκπαίδευσης χρησιμοποιείται ως μέτρο ομοιότητας.
2. kNN300: Η συγκεκριμένη μέθοδος βασίζεται στον αλγόριθμο πλησιέστερου γείτονα k-Nearest-Neighbor. Δίνοντας μία είσοδο ψάχνει τις *playlist* που περιέχουν ίδια τραγούδια. Ο K-nn είναι μία τεχνική που αντιπροσωπεύει μια από τις πιο ισχυρές βάσεις για πολλά πειράματα τέτοιου είδους. Στη συγκεκριμένη έρευνα το K τέθηκε 300.
3. Content-Based : Τεχνική που βασίζεται σε κοινωνικές ετικέτες, υπολογίζει τα διανύσματα TF-IDF για κάθε κομμάτι.
4. kNN300PCPA: Τεχνική που βασίζεται στον K-NN με περισσότερες βαθμολογίες.
5. TEN: Τεχνική που χρησιμοποιεί η The echo Nest

Πιο συγκεκριμένα έκαναν αξιολόγηση των δεδομένων που πάρθηκαν από διάφορες διαδικτυακές πλατφόρμες. Χρησιμοποίησαν έτοιμες *playlist* οι οποίες διέθεταν διακριτικά χαρακτηρι-

¹<https://blog.echonest.com/>

²<https://open.spotify.com/>

στικά. Προτείνουν τον χωρισμό της playlist στο δοκιμαστικό σετ σε δύο μισά. Το πρώτο μισό παρέχεται σαν είσοδος ώστε να δημιουργήσει μια συνέχεια που είναι τόσο μεγάλη όσο το άλλο μισό. Στη συνέχεια γίνεται σύγκριση του παραγομένου σετ τραγουδιών με το υπόλοιπο δεύτερο μισό που κρατήθηκε στην αρχή για να γίνει μέτρηση του πόσο εύστοχος είναι ένας αλγόριθμος και κατά πόσο μπορεί να μιμηθεί την ανθρώπινη συμπεριφορά. Επιπλέον εφαρμόζονται διαφορετικά μέτρα αξιολόγησης της ποιότητας των συνεχειών των playlist, αφού στη συγκεκριμένη περίπτωση γίνεται διαχωρισμός των δεδομένων των playlist σε 75 % για εκπαίδευση και 25 % για δοκιμές με βάση ένα κριτήριο για κάθε χρήστη. Έγιναν τέσσερις παραλλαγές του μέτρου απόδοσης ανάκλησης (Στην Αγγλική και εφεξής στο κείμενο για συντομία recall) και του μέτρου απόδοσης ακρίβειας (Στην Αγγλική και εφεξής στο κείμενο για συντομία precision), για να προσδιορίσουν εάν οι αλγόριθμοι προτείνουν τα σωστά κομμάτια, τους σωστούς καλλιτέχνες, τα σωστά είδη και τις κατάλληλες ετικέτες. Η υβριδική μέθοδος (kNN300PCPA), ξεπέρασε όλες τις άλλες τεχνικές όσον αφορά το recall και το precision. Βέβαια επισημαίνεται ότι άλλες έρευνες έχουν δείξει ότι πολλές φορές αλγόριθμοι που οδηγούν σε υψηλό recall και precision εμφανίζουν συχνά μια προκατάληψη δημοτικότητας. Στη συνέχεια αναλύουν τις λίστες με βάση δύο επιλεγμένα χαρακτηριστικά, τη δημοτικότητα και το ρυθμό. Επιπλέον, συγκρίνουν τις δημιουργούμενες συνέχειες με αυτές που δημιουργήθηκαν από τους χρήστες. Για την αξιολόγηση των επίπεδων διαφορετικότητας των παραγομένων playlist, με τις λίστες που έχουν δημιουργηθεί από του χρήστες έγινε προσδιορισμός της απόστασης του μέσου όρο των δύο χαρακτηριστικών tempo - popularity μεταξύ όλων των ζευγών κομματιών για την κάθε λίστα. Στη συνέχεια μέσα από τις τιμές απόλυτης απόστασης ερμηνεύτηκαν τα εξής αποτελέσματα. Για τον ρυθμό (tempo) όλες οι λίστες αναπαραγωγής κατάφεραν να μιμηθούν τις επιλογές των χρηστών σε κάποιο βαθμό. Επίσης παρατηρήθηκε ότι η συνολική ποικιλομορφία ρυθμού (tempo) είναι συγκριτικά υψηλή. Όσον αφορά τη δημοτικότητα (popularity) οι περισσότεροι ακαδημαϊκοί δημιουργοί μουσικών λιστών (Στην Αγγλική και εφεξής στο κείμενο για συντομία, playlisters) δεν εστιάζουν μόνο στο χαρακτηριστικό της δημοτικότητας. Παρατηρήθηκε ότι η μεταβλητότητα ως προς τη δημοτικότητα είναι επίσης πολύ μεγαλύτερη από ό,τι για τα δοκιμαστικά μισά που δημιουργήθηκαν από τους χρήστες. Οι μετρήσεις επιβεβαιώνουν ότι μερικοί από τους playlisters αναπαραγωγής παρουσιάζουν προκαταλήψεις με αποτέλεσμα να δημιουργήσουν είτε πιο ομοιογενείς λίστες είτε πιο ποικιλόμορφες λίστες.

Οι Vatulkin, Bonnin, και Jannach (2016) αναφέρουν ότι τα τελευταία χρόνια έχουν γίνει αρκετές προσεγγίσεις για την αυτόματη αναγνώριση μουσικών ειδών και για περισσότερες κατηγορίες, όπως προσωπικές προτιμήσεις, στυλ και διάθεση. Μεταξύ των διαφορετικών πηγών που χρησιμοποιούνται για το χτίσιμο μοντέλων ταξινόμησης που σχετίζονται με τη μουσική, ιδιαίτερο ρόλο παίζουν τα χαρακτηριστικά του ηχητικού σήματος. Όσον αφορά τη βιβλιογραφία του MR η κατασκευή μοντέλων ταξινόμησης από χαρακτηριστικά ήχου είναι ίσως η προσέγγισή που εμφανίζεται περισσότερο. Ένα από τα βασικά πλεονεκτήματα της συγκεκριμένης τεχνικής είναι ότι τα συγκεκριμένα χαρακτηριστικά μπορούν να εξαχθούν από ψηφιακά μουσικά κομμάτια. Βέβαια πολλές φορές η εξαγωγή τέτοιων πληροφοριών απαιτεί ένα σημαντικό υπολογιστικό κόστος και ο ήχος πολλές φορές, αν δεν συνδυάζεται με κάποιο άλλο χαρακτηριστικό, μπορεί να μην παρέχει τις απαραίτητες πληροφορίες για τον προσδιορισμό των διακριτών ιδιοτήτων μίας μουσικής κατηγορίας. Στο συγκεκριμένο έργο χρησιμοποιούνται οι λίστες των ακροατών σαν μία άλλη ενδιαφέρουσα πηγή για την εξαγωγή απαραίτητων χαρακτηριστικών και κατηγοριοποίηση μουσικής. Επίσης οι συγκεκριμένοι ερευνητές αναφέρουν ότι πολλά κομμάτια μιας λίστας συχνά ανήκουν στον ίδιο καλλιτέχνη ή στο ίδιο στυλ μουσικής. Τα τυπικά βήματα ενός αλγορίθμου για την κατηγοριοποίηση της μουσικής είναι :

1. Η εξαγωγή χαρακτηριστικών (feature extraction): Επιλογή των σωστών χαρακτηριστικών που ταυτίζονται με τα μουσικά δεδομένα.
2. Η επεξεργασία των χαρακτηριστικών. Σε επόμενο βήμα από την εξαγωγή των χαρακτηριστικών τα συγκεκριμένα δεδομένα υποβάλλονται σε επεξεργασία, όπως η κανονικοποίηση. Στόχος είναι να έρθουν τα δεδομένα στην κατάλληλη μορφή μέσα από μετασχηματισμούς για τη μείωση του κόστους υπολογισμού.
3. Η δημιουργία μοντέλων ταξινόμησης με βάση τα παραδείγματα εκπαίδευσης του αλγορίθμου. Τα χαρακτηριστικά που προέκυψαν από τις προηγούμενες διαδικασίες χρησιμοποιούνται για την κατασκευή μοντέλων ταξινόμησης.

Στη συγκεκριμένη έρευνα γίνεται μια διαφορετική προσέγγιση από αυτή που συνηθίζεται αφού βασίζεται στο ότι η ομοιογένεια είναι ένα βασικό κριτήριο για τα άτομα που δημιουργούν λίστες. Επίσης για το πρόβλημα ταξινόμησης, υποθέτουν ότι η παρουσία ενός συγκεκριμένου μουσικού κομματιού σε μια δεδομένη λίστα αναπαραγωγής συνεπάγεται μεγαλύτερη πιθανότητα τα άλλα τραγούδια αυτής της λίστας να ανήκουν στην ίδια ή παρόμοια κατηγορία. Στόχος ήταν η χρήση ενός μοντέλου μηχανικής μάθησης που εκπαιδεύεται με βάση τις πληροφορίες από πιο συχνούς συνυπάρχοντες καλλιτέχνες. Η μέθοδος που ακολουθήθηκε αποτελείται από 5 βήματα :

1. Resolving spelling problems (Επίλυση προβλημάτων ορθογραφίας)
2. Identifying relevant artist co-occurrences (Προσδιορισμός σχετικών συνυπαρχόντων καλλιτεχνών)
3. Removing duplicates (Αφαίρεση όλων των διπλότυπων)
4. Normalisation (Ομαλοποίηση)
5. Training of classification models (Εκπαίδευση του μοντέλου ταξινόμησης).

Για να μπορέσουν να συγκρίνουν τη δική τους προσέγγιση που βασίζεται σε playlist με μια τυπική που βασίζεται σε σήμα ήχου χρησιμοποίησαν ένα συγκεκριμένο πειραματικό σετ. Ο στόχος ήταν να κατηγοριοποιήσουν τραγούδια σε 6 είδη μουσικής (Classic, Jazz, Pop, etc.) και σε 8 στυλ μουσικής (e.g., ClubDance, HeavyMetal, Urban). Για κάθε μία από τις 14 κατηγορίες το σύνολο περιλαμβάνει 10 θετικά και 10 αρνητικά παραδείγματα. Τα μοντέλα στη συνέχεια αξιολογήθηκαν με βάση ένα δοκιμαστικό σετ 120 τραγουδιών που δημιούργησε ο Vatulkin, (2013) το οποίο είχε την ίδια κατανομή είδους με το optimization set. Για κάθε μία από τις δύο προσεγγίσεις δημιουργήθηκαν 4 σετ. Για τη μέθοδο που αφορά τα ηχητικά χαρακτηριστικά χρησιμοποιήθηκαν 4 σετ χαρακτηριστικών ήχου. Η πρώτη ομάδα περιγράφει 636 χαρακτηριστικά χαμηλού ηχητικού σήματος. Η δεύτερη αποτελείται από 566 υψηλού επιπέδου σε σημασία περιγραφής (όπως τα όργανα, οι διαθέσεις και οι αρμονικές ιδιότητες). Η τρίτη αποτελείται από 13 Mel Frequency Cepstral Coefficients (MFCCs) που χρησιμοποιούνται για τη φωνητική αναγνώριση αλλά χρησιμοποιούνται και για ταξινόμηση μουσικής. Η τέταρτη ομάδα, με τη χρήση μιας εξελικτικής στρατηγικής επιλογής των χαρακτηριστικών, περιέχει τα βελτιστοποιημένα σύνολα των χαρακτηριστικών. Όσον αφορά την προσέγγιση με τα στατιστικά των playlist για τις τέσσερις ομάδες, χρησιμοποιήθηκαν 2 dataset από δημόσιες πηγές και 2 μέθοδοι κανονικοποίησης.

Όσον αφορά τις τεχνικές ταξινόμησης χρησιμοποίησαν Decision tree, Random Forest, Naive Bayes και Support Vector Machines. Επίσης επισημαίνουν ότι η κατανομή των ειδών και πιο συγκεκριμένα η κατανομή των ειδών μεταξύ των στυλ μουσικής δεν είναι ισορροπημένη. Έτσι τα μοντέλα ταξινόμησης αξιολογούνται με το ισορροπημένο σφάλμα σχετικής ταξινόμησης (balanced relative classification error). Τα αποτελέσματα έδειξαν ότι τα μοντέλα που δημιουργήθηκαν με βάση τις playlist σε αντίθεση με σύνολα χαρακτηριστικών που βασίζονται σε ήχο φαίνεται να είναι πιο αποδοτικά, όσον αφορά τη διαδικασία της ταξινόμησης σε

περισσότερα από τα μισά είδη.

Υπάρχουν επίσης σημαντικές έρευνες όπως των Ikeda, Oku και Kawagoe (2016), οι οποίοι χρησιμοποιούν τις μεταβάσεις ακουστικών χαρακτηριστικών για την αυτόματη δημιουργία playlist. Στην συγκεκριμένη έρευνα το επόμενο τραγούδι επιλέγεται με φυσική μετάβαση από το τρέχων. Οι αρχικές αξιολογήσεις της συγκεκριμένης έρευνας δείχνουν ότι η προτεινόμενη μέθοδος είναι πιο αποτελεσματική σε σύγκριση με άλλες μεθόδους όπως η τυχαία επιλογή και οι μέθοδοι του πλησιέστερου γείτονα. Ιδιαίτερα σημαντική ήταν και η προσέγγιση των Andrieu και Haus (2006), οι συγκεκριμένοι εστίασαν στην εξέταση των ακουστικών συνηθειών. Δημιούργησαν αλγόριθμους για την παρακολούθηση των συνηθειών ακρόασης και σχημάτισαν ένα μοντέλο ακροατή το οποίο χρησιμοποιήθηκε για την δημιουργία αυτόματων playlist. Άλλη μια έρευνα με μεγάλη αξία είναι των Chi, Tsai, Lai και Hsu (2007), όπου η βασική τους ιδέα ήταν η συλλογή της συμπεριφοράς ενός χρήστη (αξιολόγηση, παράβλεψη και επανάληψη τραγουδιού) κατά την αναπαραγωγή της μουσικής. Με αυτόν τον τρόπο γίνεται εκμάθηση των μουσικών συναισθημάτων του χρήστη και μέσα από το Reinforcement learning επιτυγχάνεται η δημιουργία αυτόματων playlist.

Β'.2 Σύγκριση εργασίας με διάφορες έρευνες

Οι παραπάνω έρευνες βοήθησαν αρκετά στην κατανόηση κάποιων προβλημάτων κατηγοριοποίησης μουσικής. Στις συγκεκριμένες έρευνες αναλύθηκαν διαφορετικές προσεγγίσεις για την επίλυση του προβλήματος. Η κατηγοριοποίηση της μουσικής και ειδικότερα η δημιουργία μουσικών λιστών (playlist) μπορεί να προσεγγιστεί μέσα από πάρα πολλά χαρακτηριστικά. Η συγκεκριμένη εργασία βασίστηκε αρκετά στη λογική και στη φιλοσοφία απαραίτητων βημάτων που αναλύονται στις παραπάνω έρευνες, όπως η εξαγωγή των δεδομένων, η επεξεργασία τους αλλά και επιλογή χαρακτηριστικών κατηγοριοποίησης. Σημαντικά κοινά στοιχεία ανάμεσα στις έρευνες που αναλύθηκαν με τη συγκεκριμένη εργασία είναι :

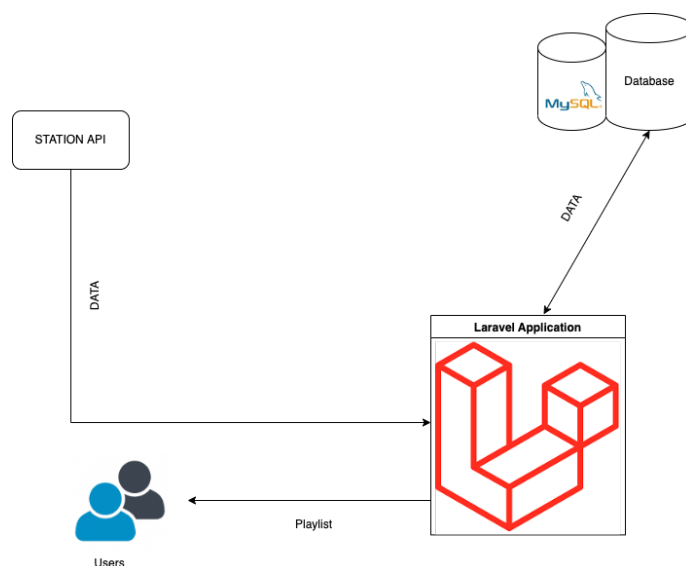
- Αναζήτηση επόμενου υποψήφιου τραγουδιού, δημιουργία playlist με συνέχεια.
- Δημιουργία λιστών αναπαραγωγής από μεγαλύτερες λίστες αναπαραγωγής .
- Οι λίστες των τραγουδιών που χρησιμοποιούνται για να δημιουργηθούν μικρότερες playlist επιλέγονται από χρήστες. Στην περίπτωση μας από τους παραγωγούς των σταθμών.

- Τα δεδομένα αντλήθηκαν από δημόσιες πηγές APIs (τις σελίδες των διαδικτυακών σταθμών). Συγκεκριμένα από πέντε πηγές και ενημερώνονται σε πραγματικό χρόνο.

Πέρα από τις παραπάνω ομοιότητες η προσέγγιση της παρούσας εργασίας είναι αρκετά διαφορετική από τις έρευνες που αναλύθηκαν στο συγκεκριμένο κεφάλαιο. Η πιο σημαντική διαφορά είναι στα χαρακτηριστικά που επιλέχθηκαν για τη δημιουργία της αυτόματης παραγωγής μουσικών λιστών. Στις παραπάνω έρευνες επιλέχθηκαν χαρακτηριστικά όπως η δημοτικότητα, ο ρυθμός, τα χαρακτηριστικά του ηχητικού σήματος και η εμφάνιση των καλλιτεχνών στις λίστες. Στην περίπτωση της συγκεκριμένης εργασίας σημαντικός παράγοντας είναι η χρονική στιγμή που έχει αναπαραχθεί το κάθε τραγούδι σε κάθε σταθμό. Στη βάση όλα τα τραγούδια έχουν γραφτεί με χρονική αλληλουχία. Με αποτέλεσμα να κρατάμε τη σειρά που μεταδόθηκαν τα τραγούδια για κάθε σταθμό. Σημασία για τον αλγόριθμο που προτείνεται στην παρούσα δουλειά, είναι να δημιουργηθούν ζευγάρια μουσικών κομματιών. Πιο συγκεκριμένα υπολογίζουμε το ποσοστό εμφάνισης ενός τραγουδιού όταν προηγείται ένα συγκεκριμένο τραγούδι. Κατά ένα μεγάλο ποσοστό βασιζόμαστε στις επιλογές των ραδιοφωνικών παραγωγών με αποτέλεσμα οι λίστες που έχουν δημιουργηθεί να εξαρτώνται από τις επιλογές τους. Για την αξιολόγηση των αποτελεσμάτων χρησιμοποιήθηκαν ερωτηματολόγια που δόθηκαν σε χρήστες με στοχευμένες ερωτήσεις που βοήθησαν σημαντικά να βγούνε τα απαραίτητα συμπεράσματα για τη συγκεκριμένη δουλειά. Στις προϋπάρχουσες έρευνες που αναλύθηκαν, τα αποτελέσματα των αλγορίθμων και των διαφορετικών τεχνικών αξιολογήθηκαν με βάση συγκεκριμένες τεχνικές όπως *balanced relative classification error*. Περισσότερες πληροφορίες για τη διαδικασία δημιουργίας αυτόματων playlist και την περιγραφή των βημάτων αναφέρονται στο κεφάλαιο Δ'.

Κεφάλαιο Γ'

Ανάπτυξη εφαρμογής



Σχήμα Γ'.1: Στο συγκεκριμένο σχήμα παρουσιάζεται η βασική λειτουργία της εφαρμογής.

Η ροή δεδομένων από τους διαδικτυακούς ραδιοφωνικούς σταθμούς προς την εφαρμογή που γίνεται η επεξεργασία των δεδομένων, στη συνέχεια η αποθήκευση τους στη βάση δεδομένων και τέλος η παρουσίαση της πληροφορίας στον χρήστη μέσω της εφαρμογής.

Γ'.1 Εργαλεία και Τεχνολογίες

Για τη δημιουργία ενός πληροφοριακού συστήματος είναι απαραίτητο να χρησιμοποιηθούν τεχνολογίες και εργαλεία. Με τον καιρό όσο αναπτύσσεται η τεχνολογία εμφανίζονται όλο και περισσότερα εργαλεία που βοηθούν τους χρήστες για να δημιουργήσουν καλύτερα άλλα και με περισσότερη ευκολία πληροφοριακά συστήματα. Πιο συγκεκριμένα για τη δημιουργία μιας web εφαρμογής χρειάζονται αρκετά. Όπως ένας διακομιστής (Στην Αγγλική κι εφεξής στο κείμενο για συντομία server) που φιλοξενεί την εφαρμογή, η MySQL ¹ που είναι ένα από τα πιο δημοφιλή συστήματα διαχείρισης βάσης δεδομένων και η γλώσσα προγραμματισμού PHP ² η οποία χρησιμοποιείται για τη δημιουργία διαδικτυακών ιστοσελίδων και εφαρμογών. Πέρα από τις βασικές τεχνολογίες χρειάζονται πλήρεις γνώσεις λειτουργικών συστημάτων. Η παρούσα διαδικτυακή εφαρμογή δημιουργήθηκε με τον συνδυασμό (LAMP). Πιο συγκεκριμένα σε linux ³ περιβάλλον έγινε η εγκατάσταση του server και ρυθμίστηκαν όλες οι διαδικασίες και τα αναγκαία αρχεία για την ομαλή λειτουργία του. Εγκαταστάθηκε η γλώσσα PHP τόσο στο τερματικό όσο και στον server. Έγινε εγκατάσταση της MySQL αλλά και του PhpMyAdmin ⁴ (εργαλείο διαχείρισης για την MySQL) (What is LAMP Stack?, 2022). Τέλος χρειάστηκε να εγκατασταθεί και ο composer ⁵ ο οποίος είναι ένας διαχειριστής πακέτων για τη γλώσσα PHP. Στις επόμενες ενότητες γίνεται αναλυτική περιγραφή των εργαλείων και των τεχνολογιών που χρησιμοποιήθηκαν.

Γ'.1.1 Apache

Ο Apache HTTP Server είναι ένας διαδικτυακός server ή αλλιώς εξυπηρετητής ανοιχτού κώδικα (open-source), ο οποίος είναι επίσης γνωστός ως “httpd” και Apache. Είναι ένας από τους πιο δημοφιλείς εξυπηρετητές ιστού από το 1996 και λειτουργεί σε περιβάλλοντα Windows και Unix. Δημιουργήθηκε και συντηρείται από την Apache Software Foundation (Apache, 2022).

¹<https://www.mysql.com/>

²<https://www.php.net/>

³<https://www.linux.org/>

⁴<https://www.phpmyadmin.net/>

⁵<https://getcomposer.org/>

Γ'.1.2 MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων που μετρά περισσότερες από 11 εκατομμύρια εγκαταστάσεις. Αποτελεί ελεύθερο λογισμικό και μπορεί να χρησιμοποιηθεί για ερασιτεχνικούς και επαγγελματικούς σκοπούς αντίστοιχα. Το πρόγραμμα τρέχει έναν εξυπηρετητή (server) παρέχοντας πρόσβαση πολλών χρηστών σε ένα σύνολο βάσεων δεδομένων. Σύμφωνα με τον DuBois, P. (2008) υπάρχουν αρκετοί λόγοι για να χρησιμοποιήσει κάποιος προγραμματιστής την MySQL. Τρεις βασικοί λόγοι είναι η ταχύτητα που προσφέρει, πολλοί προγραμματιστές αναφέρουν ότι ίσως είναι το πιο γρήγορο σύστημα βάσης δεδομένων. Είναι αρκετά εύκολη στη χρήση, προσφέρει υψηλές αποδόσεις σχετικά, παρόλο που είναι ένα απλό σύστημα βάσεων δεδομένων το οποίο είναι αρκετά εύκολο στη διαχείριση. Λειτουργεί πάνω στην Structured Query Language (SQL⁶) που είναι η βασική γλώσσα που χρησιμοποιείται στα περισσότερα μοντέρνα συστήματα βάσεων.

Γ'.1.3 Η γλώσσα PHP

Η PHP είναι μια γλώσσα προγραμματισμού η οποία χρησιμοποιείται για τη δημιουργία ιστοσελίδων και διαδικτυακών εφαρμογών. Είναι μία ανοικτού περιεχομένου (open source) scripting γλώσσα και μπορεί να ενσωματωθεί μέσα σε κώδικα HTML (What is PHP?, 2022). Η PHP είναι μια εξαιρετική επιλογή για ανάπτυξη από την πλευρά του διακομιστή (server) επειδή είναι γρήγορη αλλά επίσης επειδή μπορεί εύκολα να ενσωματωθεί με MySQL βάσεις δεδομένων. Αυτό της δίνει τη δυνατότητα για τη δημιουργία δυναμικών ιστοσελίδων και εφαρμογών (What is PHP? The PHP Programming Language Meaning Explained, 2022). Διαθέτει επίσης βιβλιοθήκες που μπορούν να χρησιμοποιηθούν για να δημιουργηθούν εφαρμογές web εύκολα και γρήγορα. Ένα άλλο μεγάλο πλεονέκτημα της χρήσης PHP από την πλευρά του διακομιστή είναι ότι μπορεί να χρησιμοποιηθεί η ίδια βάση κώδικα για να δημιουργηθούν τόσο δυναμικές όσο και στατικές σελίδες. Στην αρχή τα αρχικά της σήμαιναν (Personal Home Page) πλέον (Hypertext Preprocessor). Η τελευταία έκδοση μέχρι στιγμής είναι η 8.1.7, αλλά στην παρούσα εργασία χρησιμοποιήθηκε η έκδοση 7.4.14 . (php, 2022)

⁶https://www.w3schools.com/sql/sql_intro.asp

Γ'.1.4 Τεχνολογίες για τη Δημιουργία του UI.

Το εικαστικό κομμάτι της εφαρμογής χτίστηκε από σελίδα σε σελίδα με τη χρήση HTML ⁷ χρησιμοποιώντας μόνο το Bootstrap ⁸ για τη σωστή στοίχιση των στοιχείων στις σελίδες. Η μορφοποίηση των σελίδων έγινε με τη χρήση του Scss ⁹ που είναι μία τεχνολογία που βοηθάει στην καλύτερη δόμηση και δημιουργία Css ¹⁰ αρχείων. Χρησιμοποιήθηκε το Scss κυρίως γιατί διαθέτει περισσότερα χαρακτηριστικά (features) που αυτή τη στιγμή δεν υπάρχουν στο Css. Σημαντικό είναι να αναφερθεί ότι δεν χρησιμοποιήθηκε πουθενά έτοιμος κώδικας ούτε κάποιο έτοιμο template.

Γ'.1.5 Το laravel

Το laravel είναι ένα από τα πιο γνωστά php framework, το οποίο χρησιμοποιείται για τη δημιουργία διαδικτυακών εφαρμογών και ιστοσελίδων. Διαθέτει ενισχυμένα πακέτα και βιβλιοθήκες τα οποία είναι φτιαγμένα για την καλύτερη εξυπηρέτηση των προγραμματιστών. Το Laravel σε σχέση με άλλα framework, δίνει τη δυνατότητα για τη δημιουργία από μια μικρή ιστοσελίδα, μέχρι ένα τεράστιο ηλεκτρονικό κατάστημα (e-shop), αφού αυτό που το ξεχωρίζει είναι ότι είναι εύελικτο. Βοηθάει στην ασφάλεια της εφαρμογής ή της ιστοσελίδας, μας προτρέπει για καλύτερη δομή της εφαρμογής, με τελικό σκοπό ο προγραμματιστής να μπορεί τελικά να τη διαχειριστεί ευκολότερα όσο μεγαλώνει και όσο προστίθενται νέα features. Συνήθως είναι πιο «καλογραμμένο» από την αρχική βασική γλώσσα και απλοποιεί σημαντικά την ανάγνωση/κατανόηση της. Το laravel υλοποιεί την αρχιτεκτονική MVC. Η συγκεκριμένη αρχιτεκτονική λογισμικού (ολογράφως Model-View-Controller) μπορεί να βοηθήσει αρκετά στην καλύτερη κατανομή των αρχείων και γενικότερα στον διαχωρισμό της εφαρμογής σε μέρη. Στο μοντέλο αυτό η εφαρμογή διαιρείται σε τρία διασυνδεδεμένα μέρη ώστε να διαχωριστεί η παρουσίαση της πληροφορίας στον χρήστη από την μορφή που έχει αποθηκευτεί στο σύστημα (How Laravel implements MVC and how to use it effectively, 2022). Το κύριο μέρος του MVC μοντέλου είναι το αντικείμενο Model (Μοντέλο) το πρώτο κατά σειρά στο όνομα το οποίο αναπαριστά - προσομοιώνει ένα αντικείμενο ή ένα σύνολο αντικειμένων (Collection) ενός πίνακα της βάσης. Οι ενέργειες των άλλων δύο δρουν πάνω σε αυτό. Ο εκλεκτής (Στην Αγγλική και εφεξής στο κείμενο για συντομία, Controller) διαχειρίζεται την πληροφορία από

⁷<https://html.com/>

⁸<https://getbootstrap.com/>

⁹<https://sass-lang.com/>

¹⁰<https://www.w3.org/Style/CSS/Overview.en.html>

τη βάση δεδομένων, δηλαδή μπορεί να στείλει εντολή να δημιουργηθεί ένα μοντέλο είτε να γίνει ενημέρωση πάνω σε αυτό. Επίσης συγκεντρώνει την πληροφορία και διαχειρίζεται ποια πληροφορία θα περάσει στο view. Το view χρησιμοποιείται μόνο για την ανάδειξη - παρουσίαση της πληροφορίας στους χρήστες του συστήματος (Why Laravel Framework is the Best Choice for PHP Web Development, 2022).

Το ερώτημα που προκύπτει είναι τελικά γιατί να επιλεχθεί το laravel για τη δημιουργία της εφαρμογής. Τι μας πρόσφερε και κατά πόσο η διαδικασία ανάπτυξης της εφαρμογής βοηθήθηκε με τα εργαλεία που προσφέρει. Υπήρξαν πέντε βασικοί λόγοι που επιλέχθηκε το συγκεκριμένο framework:

1. PHP : Ίσως ο σημαντικότερος λόγος .Ως γνωστόν το Laravel είναι ένα framework της PHP, βοηθάει πολύ στην ανάπτυξη κώδικα αφού πολλά πακέτα που χρησιμοποιήθηκαν κάνουν απλούστερη τη δημιουργία της εφαρμογής. Η τεράστια γκάμα βιβλιοθηκών και τη μεγάλη κοινότητα (Στην Αγγλική κι εφεξής στο κείμενο για συντομία,community) που το χρησιμοποιεί, είναι ένας λόγος που κάνει αρκετούς προγραμματιστές να το επιλέξουν.
2. Η ευελιξία που προσφέρει, δηλαδή το πόσο εύκολα μπορεί να δημιουργηθεί μια εφαρμογή χωρίς περιορισμούς. Ξεκινώντας από ένα αρχικό στάδιο φτάνοντας κατά τη διαδικασία ανάπτυξης να μπορούν να προστεθούν αρκετά features ακόμα και να αλλάξει όλος ο σκελετός της εφαρμογής.
3. Ο χρόνος υλοποίησης : Με το Laravel προσφέρεται ένα μεγάλο πλεονέκτημα στον χρόνο υλοποίησης, αφού μέσα από αυτό πολλές ενέργειες μπορούν να γίνουν με πολύ πιο γρήγορους ρυθμούς. Τα εργαλεία που διαθέτει απλοποιούν κατά ένα μεγάλο ποσοστό τη διαδικασία ανάπτυξης κώδικα που χρειάζεται μια εφαρμογή, με αποτέλεσμα να οδηγεί τον προγραμματιστή μέσα από τους κανόνες του στη δημιουργία κώδικα που είναι πιο απλοποιημένος, πιο ευανάγνωστος και φυσικά στη γρηγορότερη ανάπτυξή του (Why Choose Laravel for Your Next Web Project, 2022).
4. Η αρχιτεκτονική MVC (model-view-controller) : Η συγκεκριμένη αρχιτεκτονική στην οποία βασίζεται το Laravel, προσφέρει υψηλές αποδόσεις και ασφάλεια στις εφαρμογές και δίνει τη δυνατότητα για πιο γρήγορη ανάπτυξη σε βάθος χρόνου.
5. Τελευταίος λόγος αλλά εξίσου σημαντικός, είναι ότι αποτελεί ένα από τα πιο δημοφιλή Framework παγκοσμίως, ειδικότερα όταν αναφερόμαστε στην PHP. Αυτό δίνει δύο βασικά πλεονεκτήματα. Αρχικά ότι έχει ένα πολύ μεγάλο community και στη συνέχεια

ότι υπάρχει αρκετό υλικό εκμάθησης στο διαδίκτυο και αρκετά παραδείγματα για μπορέσει ο κάθε προγραμματιστής να νιώσει οικία με το εργαλείο αλλά και να ανατρέξει σε λύσεις για διάφορα προβλήματα. Ακόμα ιδιαίτερα σημαντικό είναι ότι είναι ένα εργαλείο ανοιχτού κώδικα, όπου η ομάδα που το διαχειρίζεται το συντηρεί συχνά και προσθέτει συνέχεια καινούργια features (An Overview of the Best Laravel Security Practices, 2022).

Γ'.2 Πηγή δεδομένων και διαχείριση

Συχνά στα πληροφοριακά συστήματα συναντάμε διάφορες πηγές δεδομένων. Πηγές δηλαδή που προσφέρουν τα δεδομένα που δέχεται το σύστημα και συνήθως περνάνε από μία φάση επεξεργασίας παρέχοντας αποτελέσματα. Στη συγκεκριμένη εφαρμογή υλοποιείται ένα σύστημα πραγματικού χρόνου το οποίο δέχεται συνεχώς δεδομένα, τα επεξεργάζεται και παρέχει στους χρήστες τις απαραίτητες πληροφορίες. Πιο συγκεκριμένα τα δεδομένα αυτά παρέχονται από διαδικτυακούς ραδιοφωνικούς σταθμούς. Οι συγκεκριμένοι σταθμοί παρέχουν κάποια τελικά σημεία (Στην Αγγλική και εφεξής στο κείμενο για συντομία, endpoints) API. Τα endpoints επιστρέφουν μια απάντηση είτε σε μορφή XML¹¹ είτε σε μορφή JSON¹² στα οποία βρίσκονται απαραίτητες πληροφορίες για το τραγούδι που εκπέμπεται τη συγκεκριμένη χρονική στιγμή (What is an API endpoint?, 2022). Η εφαρμογή έχει 5 σταθμούς ως πηγή δεδομένων. Είναι σημαντικό να αναφερθεί ότι σημαντικός παράγοντας για την επιλογή τους ήταν η μορφή που παρέχουν τα δεδομένα και κατά πόσο θα ήταν εύκολη η διαχείριση τους.

Για να είναι επιτυχής η αποθήκευση αυτών των δεδομένων, με τη σωστή χρονολογική σειρά αλλά και για να κρατηθούν όλα τα τραγούδια που έχουν αναπαράγει οι συγκεκριμένοι ραδιοφωνικοί σταθμοί έπρεπε να παρθούν κάποιες αποφάσεις. Αρχικά δημιουργήθηκε ένα Cron Job. Μια διαδικασία η οποία τρέχει ανά script σε ορισμένο χρόνο χωρίς την παρέμβαση κάποιου χρήστη έχοντας ως σκοπό να ενημερώνει ανά τακτά χρονικά διαστήματα τη βάση δεδομένων (Scheduling Tasks with Cron Jobs, 2022). Η συγκεκριμένη διαδικασία δημιουργήθηκε μέσα από το Laravel αφού σου δίνει τη δυνατότητα να γράφεις κώδικα για τη διαχείριση προγραμματισμένων εργασιών στο server. Ορίζοντας τις μεταβλητές μέσα στο Laravel απαιτώντας μόνο τη μία καταχώρηση cron στον διακομιστή (Task Scheduling, 2022).

¹¹<https://www.w3.org/standards/xml/core>

¹²<https://www.json.org/json-en.html>

Listing C.1: *code example*

```

public function handle()
{
    $stations = stations::all();
    foreach ($stations as $station){
        $job=(new StationsController($this->songRepository))
            ->show($station);
        \Log::info("Cron is working fine!". $station);
    }
}

```

Στο συγκεκριμένο μπλόκ κώδικα εικόνα παρουσιάζεται ο κώδικας διαχείρισης της προγραμματισμένης διεργασίας.

Listing C.2: *bash command*

```

[2022-07-13T17:06:00+00:00] Execution #18292 output:
[2022-07-13T17:06:00+00:00] Running scheduled command: '/usr/local/Cellar/
php@7.4/7.4.14_1/bin/php' 'artisan' song:cron > '/dev/null' 2>&1

```

Σημαντική παράμετρος στη συγκεκριμένη διεργασία είναι ο χρόνος. Έπρεπε να βρεθεί το κατάλληλο χρονικό διάστημα για να έχει νόημα το συγκεκριμένο script. Στόχος ήταν να μπορέσει να καλύψει όλες τις περιπτώσεις και να μειωθεί στο μεγαλύτερο ποσοστό το σενάριο σφάλματος. Μετά από αρκετή μελέτη στη συμπεριφορά των σταθμών, παρατηρήθηκε ότι οι περισσότεροι σταθμοί ανά ταχτά χρονικά διαστήματα παίζουν διαφημίσεις οι οποίες παρουσιάζουν μια συγκεκριμένη ένδειξη στο σώμα τον JSON και XML αντίστοιχα. Σε περίπτωση που το πρόγραμμα συναντήσει μια τέτοια περίπτωση, η διεργασία σταματάει και επαναλαμβάνεται μετά από ενάμιση λεπτό. Μετά από αρκετά πειράματα το συγκεκριμένο χρονολογικό διάστημα φαίνεται να καλύπτει όλες τις περιπτώσεις με μηδενικό ποσοστό λάθους. Το ίδιο χρονολογικό όριο καλύπτει αποτελεσματικά την περίπτωση που έχει ξεκινήσει η αναπαραγωγή καινούργιου τραγουδιού. Κάθε φορά που τρέχει το cronjob γίνεται έλεγχος αν στον συγκεκριμένο σταθμό έχει αλλάξει το μουσικό κομμάτι φέρνοντας κάθε φορά την τελευταία εγγραφή στη βάση για τον συγκεκριμένο σταθμό. Ο cron διαβάζει από τα δεδομένα τον τίτλο του τραγουδιού που είναι στον αέρα τη συγκεκριμένη στιγμή στον συγκεκριμένο σταθμό και αν ταυτίζεται με την προηγούμενη εγγραφή τότε δεν θα γίνει καμία ενέργεια στη βάση και θα πυροδοτηθεί ξανά σε ενάμιση λεπτό. Έτσι για να υπάρξει μια νέα εγγραφή στη βάση για τον συγκεκριμένο σταθμό θα πρέπει να τηρούνται οι παραπάνω προϋποθέσεις.

Πριν την αποθήκευση των δεδομένων στη βάση, υπάρχει μια διαδικασία επεξεργασίας αυτών από την αρχική μορφή που διαβάζονται από τους σταθμούς. Στη συγκεκριμένη εφαρμογή η επεξεργασία των δεδομένων έγινε με τέτοιο τρόπο ώστε να μπορούν να προστεθούν όλο και περισσότεροι σταθμοί. Βασική προϋπόθεση είναι τα δεδομένα τους να έρχονται σε παρόμοια μορφή με τους προϋπάρχοντες σταθμούς. Όπως αναφέρθηκε παραπάνω τα δεδομένα αρχικά έρχονται σε μορφή JSON και XML. Αυτό χώρισε τη διαδικασία επεξεργασίας των δεδομένων σε δύο βασικές περιπτώσεις μία για κάθε μορφή. Δημιουργήθηκαν έτσι δύο βασικές συναρτήσεις επεξεργασίας δεδομένων που καλούνται ανά περίπτωση.

Listing C.3: *Code of data processing and storing to database*

```
public function storeTypeTwo(stations $station)
{
    $url = $station->link;

    $response_json_data = file_get_contents($url);

    $data = json_decode($response_json_data, true, JSON_FORCE_OBJECT);
    $song_info = $data['nowPlaying'][0];

    if ($song_info['type'] == "Song") {
        $lastSong = Song::query()->where('station_id', '=', $station->id)
            ->orderBy('created_at', 'desc')->first();
        if ($lastSong != null
            and $name = $song_info['name'] == $lastSong->title) {
            $song = $lastSong;
            $artist = Artist::query()->where('song_id', '=', $song->id)->first();
        } else {
            $name = $song_info['name'];
            $artist = $song_info['artist'];
            $played_at = $song_info['played_date'];
            $seconds = ($song_info['length_in_secs']) / 60;
            $artwork = $song_info['artwork'];
            $duration = number_format((float)$seconds, 2, '.', '');
            $song = new Song([
                'title' => $name,
                'station_id' => $station->id,
                'duration' => $duration,
                'played_at' => $played_at,
```

```

        'artwork' => $artwork ,
    ]);
    $song->save();
    $artist = new Artist([
        'nickname' => $artist ,
        'song_id' => $song->id ,
    ]);

    $artist->save();

}
$lastSongs = Song::query()->where('station_id', '=', $station->id)
->latest()->take(5)->get();

return $data = [
    'latest' => $lastSongs ,
    'song' => $song ,
    'artist' => $artist ,

];
}
}

```

Γ'.3 Ανάλυση και σχεδιασμός εφαρμογής

Στα περισσότερα πληροφοριακά συστήματα εκτός από την υλοποίηση τους που είναι μια χρονοβόρα διαδικασία, σημαντική έμφαση πρέπει να δοθεί και στην ανάλυση των αναγκών. Το πρώτο βήμα που εΐθισται να γίνεται σε αυτές της περιπτώσεις είναι η καταγραφή των αναγκών. Το ίδιο έγινε και στην παρούσα εργασία. Αναζητήθηκαν οι ανάγκες που χρειάζεται να καλυφθούν και ορίστηκαν συγκεκριμένοι στόχοι. Η ανάλυση των αναγκών είναι ένα βήμα που δεν πρέπει να παραλείπετε αλλά πρέπει να δίνεται η απαραίτητη προσοχή διότι αν δεν δοθεί καταλήγουμε σε πληροφοριακά συστήματα όπου η χρήση τους είναι περιορισμένη και εμφανίζουν σοβαρές ελλείψεις αφού δεν ανταποκρίνονται στα αρχικά ζητούμενα. Οι βασικές ανάγκες που εμφανίστηκαν και έπρεπε να καλυφθούν είναι η εξής:

- Η συλλογή μεγάλου όγκου μουσικής όπου θα προστίθενται συνεχώς νέα τραγούδια.
- Να δημιουργηθεί ένα περιβάλλον που θα βοηθάει τον χρήστη και θα του παρέχει την

απαιτούμενη πληροφορία γρήγορα.

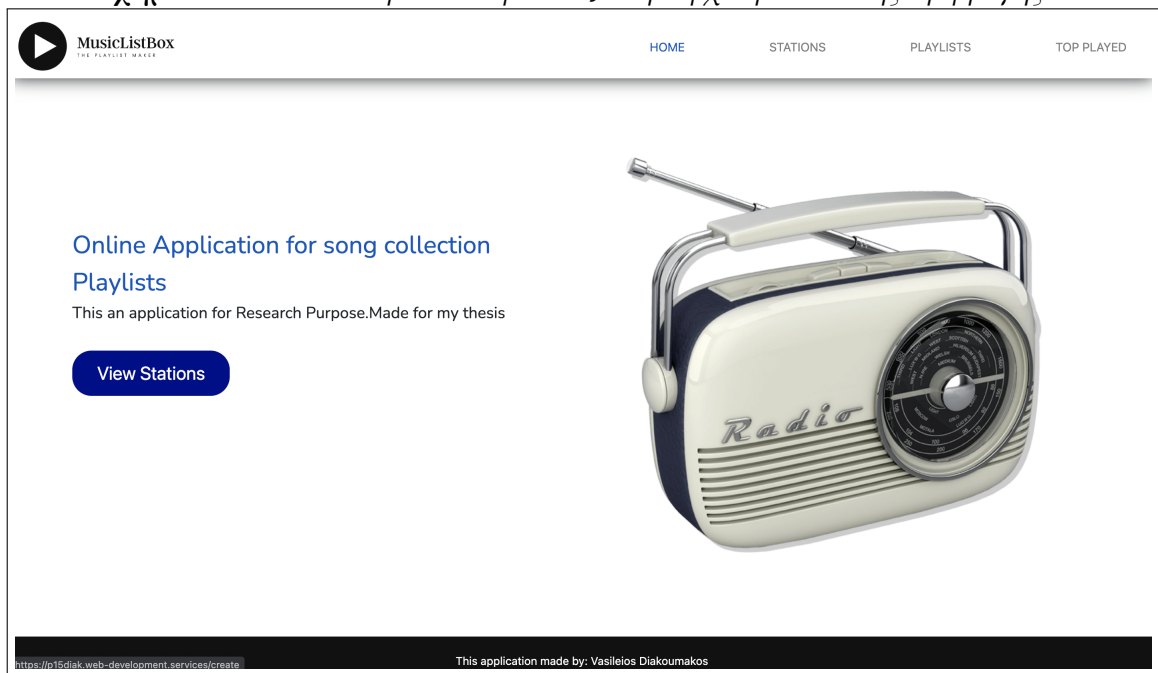
- Να δημιουργηθεί ένα περιβάλλον που θα μπορέσει να υποστηρίξει το ερευνητικό κομμάτι της εφαρμογής.
- Να υπάρξει διάδραση με τους χρήστες για να καταλήξουμε σε κάποια συμπεράσματα.

Αφού γίνει η καταγραφή των αναγκών, ξεκινάει η διαδικασία εύρεσης λύσεων. Έτσι το επόμενο στάδιο ήταν να βρεθεί ο τρόπος και τα μέσα υλοποίησης. Μετά από αρκετή σκέψη η κατάληξη ήταν η δημιουργία μιας διαδικτυακής εφαρμογής η οποία θα διαθέτει μία βάση δεδομένων για την υποστήριξη του μεγάλου όγκου πληροφοριών αλλά και να καλύπτει το διαδραστικό κομμάτι με τους χρήστες ώστε να βγουν τα απαραίτητα συμπεράσματα.

Στην παρούσα εργασία δόθηκε μεγάλο μέρος του χρόνου στην ανάλυση και στον καλύτερο σχεδιασμό. Ο σχεδιασμός είναι ένα στάδιο που προηγείται της υλοποίησης αφού όσο καλύτερος σχεδιασμός γίνει τόσο πιο εύκολη θα είναι και η υλοποίηση στη συνέχεια. Επίσης καταλήγουμε σε καλύτερες αποφάσεις - τρόπους ανάπτυξης. Σημαντικό είναι να σχεδιάζουμε ένα πληροφοριακό σύστημα - εφαρμογή ανάλογα με τις μελλοντικές ανάγκες, να προβλέψουμε ουσιαστικά προβλήματα που πρέπει να αντιμετωπιστούν ή λειτουργίες που μπορεί να εισαχθούν. Συχνά ο σχεδιασμός ταυτίζεται με τον χρόνο υλοποίησης, αφού πολλές φορές ψάχνουμε να βρούμε τον πιο σύντομο δρόμο που ταυτόχρονα θα είναι και ο πιο ορθός (The Benefits of System Analysis, 2022). Πιο συγκεκριμένα η παρούσα εφαρμογή σχεδιάστηκε με γνώμονα να είναι φιλική προς τους χρήστες, να παρέχει γρήγορη πληροφορία χωρίς να υπάρχουν αχρεΐαστα βήματα, όπως σύστημα δημιουργίας λογαριασμού και σύνδεσης στην πλατφόρμα. Έτσι εμφανίστηκε το ζήτημα του τι πρέπει να σχεδιαστεί πρώτο, η βάση δεδομένων ή το εικαστικό κομμάτι της εφαρμογής. Τελικά επιλέχθηκε να γίνει πρώτα η σχεδίαση της βάσης δεδομένων με τη λογική ότι στη συνέχεια ο εικαστικός σχεδιασμός της εφαρμογής θα είναι ευκολότερος δίνοντας το πλεονέκτημα ότι υπάρχουν ήδη διαθέσιμα δεδομένα για πειραματισμό. Ο σχεδιασμός της βάσης είναι μια από τις βασικότερες διαδικασίες αφού αν υπάρξει κάποιο λάθος η κάποια λανθασμένη πρόβλεψη μπορεί να περιορίσει αρκετά τις δυνατότητες της εφαρμογής. Αφού ολοκληρώθηκε το βασικό χτίσιμο της βάσης στη συνέχεια έπρεπε να σχεδιαστεί το εικαστικό κομμάτι της εφαρμογής και λειτουργίες (front-end development). Το οποίο ασχολείται με τη δημιουργία λειτουργιών και σελίδων τα οποία έρχονται σε επαφή με τον χρήστη (Front End vs. Back End: What's the Difference?, 2022).

Η αρχική οθόνη της εφαρμογής είναι η Home (Σχήμα Γ'2.) στην οποία ο χρήστης βλέπει ένα κουμπί με το όνομα stations στο κέντρο της σελίδας και μια μικρή περιγραφή της εφαρμογής. Στο πάνω μέρος της εφαρμογής βρίσκεται το μενού στο οποίο μπορεί να δει όλες τις επιλογές που έχει για να περιηγηθεί. Περνώντας τον κέρσορα πάνω από τις επιλογές αλλάζει το χρώμα της λέξης.

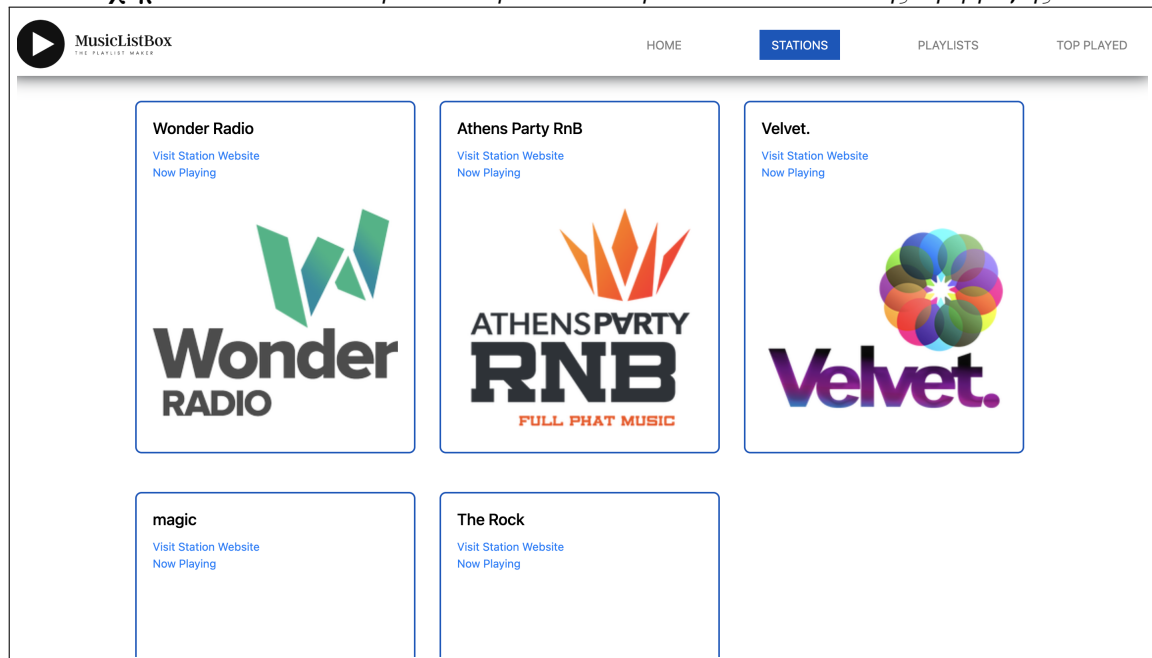
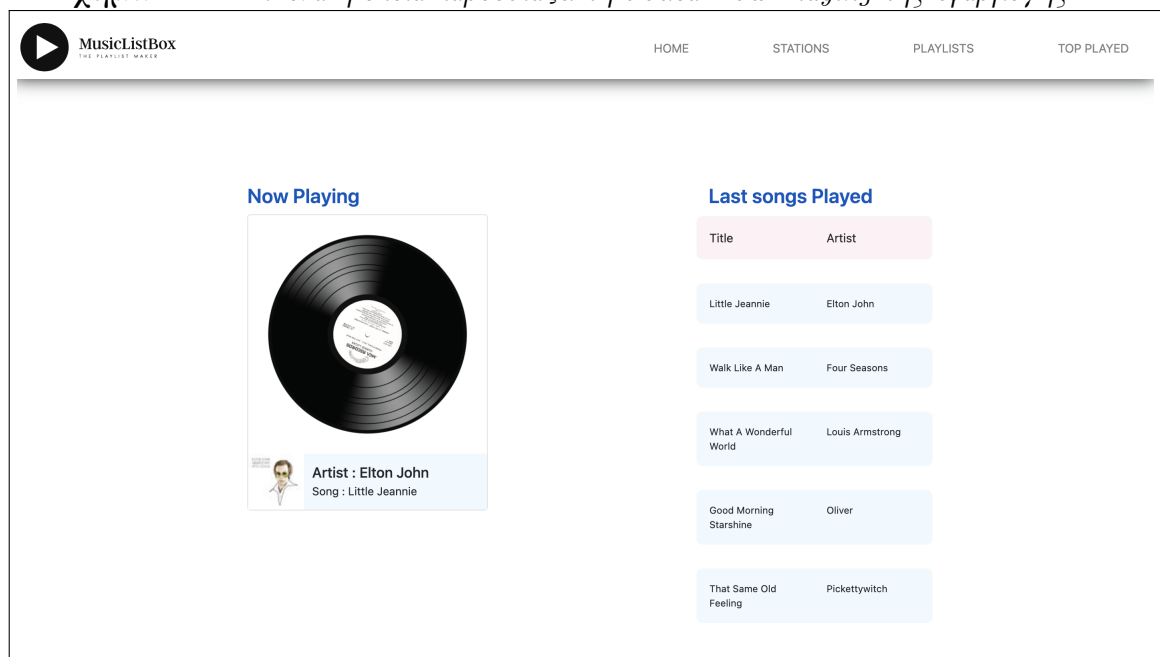
Σχήμα Γ'.2: Εικόνα η οποία παρουσιάζει την αρχική σελίδα της εφαρμογής .



Πατώντας το κουμπί View Stations στην αρχική σελίδα ή το stations στο μενού ο χρήστης μεταφέρεται σε μία λίστα όπου εμφανίζονται όλοι οι διαθέσιμοι σταθμοί (εικόνα Γ'3). Εκεί για κάθε σταθμό έχει δύο επιλογές, η πρώτη είναι να επισκεφτεί την επίσημη σελίδα του σταθμού και η δεύτερη είναι να μεταβεί στην single σελίδα του σταθμού για να δει βασικές πληροφορίες.

Επιλέγοντας το Now Playing ο χρήστης βλέπει το τραγούδι που παίζεται αυτή τη στιγμή στον σταθμό. Μαζί με το όνομα του καλλιτέχνη και το artwork του. Στα δεξιά όπως φαίνεται στο σχήμα Γ'4. βρίσκεται μία λίστα με τα 5 τελευταία τραγούδια που έχει παίξει ο κάθε σταθμός.

Στο μενού υπάρχουν ακόμα δυο επιλογές για να περιηγηθεί. Η μία είναι η σελίδα Playlists στην οποία ο χρήστης μπορεί να δει 10 τραγούδια από το σύνολο όλων των τραγουδιών για κάθε σελίδα (Σχήμα Γ'5). Για κάθε τραγούδι υπάρχουν τρεις στήλες (columns). Στην πρώτη είναι ο τίτλος του τραγουδιού στη δεύτερη το όνομα του καλλιτέχνη και στη τρίτη το κουμπί

Σχήμα Γ'.3: Εικόνα η οποία παρουσιάζει την σελίδα *stations* της εφαρμογής .Σχήμα Γ'.4: Εικόνα η οποία παρουσιάζει τη σελίδα *Now Playing* της εφαρμογής .

Σχήμα Γ'.5: Εικόνα η οποία παρουσιάζει τη διαδικασία δημιουργίας

Create Playlist

Search a song to create a playlist

Rock y

Please select the playlist length : 7

Song Name	Artist	Get Recommendation
We Will Rock You	Queen	Playlist
Rock You Like A Hurricane	Scorpions	Playlist
Rock Your Baby	George McCrae	Playlist

Σχήμα Γ'.6: Εικόνα η οποία παρουσιάζει την playlist που δημιουργήθηκε.

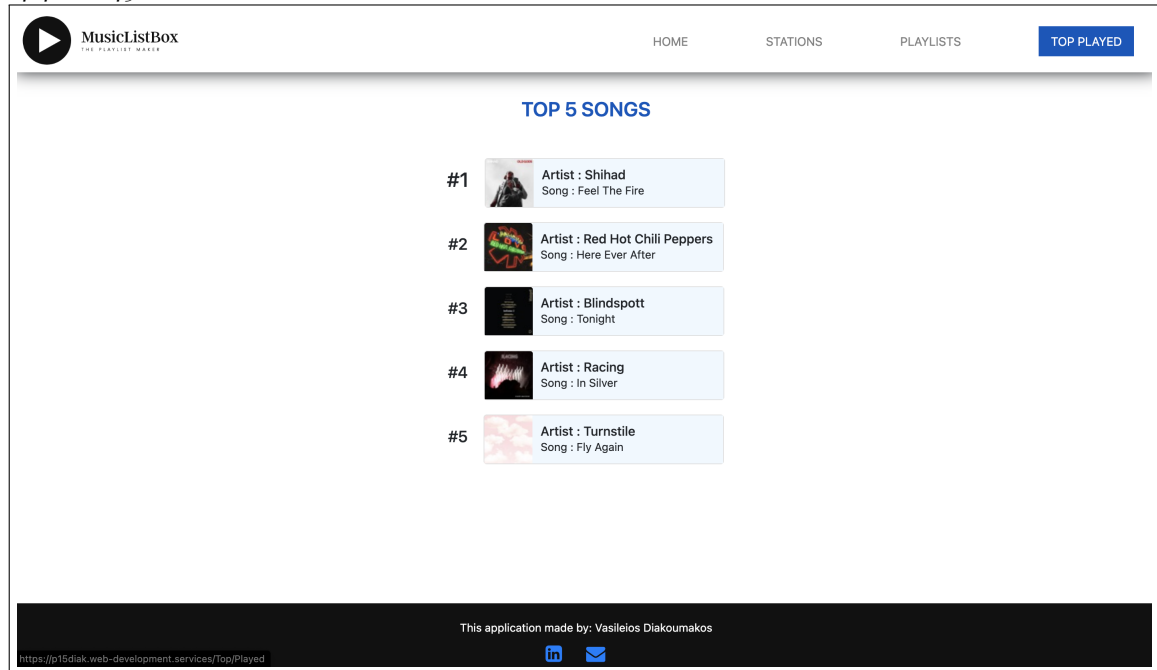
[Go back](#)

Auto Generated Playlist :

Song Name	Artist
Blow Up The Outside World	Soundgarden
Out Of The Black	Royal Blood
From The Inside	Linkin Park
Welcome To Paradise	Green Day
Dirty Deeds Done Dirt Cheap	AC/DC
What's My Age Again ?	Blink-182
Chop Suey!	System Of A Down

playlist που του δημιουργεί την playlist για το συγκεκριμένο τραγούδι. Για τη βοήθεια του υπάρχει μια μπάρα αναζήτησης (search bar) πάνω από τον πίνακα (table) το οποίο ανανεώνει

Σχήμα Γ'.7: Εικόνα η οποία παρουσιάζει τη λίστα τραγουδιών με το μεγαλύτερο ποσοστό εμφάνισης



τη λίστα ζωντανά, φιλτράροντας την ανάλογα με το τι έχει αρχίσει να πληκτρολογεί. Έτσι ο χρήστης μπορεί ανάμεσα από χιλιάδες τραγούδια να ψάξει αυτό ακριβώς που θέλει. Στην ίδια οθόνη υπάρχει πεδίο στο οποίο μπορεί να δώσει το σύνολο των τραγουδιών που θέλει στην playlist, που θα του δημιουργήσει ο αλγόριθμος της εφαρμογής. Ο χρήστης μπορεί αν επιλέξει από 1 έως 25 τραγούδια και κάθε φορά που μπαίνει στη συγκεκριμένη σελίδα αυτό έχει σαν προεπιλεγμένο αριθμό το 5. Πατώντας το κουμπί playlst φορτώνει μια καινούργια σελίδα (Σχήμα Γ'6) που περιέχει τη λίστα των τραγουδιών της Playlist που δημιουργήθηκε. Επίσης στο πάνω μέρος αριστερά της σελίδας υπάρχει το κουμπί Go back, που του φορτώνει την προηγούμενη σελίδα οπότε μπορεί να επαναλάβει τη διαδικασία.

Η τελευταία επιλογή στο μενού είναι η σελίδα TOP PLAYED, στην οποία ο χρήστης μπορεί να δει τη λίστα τραγουδιών με το μεγαλύτερο ποσοστό εμφάνισης κατά φθίνουσα σειρά για όλους τους σταθμούς (Σχήμα Γ'7).

Γ'.4 Η βάση δεδομένων

Το πρώτο βήμα για τη δημιουργία της εφαρμογής , ήταν ο σχεδιασμός της βάσης δεδομένων, ο οποίος έγινε με βάση τους εξής παράγοντες

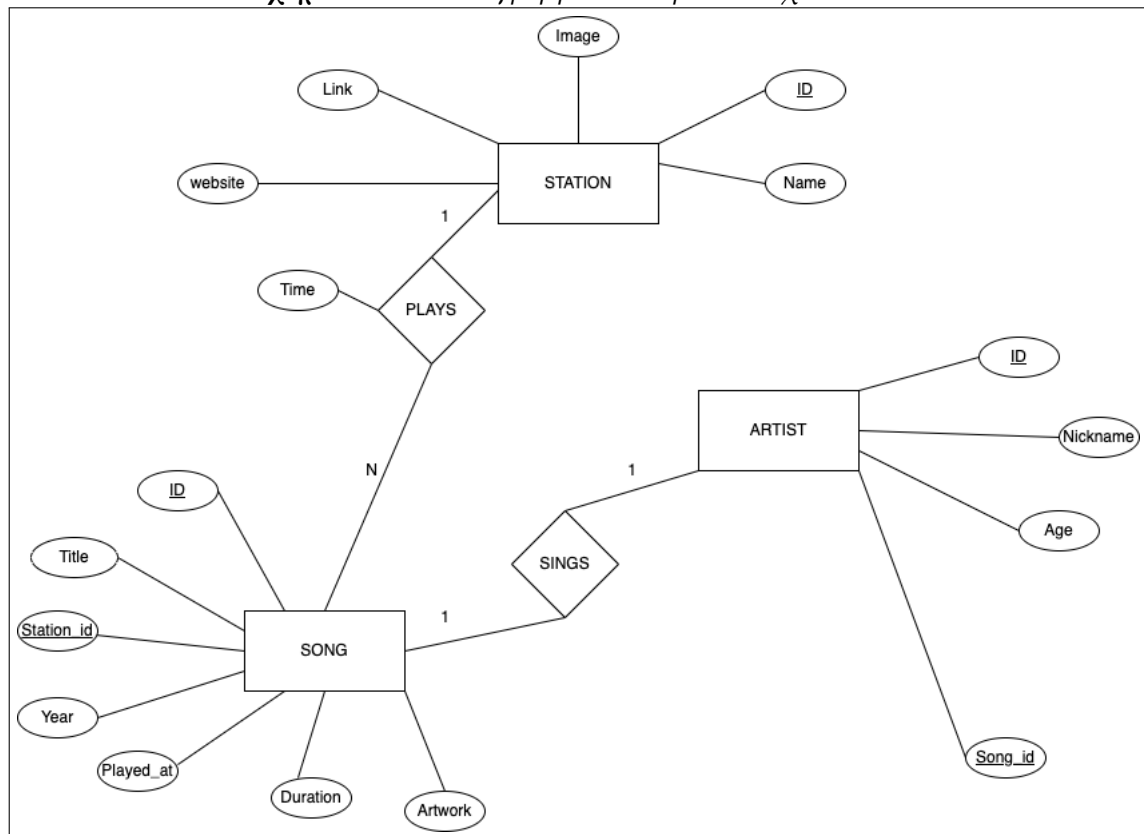
- Το μέγεθος της πληροφορίας : Στη συγκεκριμένη περίπτωση η βάση θα ενημερώνεται συνεχώς , αφού θα αποθηκεύει δεδομένα περίπου ανά ενάμιση λεπτό. Η συγκεκριμένη λειτουργία δεν θα σταματήσει να τρέχει οπότε έπρεπε να γίνει σωστός σχεδιασμός για την καλύτερη αποθήκευση της πληροφορίας και την αποφυγή άσκοπων ερωτημάτων στη βάση.
- Την καλύτερη επεξεργασία των δεδομένων : Τόσο για την ταχύτητα της εφαρμογής αλλά και για την ποιότητα του κώδικα. Οι πίνακες δημιουργήθηκαν με το σκεπτικό αποφυγής πολύπλοκων ερωτημάτων SQL (queries). Δημιουργώντας μια δομή η οποία θα έκανε πιο δομημένο και τον κώδικα της εφαρμογής ακολουθώντας βασικές αρχές των βάσεων δεδομένων
- Βοήθεια στον Προγραμματιστή : Ο πιο σημαντικός παράγοντας. Στο επίπεδο κώδικα ο προγραμματιστής θα πρέπει να νιώθει ότι η βάση δεδομένων που έχει στα χέρια του, καλύπτει όλες τις ανάγκες σε επίπεδο κώδικα, τον καθοδηγεί και μπορεί να προσεγγίσει διαισθητικά τα περιεχόμενα της. Πιο συγκεκριμένα να τον βοηθάει με το τρόπο που έχει σχεδιαστεί στην ανάπτυξη κώδικα αλλά και στην προγραμματιστική λογική.
- Μελλοντική χρήση - Νέα features: Σχεδιάζοντας τη βάση δεδομένων το επόμενο βήμα είναι να πορευτεί ο προγραμματιστής με τις αποφάσεις που έχουν παρθεί στο σχεδιασμό. Έτσι θα πρέπει να γίνει σωστή πρόβλεψη για την κάλυψη όλων των αναγκών της εφαρμογής. Θα πρέπει να υπάρχει η δυνατότητα προσαρμογής σε νέες καταστάσεις χωρίς να επιβαρύνει την προηγούμενη έκδοσή της.

Γ'.4.1 Σχεδιασμός και σχήμα

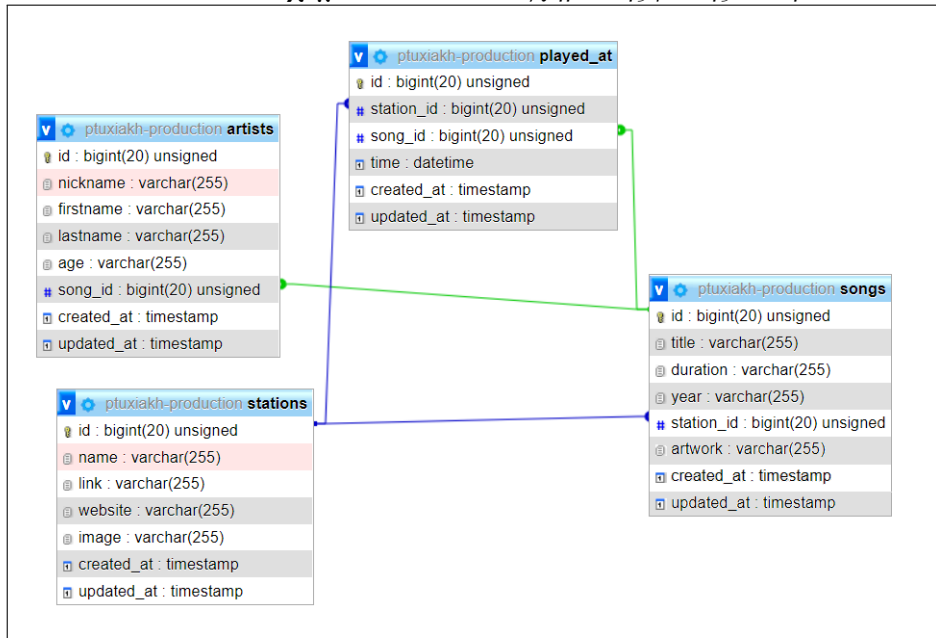
Έχοντας τις απαραίτητες πληροφορίες από τους σταθμούς, δηλαδή τα δεδομένα που παρέχουν οι 5 διαδικτυακοί σταθμοί, ξεκινάει μια διαδικασία ανάλυσης και ομαδοποίησης τους. Οι σταθμοί παρέχουν δεδομένα όπως ο τίτλος τραγουδιού, το όνομα του καλλιτέχνη, η διάρκεια του τραγουδιού, η ώρα που έγινε η μετάδοση του τραγουδιού, την ακριβή ημερομηνία μετάδοσης,

τον τύπο μετάδοσης αν είναι τραγούδι, διαφήμιση, την κατάσταση (status) του σταθμού και το artwork του τραγουδιού. Επίσης πολλοί από τους σταθμούς παρέχουν τα ίδια δεδομένα για τα τρία κομμάτια που μεταδόθηκαν πριν από το τραγούδι που μεταδίδεται τη συγκεκριμένη χρονική στιγμή με χρονολογική διάταξη. Από αυτές τις πληροφορίες έπρεπε να γίνει καθορισμός ποιων πεδίων θα ήταν χρήσιμα για το σύστημα. Ποια πεδία δηλαδή έχουν νόημα για την αποθήκευσή τους στη βάση και είναι απαραίτητα για τη λειτουργία της εφαρμογής. Μέσα από αυτές τις πληροφορίες δημιουργήθηκαν 3 βασικοί πίνακες. Ο πίνακας των τραγουδιών (στην Αγγλική κι εφεξής στο κείμενο για συντομία, songs) ο πίνακας των καλλιτεχνών (στην Αγγλική κι εφεξής στο κείμενο για συντομία, artists) και ο πίνακας των σταθμών (στην Αγγλική κι εφεξής στο κείμενο για συντομία, stations).

Σχήμα Γ'.8: Διάγραμμα οντοτήτων συσχετίσεων .



Σχήμα Γ'.9: Το σχήμα της βάσης δεδομένων .



Γ'.4.2 Πίνακες

Ο πίνακας SONGS

Ο πίνακας **songs** είναι υπεύθυνος για την αποθήκευση όλων των δεδομένων που σχετίζονται με τα μουσικά κομμάτια.

- **id - bigint(20)**: Το πρωτεύον κλείδι του πίνακα. Είναι ο μοναδικός κωδικός εγγραφής για κάθε στοιχείο του πίνακα
- **title - varchar(255)**: Ο τίτλος του τραγουδιού με τη μορφή string.
- **duration - varchar(255)**: Είναι ο χρόνος διαρκείας του τραγουδιού το αποθηκεύουμε με τη μορφή string για την καλύτερη επεξεργασία.
- **year - varchar(255)** : Η χρονιά που δημιουργήθηκε το τραγούδι.
- **station_id - bigint(20)**: Ο μοναδικός κωδικός για τον κάθε σταθμό. Αντιπροσωπεύει σε ποιόν σταθμό παίχτηκε το τραγούδι. Είναι ξένο κλειδί στον πίνακα **songs** και πρωτεύον στον πίνακα **stations**.

- played_at- varchar(255): Είναι η ημερομηνία μετάδοσης του τραγουδιού σε μορφή Ημέρα - Μήνας-Χρονιά .
- artwork - varchar(255): Είναι ο σύνδεσμος που υπάρχει το artwork του τραγουδιού .
- created_at- timestamp: Ημερομηνία εγγραφής στη βάση.
- updated_at - timestamp: Ημερομηνία επεξεργασίας στη βάση .

Ο πίνακας ARTISTS

Ο πίνακας artists είναι υπεύθυνος για την αποθήκευση όλων των πληροφοριών που σχετίζονται με τους καλλιτέχνες.

- id - bigint(20): Το πρωτεύον κλειδί του πίνακα. Είναι ο μοναδικός κωδικός εγγραφής για κάθε στοιχείο του πίνακα
- nickname - varchar(255): Η βασική ονομασία του καλλιτέχνη με τη μορφή string.
- firstanme - varchar(255): Το μικρό όνομα του καλλιτέχνη
- lastname - varchar(255): Το επίθετο του καλλιτέχνη.
- age - varchar(255): Η ηλικία του καλλιτέχνη.
- song_id - bigint(20): ο μοναδικός κωδικός για το κάθε τραγούδι. Κάθε καλλιτέχνης συνδέεται με πολλά τραγούδια. Είναι ξένο κλειδί στον πίνακα artists και πρωτεύον στον πίνακα songs.
- created_at- timestamp: Ημερομηνία εγγραφής στη βάση.
- updated_at - timestamp: Ημερομηνία επεξεργασίας στη βάση.

Ο πίνακας STATIONS Ο πίνακας stations είναι υπεύθυνος για την αποθήκευση όλων των πληροφοριών που σχετίζονται με τους σταθμούς ,

- id - bigint(20): Το πρωτεύον κλειδί του πίνακα. Είναι ο μοναδικός κωδικός εγγραφής για κάθε στοιχείο του πίνακα
- name - varchar(255): Το όνομα του σταθμού.

- `link - varchar(255)`: Είναι ο σύνδεσμος όπου γίνεται η εξαγωγή της πληροφορίας για κάθε τραγούδι που μεταδίδει.
- `website - varchar(255)`: Η σελίδα του σταθμού σε μορφή string.
- `image - varchar(255)`: Το λογότυπο του σταθμού. Αποθηκεύουμε το σύνδεσμο που βρίσκεται η εικόνα σε μορφή string.
- `created_at - timestamp`: Ημερομηνία εγγραφής στη βάση.
- `updated_at - timestamp`: . Ημερομηνία επεξεργασίας στη βάση.

Η δημιουργία των πινάκων στο Laravel γίνεται μέσω μια λειτουργίας που λέγεται migrations. Μέσα από τα migration στα ελληνικά 'μετεγκατάσταση' το Laravel μπορεί να δημιουργήσει έναν νέο πίνακα ή να αναβαθμίσει τα πεδία κάποιου ήδη υπάρχοντος. Για κάθε migration δημιουργείτε ένα αρχείο στο `database/migrations` και κρατάει χρονολογική σειρά για το καθένα για τον καλύτερο προσδιορισμό της σειράς που πρέπει να εκτελεστούν (Database: Migrations, 2022).

Η δημιουργία σχέσεων (relationships) μεταξύ των πινάκων έγινε με τη βοήθεια του eloquent (Εργαλείο του Laravel) το οποίο δημιουργεί μέσω κώδικα τις σχέσεις μεταξύ των πινάκων αφού όμως έχει γίνει πρώτα σωστά η διαδικασία δήλωσης των κλειδιών στα migrations. Κάθε πίνακας της βάσης στο Laravel αναπαρίσταται και ως μοντέλο. Έτσι σε κάθε μοντέλο αντίστοιχα δηλώνουμε τις σχέσεις με τα υπόλοιπα μοντέλα για την καλύτερη διαχείριση των δεδομένων αλλά και για την εύκολη πρόσβαση στους πίνακες σε επίπεδο κώδικα.

Γ'.5 Τεχνικά χαρακτηριστικά

Το Laravel δίνει στον προγραμματιστή ένα τεράστιο πακέτο με εργαλεία και βιβλιοθήκες που με τη σωστή χρήση του μπορεί να δημιουργηθούν εφαρμογές με σωστή δομή. Μία από τις βασικές τεχνικές αναπαράστασης ενός αντικειμένου της βάσης είναι με τη μορφή μοντέλου. Έτσι για κάθε πίνακα στη βάση έχει δημιουργηθεί ένα αντίστοιχο μοντέλο το οποίο έχει τα πεδία του πίνακα ως χαρακτηριστικά (properties). Συγκεκριμένα υπάρχουν τρία μοντέλα για την παρούσα εργασία. Το μοντέλο Song, το μοντέλο Artist και το Μοντέλο Station. Το μοντέλο αποτελεί μια κλάση μέσα στην οποία ο προγραμματιστής μπορεί να δημιουργήσει τις

δικές του μεθόδους - συναρτήσεις (functions). Όλα αυτά βεβαία γίνονται με τη χρήση του Eloquent το οποίο εξειδικεύεται για την τεχνική της αντικειμενικής-σχεσιακής χαρτογράφησης στα αγγλικά (Object-Relational Mapping). Με το συγκεκριμένο πακέτο επιτυγχάνονται τα περισσότερα ερωτήματα στη βάση, τα οποία γράφονται με συγκεκριμένη μορφή μέσα από κώδικα php (Eloquent ORM, 2022).

Η δημιουργία σχέσεων (relationships) μεταξύ των πινάκων έγινε με τη βοήθεια του eloquent (Εργαλείο του Laravel) το οποίο δημιουργεί μέσω κώδικα τις σχέσεις μεταξύ των πινάκων αφού όμως έχει γίνει πρώτα σωστά η διαδικασία δήλωσης των κλειδιών στα migrations. Όπως προαναφέρθηκε ένας πίνακας της βάσης στο Laravel αναπαρίσταται και ως μοντέλο. Έτσι σε κάθε μοντέλο αντίστοιχα δηλώνουμε τις σχέσεις με τα υπόλοιπα μοντέλα για την καλύτερη διαχείριση των δεδομένων άλλα και για την εύκολη πρόσβαση στους πίνακες σε επίπεδο κώδικα (Eloquent: Relationships, 2022). Πρωταρχικό ρόλο στο συγκεκριμένο framework παίζει και η λειτουργία των controller αφού διαχειρίζονται τα δεδομένα και έρχονται σε επαφή με τα μοντέλα αντλώντας τις απαραίτητες πληροφορίες που θέλει να δει ο χρήστης στην οθόνη του. Ένας controller συνήθως είναι γραμμένος πάνω σε κάποιο action της εφαρμογής το οποίο δηλώνεται ως route σε συγκεκριμένο σημείο μέσα στο Laravel μαζί με το σύνδεσμο (url) της σελίδας και την μέθοδο (http request) που υλοποιεί (Controllers, 2022). Τα html αρχεία δημιουργούνται μέσα από το πρότυπο blade το οποίο βοηθάει στην καλύτερη χρήση του Html χωρίζοντας το σε κομμάτια δημιουργώντας components. Στην εφαρμογή τα blade αρχεία βρίσκονται στο φάκελο resources/views/ και έχουν την κατάληξη .blade.php. Στον συγκεκριμένο φάκελο υπάρχει ακόμα ένας φάκελος με όνομα Layouts στον οποίο βρίσκεται app.blade.php όπου ορίζεται η βασική δομή των σελίδων και δηλώνονται τα απαραίτητα css αρχεία, οι σύνδεσμοι των css αρχείων, κάποια script που χρειάζονται για όλες τις σελίδες της εφαρμογής το μενού και το footer. Κάθε blade (σελίδα) επεκτείνει ουσιαστικά το layout και κληρονομεί τον σκελετό της σελίδας. Στη συγκεκριμένη εφαρμογή το layout χωρίζει της σελίδα σε τέσσερα βασικά μέρη ('stylesheets, header, content , footer καί javascripts) (Blades, 2022).

Τα αρχικά απαραίτητα δεδομένα που χρειάζονται στην εφαρμογή έχουν εισαχθεί με τη δημιουργία αρχείου εισαγωγής δεδομένων (Στην Αγγλική και εφεξής στο κείμενο για συντομία, seeder) . Για την εισαγωγή των σταθμών στο σύστημα γράφτηκε ένας seeder ο StationSeeder ο οποίος καλείται από τον γενικό seeder του συστήματος τον DatabaseSeeder (Database

Seeding, 2022).

Γ'.5.1 Ανάλυση τεχνικών ανάπτυξης κώδικα

Στην παρούσα εργασία χρησιμοποιήθηκαν αρκετές τεχνικές ανάπτυξης κώδικα. Στο παρόν κεφάλαιο γίνεται ανάλυση τεχνικών για τις βασικές λειτουργίες του Laravel με παραδείγματα κώδικα αλλά και επεξήγηση διαφόρων design patterns που χρησιμοποιήθηκαν.

Παράδειγμα Πρώτο: Για τις ανάγκες ανανέωσης δεδομένων μιας σελίδας ασύγχρονα, δηλαδή χωρίς να φορτωθεί εκ νέου ολόκληρη η σελίδα, χρησιμοποιήθηκε το Livewire το οποίο είναι μία βιβλιοθήκη που διαθέτει το Laravel για δυναμικές εφαρμογές χωρίς να χρησιμοποιηθεί κάποιο Full-stack javascript framework. Το Livewire δημιουργεί AJAX requests στο server με τα ενημερωμένα δεδομένα φορτώνοντας ξανά το HTML της σελίδας. Το συγκεκριμένο εργαλείο χρησιμοποιήθηκε για να γίνει πιο εύκολη η αναζήτηση των τραγουδιών στη φάση της δημιουργίας playlist. Ο χρήστης ξεκινάει να πληκτρολογεί το τραγούδι που θέλει για να δημιουργήσει μία playlist. Όσο πληκτρολογεί, αυτόματα ανανεώνεται και η λίστα των τραγουδιών ανάλογα με το τι έχει πληκτρολογήσει στην αναζήτηση. Επίσης το μέγεθος της playlist ενημερώνεται και αυτό ασύγχρονα και σε περίπτωση που ο χρήστης προσπαθήσει να ενημερώσει τη μεταβλητή του μήκους της λίστας με τιμή που δεν επιτρέπει το σύστημα, τότε η τιμή γίνεται αυτόματα πέντε. Αφού γίνει η εγκατάσταση του πακέτου μέσω του composer με την εντολή `php artisan make:livewire SearchSong`, δημιουργούνται δύο αρχεία, ένα το `SearchSong.php` το οποίο είναι μία PHP κλάση με path `App`, και ένα τύπου blade αρχείο το `search-song.blade` το οποίο είναι το view όπου βρίσκεται στον φάκελο `views/livewire`. Στην κλάση `SearchSong.php` υπάρχουν δύο βασικές συναρτήσεις η `mount` και η `render`, επίσης υπάρχει και μία μέθοδος η `updatingSearch` που χρησιμοποιείται για να λειτουργεί σωστά η σελιδοποίηση των τραγουδιών. Η `mount` χρησιμοποιείται για την αρχικοποίηση των μεταβλητών με μία συγκεκριμένη τιμή. Η `render` χρησιμοποιείται για την αρχική φόρτωση της σελίδας και για κάθε επόμενη ενημέρωση των δεδομένων ή των στοιχείων html (Livewire, 2022).

Listing C.4: *SearchSong.php*

```
namespace App\Http\Livewire;  
  
use Livewire\Component;  
use Livewire\WithPagination;
```

```

use App\Models\Song;
class SearchSong extends Component
{
    use WithPagination;
    public $search;
    public $length ;

    public function mount()
    {
        $this->length = 5;
    }
    public function updatingSearch(){
        $this->resetPage();
    }

    public function render()
    {
        $length = $this->length;
        $search = '%' . $this->search . '%';

        if ($this->length == null or
            $this->length == '' or
            !isset($this->length))
        {
            $this->length =5;
        }

        return view('livewire.search-song',[
            'songs' => Song::query()
                ->where('title','like',$search)
                ->paginate(10),
            'length' => $length
        ]);
    }
}

```

Όσον αφορά το view δηλαδή το Blade αρχείο, υπάρχουν δύο δυναμικές μεταβλητές το search

που αντιπροσωπεύει το σύνολο των χαρακτήρων που πληκτρολογεί ο χρήστης και το length το οποίο αντιπροσωπεύει το μέγεθος της playlist. Οι δύο μεταβλητές κάθε φορά που χρήστης αλλάζει τιμές σε μία από αυτές το html ενημερώνεται ταυτόχρονα.

Listing C.5: *search-song.blade.php*

```
<div class="input" style="display: flex">
  <input type="text" class="form-control" placeholder="Search"
  wire:model="search" />
</div>
<div class="mt-3">
  <label for="number">Please select the playlist length :</label>
  <input id="number" type="number" min="1" max="25" wire:model="length">
</div>
```

Παράδειγμα Δεύτερο:

Για την καλύτερη διαχείριση των δεδομένων και για τις απαραίτητες μεθόδους που σχετίζονται με το Μοντέλο Song δημιουργήθηκε πρότυπο αποθετηρίου (Repository). Το repository design pattern είναι ένα από τα πιο γνωστά που χρησιμοποιούν οι προγραμματιστές στο Laravel. Η δημιουργία του SongRepository σχεδιάστηκε με τη λογική ότι όλα τα βασικά actions που υλοποιεί το μοντέλο που σχετίζονται με την βάση, όπως η εγγραφή δεδομένων αλλά και τα διαφορετικά ερωτήματα αντί να βρίσκονται στη λειτουργία του controller, βρίσκονται πλέον στο repository. Ο StationsController είναι υπεύθυνος για την αποθήκευση των τραγουδιών στη βάση δεδομένων. Ουσιαστικά είναι ο controller που καλεί το cron job για την αυτόματη αποθήκευση των τραγουδιών για τον εκάστοτε σταθμό. Μέσα στον controller με τη χρήση μιας construct μεθόδου, γίνεται η αρχικοποίηση του repository και στην συνέχεια ανάλογα με τον τύπο του σταθμού (τύπος σταθμού μόνο για το σύστημα) καλείται η αντίστοιχη μέθοδος του repository (How to Use the Repository Pattern in a Laravel Application, 2022).

Listing C.6: *store type selection*

```
if ($station->id == 1 || $station->id == 2 || $station->id == 3){
    $data = $this->songRepository->storeTypeOne($station);
} elseif ($station->id == 4 || $station->id == 5){
    $data = $this->songRepository->storeTypeTwo($station);
}
```

Κεφάλαιο Δ΄

ΑΥΤΟΜΑΤΗ ΔΗΜΙΟΥΡΓΙΑ **PLAYLIST**

Δ΄.1 Αποφάσεις υλοποίησης αλγόριθμου

Για τη δημιουργία της αυτόματης δημιουργίας λίστας μουσικών τραγουδιών γράφτηκε αλγόριθμος ο οποίος είχε ως στόχο να παρέχει μουσική στους χρήστες με βάση ένα μουσικό κομμάτι της προτίμησης τους. Ο αλγόριθμος σχετίζεται άμεσα με τους ραδιοφωνικούς παραγωγούς που κάνουν την επιλογή μουσικών κομματιών σε κάθε σταθμό. Για την ανάπτυξη του αλγορίθμου έπρεπε να παρθούν αρκετές αποφάσεις και να δοθούν τα κατάλληλα όρια για την καλύτερη προσέγγιση των αποτελεσμάτων. Οι αποφάσεις που πάρθηκαν είναι οι εξής :

- Η βασική ιδέα: ο αλγόριθμος βασίζεται στις επιλογές τραγουδιών από τους παράγωγους των σταθμών, αφού μελετάμε το ποσοστό ακολουθίας ενός τραγουδιού από το τραγούδι που προηγείται. Πιο συγκεκριμένα οι μουσικές λίστες δημιουργούνται ανάλογα με το ποσοστό εμφάνισης ενός τραγουδιού όταν ακολουθείται από ένα συγκεκριμένο κομμάτι. Όσο μεγαλύτερο είναι το ποσοστό εμφάνισης του συγκεκριμένου τραγουδιού τόσο μεγαλύτερη είναι και η πιθανότητα εισαγωγής του στην playlist. Η ίδια διαδικασία επαναλαμβάνεται για κάθε τραγούδι που μπαίνει στην playlist δημιουργώντας έτσι μία μουσική λίστα.
- Αποφυγή σφαλμάτων: στη συγκεκριμένη διαδικασία εμφανίστηκαν αρκετές δυσκολίες και περιπτώσεις που μπορεί να οδηγήσουν σε σφάλμα. Το βασικό σενάριο που έπρεπε να αποφευχθεί ήταν η προσθήκη τραγουδιού που ήδη υπάρχει στην playlist, δηλαδή με ποιόν τρόπο ο αλγόριθμος θα έπρεπε να αντιμετωπίσει μία τέτοια περίπτωση που υπήρχε

μεγάλη πιθανότητα να συμβεί και να οδηγήσει σε ατέρμονα βρόγχο. Στη συγκεκριμένη περίπτωση ο αλγόριθμος κάνει έλεγχο για το αμέσως επόμενο τραγούδι με τη μεγαλύτερη εμφάνιση, αν υπάρχει και το συγκεκριμένο στην playlist συνεχίζει στο επόμενο μέχρι να βρει κάποιο που δεν εμπεριέχεται, αν δεν βρει κανένα η διαδικασία τερματίζεται και επιστρέφει την playlist που έχει φτιαχτεί στη συγκεκριμένη χρονική στιγμή.

- Επιλογή επόμενου τραγουδιού όταν παραπάνω από ένα τραγούδια έχουν ίσο ή μεγαλύτερο ποσοστό εμφάνισης ή όλα τα επόμενα τραγούδια εμφανίζονται μία φορά. Στη συγκεκριμένη περίπτωση ο αλγόριθμος επιλέγει το πρώτο τραγούδι που βρίσκεται στη λίστα σειριακά ακόμα και αν ο αριθμός εμφανίσεων είναι ισόβαθμος. Η συγκεκριμένη λογική υιοθετήθηκε για να απλοποιηθεί τη διαδικασία, για να μην επιβαρυνθεί ο αλγόριθμος με περισσότερες χρονοβόρες τεχνικές επεξεργασίας δεδομένων και να αυξηθεί αρκετά ο χρόνος εκτέλεσης του.
- Σημαντική ήταν και η απόφαση για το μέγεθος της λίστας, δηλαδή πότε η συγκεκριμένη διαδικασία θα τερματίζεται. Όσο μεγαλύτερη είναι η λίστα τραγουδιών φαίνεται ότι σιγά σιγά χάνεται και η συνοχή της playlist μετά από μελέτη των αποτελεσμάτων. Έτσι έπρεπε να βρεθεί το κατάλληλο εύρος. Μετά από πειραματισμό με τον αλγόριθμο και μελέτη των αποτελεσμάτων, πάρθηκε η απόφαση η εκάστοτε playlist να περιέχει από 1 έως 25 τραγούδια. Επίσης στον χρήστη δίνεται η δυνατότητα να επιλέξει ο ίδιος το μέγεθος της λίστας από το συγκεκριμένο εύρος τιμών.
- Επέκταση αλγορίθμου: ο αλγόριθμος σχεδιάστηκε με το σκεπτικό μελλοντικής επέκτασης. Πιο συγκεκριμένα πόσο εύκολα θα μπορέσει να προσαρμοστεί σε νέες παραμέτρους προσφέροντας καλύτερες προσεγγίσεις. Μια από τις σημαντικότερες αποφάσεις που έπρεπε να παρθούν, αφού επηρεάζει σε καταληκτικό βαθμό την ανάπτυξη κώδικα και σχεδίαση του.

Δ'.2 Ανάπτυξη αλγορίθμου

Για την καλύτερη ανάπτυξη του αλγορίθμου χρειάστηκε να γίνει η ανάλυσή του σε βασικά βήματα και να μελετηθεί κάθε μέρος του ξεχωριστά.

- Βήμα πρώτο : Η επιλογή τραγουδιού εισόδου. Η συγκεκριμένη επιλογή γίνεται από τον χρήστη αφού επιλέγει ανάμεσα από ένα πλήθος τραγουδιών δημιουργώντας τη βάση του

αλγορίθμου που θα εκτελεστεί η συγκεκριμένη διαδικασία.

- Βήμα δεύτερο : Η μέτρηση των εμφανίσεων για το τραγούδι εισόδου που έχει δώσει ο χρήστης.
- Βήμα τρίτο : Για κάθε εμφάνιση του τραγουδιού γίνεται αναζήτηση του αμέσως επόμενου τραγουδιού στον συγκεκριμένο σταθμό που αναμεταδόθηκε. Για παράδειγμα αν το τραγούδι εισόδου έχει αναμεταδοθεί πέντε φορές, για κάθε μία από αυτές βρίσκουμε τον σταθμό και το επόμενο τραγούδι που παίχτηκε στον συγκεκριμένο σταθμό.
- Βήμα τέταρτο : Δημιουργία πίνακα με τις βασικές επόμενες εμφανίσεις. Στον συγκεκριμένο πίνακα έχουμε όλα τα τραγούδια που έχουν μεταδοθεί αμέσως μετά από το τραγούδι εισόδου.
- Βήμα Πέμπτο : Αναδιαμόρφωση πίνακα επόμενων τραγουδιών. Αν στο συγκεκριμένο πίνακα κάποια τραγούδια υπάρχουν παραπάνω από μία φορά, γίνεται ταξινόμηση του πίνακα με το ποσοστό εμφανίσεων για το κάθε τραγούδι. Για παράδειγμα αν μετά το τραγούδι εισόδου με αριθμό εμφανίσεων πέντε, έχουμε τα εξής επόμενα τραγούδια, το τραγούδι α' που έχει εμφανιστεί 3 φορές μετά από το τραγούδι εισόδου με πέντε συνολικές εμφανίσεις και το τραγούδι β' και γ' που έχουν εμφανιστεί μία, δημιουργείται ο πίνακας επιλογής με πρώτο τραγούδι το τραγούδι α' με τρεις από τις πέντε εμφανίσεις και το τραγούδι β' και γ' με μία στις πέντε εμφανίσεις.
- Βήμα Έκτο : Επιλογή επόμενου τραγουδιού από τον πίνακα εμφανίσεων. Αφού έχει γίνει ταξινόμηση των επόμενων τραγουδιών από το τραγούδι εισόδου ανάλογα με τον αριθμό εμφανίσεων, ο αλγόριθμος επιλέγει το πρώτο τραγούδι του πίνακα (τραγούδι με το μεγαλύτερο ποσοστό εμφανίσεων) για να το εισάγει στην playlist.
- Βήμα Έβδομο : Έλεγχος πριν την εισαγωγή, για κάθε τραγούδι πριν γίνει η εισαγωγή του στην Playlist, γίνεται ο εξής έλεγχος στην περίπτωση που είναι το πρώτο τραγούδι στην playlist, δηλαδή το τραγούδι που επιλέγεται μετά το τραγούδι εισόδου εισάγεται κατευθείαν στη λίστα. Αν δεν είναι το πρώτο τραγούδι γίνεται έλεγχος αν το τραγούδι υπάρχει ήδη στη λίστα. Αν υπάρχει ο αλγόριθμος επιλέγει το αμέσως επόμενο τραγούδι του πίνακα με το αμέσως μεγαλύτερο ποσοστό εμφανίσεων και εκτελεί την ίδια διαδικασία για κάθε τραγούδι που υπάρχει στον πίνακα εμφανίσεων μέχρι να μπορέσει να εισάγει κάποιο. Αν υπάρχουν όλα τα τραγούδια ήδη στη λίστα η διαδικασία τερματίζεται και επιστρέφει την playlist που έχει δημιουργηθεί τη συγκεκριμένη χρονική στιγμή.

- Βήμα Όγδοο : Εκτέλεση της ίδιας διαδικασίας για το τραγούδι επιλογής. Η ίδια διαδικασία γίνεται όσες φορές επιλέξει ο χρήστης δίνοντας τον αριθμό τραγουδιών που θέλει να έχει η playlist που θα του δημιουργήσει το σύστημα.

Όσον αφορά το επίπεδο κώδικα δημιουργήθηκαν αρκετές συναρτήσεις για το χωρισμό του αλγορίθμου σε βήματα αλλά και για την καλύτερη διαχείριση των εργασιών του αλγορίθμου. Για την ανάπτυξη του αλγορίθμου χρησιμοποιήθηκαν πέντε συναρτήσεις. Η μεθοδολογία σχεδιάστηκε με βάση τις αναδρομικές συναρτήσεις. Η βασική συνάρτηση SongSelection καλεί άλλη μία συνάρτηση την checkPlaylist η οποία καλεί άλλες τρεις συναρτήσεις και στο τέλος καλεί πάλι την αρχική συνάρτηση SongPlaylist μέχρι να σταματήσει μια συγκεκριμένη συνθήκη (Recursion, 2022). Η βασική συνάρτηση είναι η SongSelection η οποία δέχεται τρία ορίσματα τον τίτλο του τραγουδιού, το μέγεθος της playlist και την playlist (\$title, \$length, \$playlist). Η SongSelection είναι η υπεύθυνη για την εύρεση του επομένου τραγουδιού. Η μεταβλητή title που δέχεται ο αλγόριθμος αντιπροσωπεύει τον τίτλο του τραγουδιού εισόδου από τον χρήστη. Ο αλγόριθμος βρίσκει όλες τις εμφανίσεις για το τραγούδι εισόδου και για κάθε μία από αυτές συλλέγει το επόμενο τραγούδι. Δημιουργεί έναν πίνακα με τα επόμενα τραγούδια και τους καλλιτέχνες αντίστοιχα. Στο συγκεκριμένο πίνακα αν υπάρχει κάποιο στοιχείο περισσότερες φορές από μία δημιουργεί ένα καινούργιο πίνακα που ανά-ταξινομεί τα στοιχεία εμφανίζοντάς τα μοναδικά μαζί με το πόσες φορές εμφανίζονται. Στη συνέχεια καλείται η συνάρτηση checkPlaylist η οποία δέχεται σαν παραμέτρους τον ταξινομημένο πίνακα με τα επόμενα τραγούδια που δημιουργεί η προηγούμενη συνάρτηση, τον πίνακα playlist που μέχρι στιγμής είναι άδειος και το μέγεθος της playlist. Το πρώτο βήμα στη συγκεκριμένη συνάρτηση είναι να ελέγχει το μέγεθος της playlist αν είναι μικρότερο ή ίσο με το μέγεθος που έχει δώσει ο χρήστης και όταν η συγκεκριμένη συνθήκη σταματήσει να ισχύει επιστρέφει τον πίνακα της playlist. Στη συνέχεια επεξεργάζεται τη μορφή του πίνακα με τα επόμενα τραγούδια, και ελέγχει με μία βοηθητική συνάρτηση που καλείται την multiArrayKeyExists() , σειριακά αν υπάρχει το συγκεκριμένο στον πίνακα playlist. Αν δεν υπάρχει το επιστρέφει, αλλιώς συνεχίζει στο επόμενο σειριακά τραγούδι (Ταξινομημένο με τον αριθμό εμφανίσεων) και κάνει τον ίδιο έλεγχο μέχρι να βρει ποιο τραγούδι μπορεί να προστεθεί. Το τραγούδι που επιστρέφεται μαζί με τον πίνακα playlist είναι το όρισμα της επόμενης συνάρτησης που καλείται, της addToPlaylist η οποία προσθέτει στον πίνακα playlist το τραγούδι. Τέλος ξανα-καλείτε η songSelection για την επανάληψη της διαδικασίας με βάση τον αριθμό επαναλήψεων που δηλώνει ο χρήστης με το μέγεθος της playlist.

```

Data: title, maxLength, playlist
Result: playlist (Array)
if NOT FILLED(playlist) then
  | playlist = [ ]
end
songs  $\Leftarrow$  get all songs with the same title (title) //DatabaseQuery
foreach songs as key  $\Rightarrow$  song do
  | nextSongs[key]  $\Leftarrow$  get next song (song);
end
results = Call array_count_values(parameters: nextSongs);
sort result array in ascending order //Sort by count
if COUNT (playlist)  $\geq$  maxLength then
  | return playlist;
end
foreach results as result do
  | if result EXISTS IN playlist then
    | CONTINUE;
  end
  Call addSongToPlaylist(parameters: result,playlist);
  Call SELF(parameters: result, maxLenght,playlist)
  BREAK;
end

```

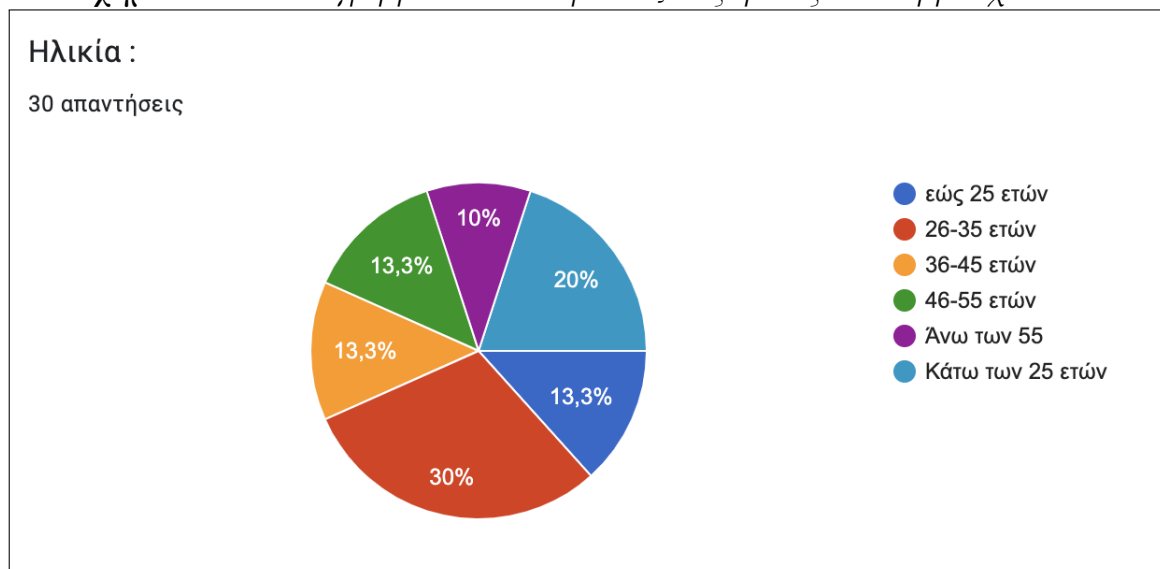
Algorithm 1: Τα βασικά βήματα του αλγορίθμου σε ψευδογλώσσα. Ο αλγόριθμος καλείται αναδρομικά.

Δ'.3 Πειραματισμός Αλγορίθμου

Για την αξιολόγηση των αποτελεσμάτων του αλγορίθμου, δημιουργήθηκε ερωτηματολόγιο με σκοπό να καταλήξουμε σε κάποια συμπεράσματα μέσα από τη βοήθεια των χρηστών της εφαρμογής. Στα ερωτηματολόγια έλαβαν μέρος 30 χρήστες της εφαρμογής, διαφόρων ηλικιών (Σχήμα Δ 9).

Δημιουργήθηκαν τρεις λίστες τραγουδιών με πέντε τραγούδια στην κάθε μία. Η πρώτη λίστα

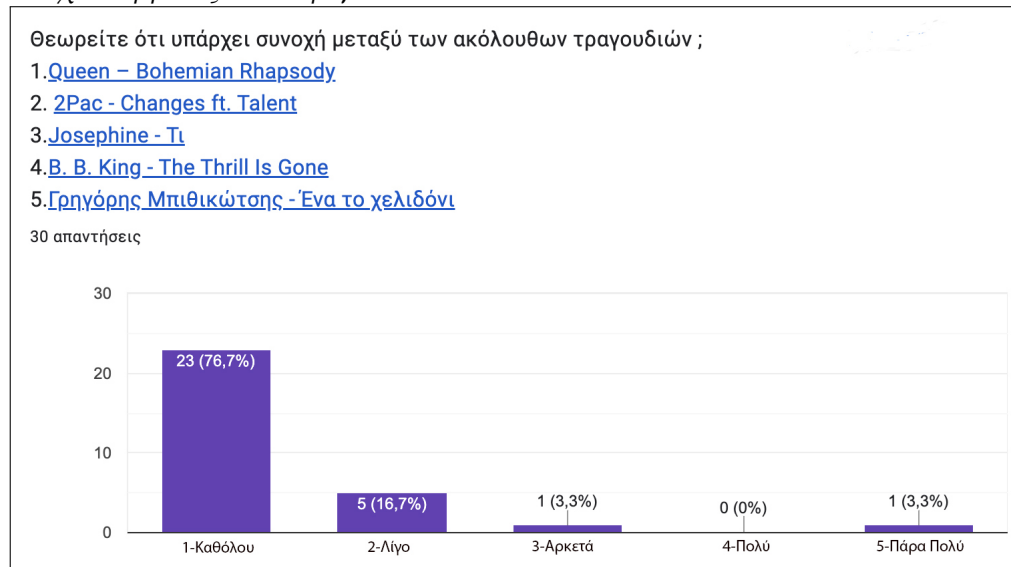
Σχήμα Δ'.10: Διάγραμμα το οποίο παρουσιάζει τις ηλικίες των συμμετεχόντων.



απαρτίζεται από τραγούδια που δεν έχουν καμία σχέση μεταξύ τους, η δεύτερη λίστα περιέχει τραγούδια που σχετίζονται κατά ένα πολύ μεγάλο ποσοστό και η τρίτη είναι μια τυχαία λίστα που δημιουργήθηκε μέσα από την εφαρμογή, με τον αλγόριθμο που αναλύθηκε στο παρών κεφάλαιο. Μέσω του ερωτηματολογίου δόθηκαν στους χρήστες, αυτές οι τρεις λίστες με σκοπό να τις αξιολογήσουν με βάση τη σχετικότητα των τραγουδιών και το ποσοστό συνοχής που θεωρούν ότι έχουν. Με τον συγκεκριμένο τρόπο γίνεται αξιολόγηση του αλγορίθμου συγκριτικά με τις άλλες δύο λίστες. Για κάθε λίστα ρωτήσαμε τους χρήστες αν θεωρούν ότι υπάρχει συνοχή μεταξύ των τραγουδιών κάθε λίστας, με εύρος απαντήσεων από ένα (καθόλου) έως πέντε (πάρα πολύ). Στην πρώτη λίστα (Σχήμα Δ'10) που απαρτίζεται από τραγούδια που δεν σχετίζονται μεταξύ τους το μεγαλύτερο ποσοστό απάντησε ένα (καθόλου) στην κλίμακα με άριστα το πέντε. Επίσης υπήρχε ένα σημαντικό μικρό ποσοστό που απάντησε δύο στα πέντε. Εύλογα, συμπεραίνουμε πως οι χρήστες απάντησαν το προβλεπόμενο για την πρώτη

λίστα καθώς από τις απαντήσεις τους δηλώνουν την μικρή έως καθόλου σχετικότητα της.

Σχήμα Δ'.11: Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις για τη λίστα με μικρή συσχέτιση μεταξύ των τραγουδιών.



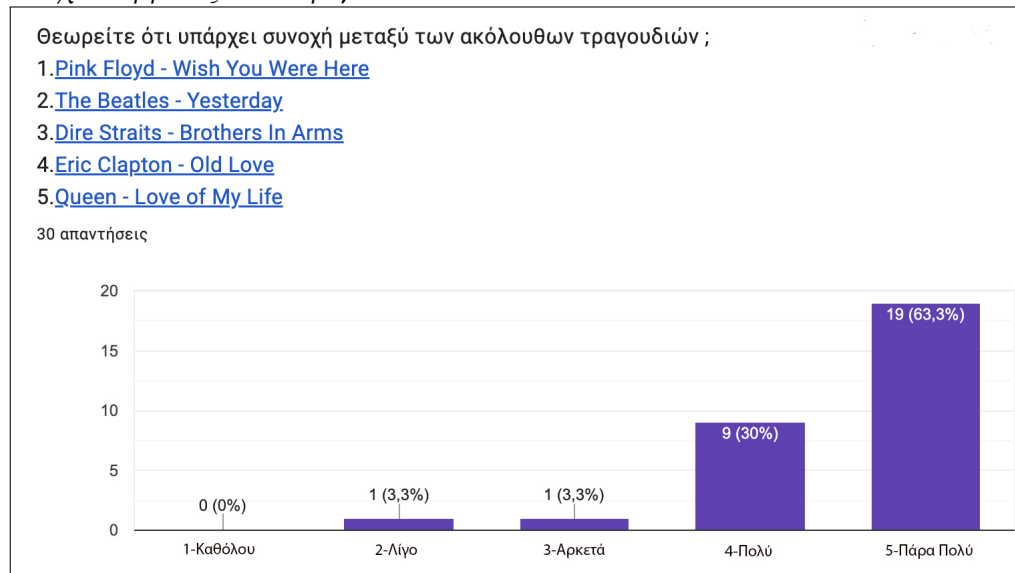
Στη λίστα με τη μεγάλη συσχέτιση των τραγουδιών, οι χρήστες κατά ένα μεγάλο ποσοστό συμφώνησαν αφού το μεγαλύτερο ποσοστό απάντησε με πέντε (πάρα πολύ) και τέσσερα (πολύ) (Σχήμα Δ'11). Οι απαντήσεις τους επιβεβαιώνουν ότι η συγκεκριμένη λίστα έχει όντως ένα μεγάλο ποσοστό σχετικότητας ανάμεσα στα τραγούδια.

Στην επόμενη λίστα που δημιουργήθηκε μέσα από την εφαρμογή, οι περισσότεροι χρήστες θεώρησαν ότι υπάρχει συνοχή μεταξύ των τραγουδιών της λίστας κατά ένα καλό ποσοστό αφού οι απαντήσεις κυμαίνονται από τρία μέχρι πέντε στα πέντε (Σχήμα Δ'12).

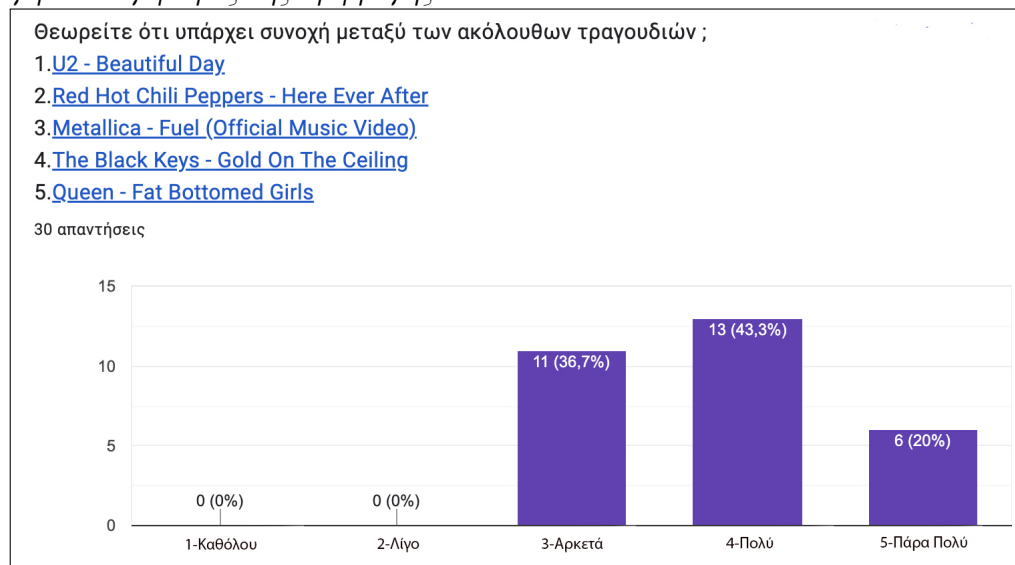
Τέλος οι χρήστες ερωτήθηκαν για τη συνοχή των playlist που δημιούργησαν με βάση την εμπειρία τους με τη συγκεκριμένη εφαρμογή (Σχήμα Δ'13).

Μέσα από τα ερωτηματολόγια μπορούν να εξαχθούν πολλά συμπεράσματα τόσο για τον αλγόριθμο που προτείνεται στην εργασία όσο και για την εφαρμογή. Οι χρήστες αξιολόγησαν θετικά κατά ένα σημαντικό ποσοστό τον αλγόριθμο και την λειτουργία της εφαρμογής. Θεώρησαν ότι οι μουσικές λίστες που δημιούργησαν μέσα από την εφαρμογή είχαν συνοχή σε έναν ικανοποιητικό βαθμό. Σίγουρα στις αποφάσεις τους έπαιξε ρόλο η αμεσότητα και η

Σχήμα Δ'.12: Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις για τη λίστα με μεγάλη συσχέτιση μεταξύ των τραγουδιών.

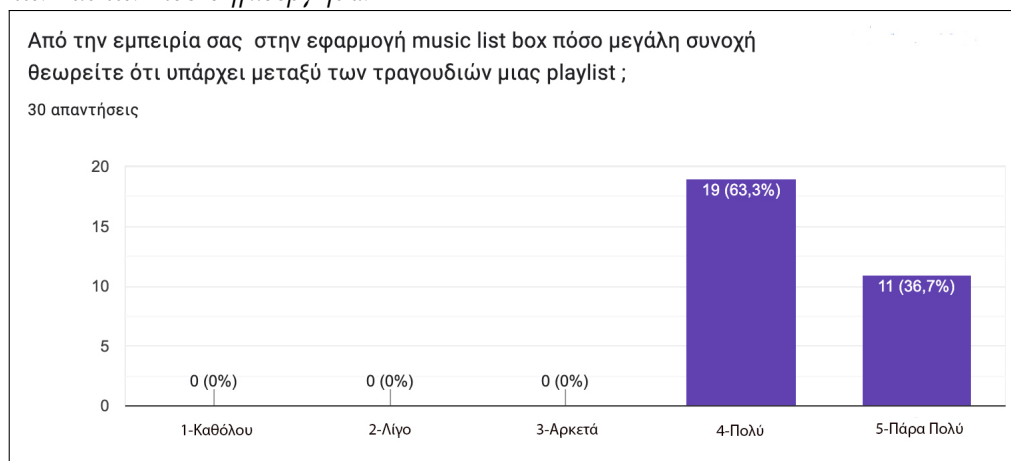


Σχήμα Δ'.13: Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις για τη λίστα που δημιούργησε ο αλγόριθμος της εφαρμογής.



ευκολία που προσφέρει η εφαρμογή όσον αφορά τη δημιουργία των playlist, αφού αξιολόγησαν το περιβάλλον της εφαρμογής θετικά σε μεγάλο βαθμό.

Σχήμα Δ'.14: Διάγραμμα το οποίο παρουσιάζει τις απαντήσεις των χρηστών για τη συνοχή των λιστών που δημιούργησαν.



Κεφάλαιο Ε΄

ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΗ ΑΝΑΠΤΥΞΗ

Ε΄.1 Συμπεράσματα

Κάνοντας μία αναδρομή σε όλη τη διαδικασία και όλα τα στάδια που χρειάστηκαν να ολοκληρωθούν για την υλοποίηση της εφαρμογής, αντιμετωπίστηκαν αρκετές δυσκολίες και αρκετές καταστάσεις που χρειάστηκαν σκέψη. Η ανάπτυξη μίας εφαρμογής ξεκινώντας από την αρχή είναι μια μοναδική εμπειρία, αν σκεφτεί κανείς πόσα εργαλεία χρειάζεται να γνωρίζει ένας προγραμματιστής για να πετύχει τον στόχο του. Εξερευνώντας τα διάφορα apis διαδικτυακών σταθμών, παρατηρήθηκε ότι οι περισσότεροι ελληνικοί σταθμοί δεν διαθέτουν την πληροφορία σε endpoints και πολλοί από αυτούς έχουν σταματήσει να το κάνουν. Δοκιμάστηκαν αρκετοί σταθμοί οι οποίο κατά τη διάρκειά της υλοποίησης σταμάτησαν να προσφέρουν την απαιτούμενη πληροφορία λόγω τεχνικών προβλημάτων. Επίσης παρατηρήθηκε ότι για τους σταθμούς του εξωτερικού υπήρχαν πολύ περισσότερα διαθέσιμα apis με αρκετά οργανωμένη την πληροφορία.

Μέσα από τη συγκεκριμένη εφαρμογή έγινε μια προσπάθειά να μπορέσει να προσφέρει μουσική πληροφορία σε χρήστες, να βοηθήσει τους χρήστες να πειραματιστούν με τη μουσική και να πρώτο ανακαλύψουν τραγούδια τα οποία δεν γνώριζαν πριν χρησιμοποιήσουν το συγκεκριμένο εργαλείο.

Τέλος αν αναφερθούμε ακαδημαϊκά από το 1974 μέχρι και σήμερα έχουν γίνει αρκετές προ-

σπάθειες δημιουργίας καλύτερων και πιο στοχευμένων λιστών αναπαραγωγής μουσικής. Πολλές προσπάθειες και έρευνες στοχεύουν στην καλύτερη ταξινόμηση των μουσικών τραγουδιών σε μία λίστα ανάλογα με το είδος μουσικής, το συναίσθημα του χρήστη, τις προτιμήσεις του χρήστη το περιεχόμενο του τραγουδιού. Σήμερα ο περισσότερος κόσμος ακούει μουσική μέσα από τις streaming πλατφόρμες και εφαρμογές. Στις περισσότερες πλατφόρμες σύμφωνα από έρευνες οι αλγόριθμοι αυτόματης αναπαραγωγής που χρησιμοποιούνται για να δημιουργήσουν μουσικές λίστες φαίνεται να λειτουργούν με βάση τη δημοτικότητα του κάθε τραγουδιού σύμφωνα από τις έρευνες που αναλύονται στο κεφάλαιο Β. Επίσης οι αυτοματοποιημένες playlist και η δημιουργία τους είναι ένα θέμα που απασχολεί εδώ και πολλά χρόνια την επιστήμη της πληροφορικής αφού μπορούν να μελετηθούν πολλά χαρακτηριστικά. Μέχρι και σήμερα οι αλγόριθμοι που έχουν αναπτυχθεί για τη δημιουργία playlist δεν έχουν τελειοποιηθεί. Όσον αφορά την προσέγγιση που παρουσιάστηκε στην παρούσα εργασία τα αποτελέσματα αξιολογήθηκαν θετικά από τους διάφορους χρήστες του συστήματος, παρόλα αυτά χρειάζονται πολλά βήματα ακόμα για να μπορέσει να χρησιμοποιηθεί ο συγκεκριμένος αλγόριθμος σε μεγαλύτερα μουσικά συστήματα και σίγουρα χρειάζεται περισσότερη μελέτη και έρευνα για φτάσει σε ένα ιδανικό επίπεδο το οποίο θα το κάνει να προτιμηθεί από άλλες παρόμοιες συσχετίσεις.

Ε'.2 Επέκταση εφαρμογής και μελλοντικά features

Η συγκεκριμένη εφαρμογή βασίζεται πολύ στις πηγές δεδομένων, αφού ο στόχος της είναι η συνεχής ενημέρωση της βάσης δεδομένων με νέες εγγραφές (τραγούδια) για να μπορεί να προσφέρει στον χρήστη μουσικές λίστες. Μία σημαντική μελλοντική επέκταση της εφαρμογής είναι η προσθήκη περισσότερων σταθμών. Η δομή της εφαρμογής βοηθάει αρκετά στο να προστεθούν περισσότερες πηγές δεδομένων. Η συγκεκριμένη επέκταση είναι ιδιαίτερα σημαντική αν σκεφτεί κανείς ότι θα υπάρχει μεγαλύτερη ποικιλία τραγουδιών για τον χρήστη και θα δημιουργούνται playlists για περισσότερα είδη μουσικής. Περισσότερες λίστες τραγουδιών οι οποίες θα δημιουργούνται με βάση το είδος μουσικής η ακόμα και ανά καλλιτέχνη. Για κάθε καλλιτέχνη χρήσιμο θα ήταν να δημιουργηθεί ένα προφίλ όπου θα εμφανίζονται πληροφορίες για τον ίδιο αλλά και για τα τραγούδια που έχουν εμφανιστεί στους σταθμούς μαζί με διάφορα στατιστικά τα οποία προέκυψαν από την ανάλυση των δεδομένων που βρίσκονται στη βάση δεδομένων. Για παράδειγμα πόσες φορές την ημέρα κατά μέσο όρο εμφανίζεται ο συγκεκριμένος καλλιτέχνης σε συγκεκριμένους σταθμούς ή ποια τραγούδια του συναντάμε πιο συχνά. Μεγάλο ρόλο στη μελλοντική χρήση της εφαρμογής θα είχε σίγουρα η εισαγωγή

ενός player όπου ο χρήστης θα μπορεί να ακούσει μουσική μέσα από το περιβάλλον της εφαρμογής κάνοντας την και περισσότερο διαδραστική προς αυτόν. Η εφαρμογή έχει δημιουργηθεί χωρίς σύστημα εισόδου και εγγραφής χρήστη. Αυτό αποφασίστηκε διότι ο βασικός στόχος ήταν τα αποτελέσματα να είναι άμεσα με τους χρήστες και να μην υπάρχουν άσκοπα ενδιάμεσα βήματα. Μετά από σκέψη ίσως θα έπρεπε να υπάρχει η δυνατότητα δημιουργίας λογαριασμού στο σύστημα. Ένας από τους βασικούς λόγους για να προστεθεί αυτό το feature είναι η αποθήκευση των playlist ανά χρήστη στο σύστημα αφού θα μπορεί να αποθηκεύσει όλες playlist θέλει, να τις επεξεργαστεί, να δώσει τίτλους ή να τις στείλει σε άλλους χρήστες μέσα από τα μέσα κοινωνικής δικτύωσης. Η συγκεκριμένη εφαρμογή είναι μια διαδικτυακή εφαρμογή η οποία μπορεί να υποστηριχτεί και σε μικρότερες αναλύσεις οθονών. Ένας μελλοντικός στόχος είναι να μπορεί να υποστηριχτεί και από όλες τις οθόνες κινητών τηλεφώνων (smartphones), ίσως με διαφορετικό design από την Desktop και την Tablet έκδοση κρατώντας μόνο την ίδια παλέτα χρωμάτων και εικόνες. Ακόμα θα μπορούσαν να προστεθούν περισσότεροι αλγόριθμοι δημιουργίας playlist. Θα μπορούσαν να παρθούν πληροφορίες από διάφορα APIs τα οποία προσφέρουν βασικές πληροφορίες για το περιεχόμενο του τραγουδιού. Συλλογή των ηχητικών χαρακτηριστικών (Audio Features) από το API που προσφέρει το spotify, τα οποία θα μπορούσαν μέσα από διάφορους αλγορίθμους μηχανικής μάθησης (Machine Learning) να προσφέρουν μουσικές λίστες με βάση τις ηχητικές πληροφορίες του τραγουδιού (Web API Libraries, 2022).

Χρήσιμοι Σύνδεσμοι

Αποθετήριο κώδικα εργασίας : <https://github.com/p15diak/thesis.git>

Ερωτηματολόγιο εργασίας : <https://docs.google.com/forms/d/e/1FAIpQLSdfRqppd8aKyYnCAwev9UX9FwrZriMmj9aRPTqgKk3gr0xuWw/viewform>

Σελίδα Εργασίας : <https://p15diak.web-development.services/>

Βιβλιογραφία

What is Cron Job?, (2022). <https://www.hivelocity.net/kb/what-is-cron-job/>

What is LAMP Stack?, (2022). <https://phoenixnap.com/kb/what-is-a-lamp-stack>

Jannach, D., Kamehkhosh, I., Bonnin, G. (2016, July). Biases in automated music playlist generation: A comparison of next-track recommending techniques. In Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization (pp. 281-285).

Jannach, D., Lerche, L., Kamehkhosh, I., Jugovac, M. (2015). What recommenders recommend: an analysis of recommendation biases and possible countermeasures. User Modeling and User-Adapted Interaction, 25(5), 427-491.

Vatolkin, I., Bonnin, G., Jannach, D. (2016). Comparing audio features and playlist statistics for music classification. In Analysis of Large and Complex Data (pp. 437-447). Springer, Cham.

Vatolkin, I. (2013). Improving supervised music classification by means of multi-objective evolutionary feature selection (Doctoral dissertation).

Ikedo, S., Oku, K., Kawagoe, K. (2016, July). Music playlist recommendation using acoustic-feature transitions. In Proceedings of the Ninth International C* Conference on Computer Science Software Engineering (pp. 115-118).

Andric, A., Haus, G. (2006). Automatic playlist generation based on tracking

user's listening habits. *Multimedia Tools and Applications*, 29(2), 127-151.

Chi, C. Y., Tsai, R. T. H., Lai, J. Y., Hsu, J. Y. J. (2010, November). A reinforcement learning approach to emotion-based automatic playlist generation. In *2010 International Conference on Technologies and Applications of Artificial Intelligence* (pp. 60-65). IEEE.

DuBois, P. (2008). *MySQL*. Pearson Education.

APACHE (2022) <https://httpd.apache.org/>

What is PHP?, (2022). <https://www.php.net/manual/en/intro-what-is.php>

What is PHP? The PHP Programming Language Meaning Explained, (2022). <https://www.freecodecamp.org/news/what-is-php-the-php-programming-language-meaning-explained/>

PHP .(2022)/<https://el.wikipedia.org/wiki/PHP>

How Laravel implements MVC and how to use it effectively, (2022). <https://blog.pusher.com/laravel-mvc-use/>

Why Laravel Framework is the Best Choice for PHP Web Development, (2022). <https://www.netsolutions.com/insights/laravel-framework-benefits/>

Why Choose Laravel for Your Next Web Project,(2022) [.https://www.fastfwd.com/why-choose-laravel-for-your-next-web-project/](https://www.fastfwd.com/why-choose-laravel-for-your-next-web-project/)

An Overview of the Best Laravel Security Practices,(2022). <https://www.cloudways.com/blog/laravel-security/>

What is an API endpoint?,(2022). <https://www.techtarget.com/searchapparchitecture/definition/API-endpoint>

Scheduling Tasks with Cron Jobs, (2022). <https://code.tutsplus.com/tutorials/scheduling-tasks-with-cron-jobs-net-8800>

Task Scheduling, (2022). <https://laravel.com/docs/9.x/scheduling>

The Benefits of System Analysis, (2022). <https://www.ftias.com/benefits-system-analysis/>

Front End vs. Back End: What's the Difference?,(2022). <https://kenzie.snhu.edu/blog/front-end-vs-back-end-whats-the-difference/>

Database: Migrations, (2022). <https://laravel.com/docs/9.x/migrations>

Eloquent ORM, (2022). <https://laravel.com/docs/9.x/eloquent-serializationserializing-models-and-collections>

Eloquent: Relationships, (2022). <https://laravel.com/docs/9.x/eloquent-relationships>

Controller, (2022). <https://laravel.com/docs/9.x/controllersmain-content>

Blades, (2022). <https://laravel.com/docs/9.x/blademain-content>

Database Seeding, (2022). <https://laravel.com/docs/4.2/migrationsdatabase-seeding>

Livewire, (2022). <https://laravel-livewire.com/> Recursion .(2022) <https://www.geeksforgeeks.org/recursion/>

Web API Libraries, (2022). <https://developer.spotify.com/documentation/web-api/libraries/>