ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Διαδικτυακή εφαρμογή για τη συλλογή, αποθήκευση, διανομή και παρακολούθηση δεδομένων προσλήψεων αναπληρωτών εκπαιδευτικών»

Του φοιτητή
Σιδηρόπουλου Δήμου
Αρ. Μητρώου: 174980

Επιβλέπων
Στέφανος Ουγιάρογλου
Επίκουρος Καθηγητής

Σεπτέμβριος 2023

Τίτλος Π.Ε. Διαδικτυακή εφαρμογή για τη συλλογή, αποθήκευση, διανομή και παρακολούθηση δεδομένων προσλήψεων αναπληρωτών εκπαιδευτικών

Κωδικός Δ.Ε. 22347

Ονοματεπώνυμο φοιτητή Δήμος Σιδηρόπουλος

Ονοματεπώνυμο εισηγητή Στέφανος Ουγιάρογλου

Ημερομηνία ανάληψης Δ.Ε. 28-11-2022

Ημερομηνία περάτωσης Δ.Ε. 09-09-2023

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 πτυχιακή εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

«Στην οικογένειά μου»

Πρόλογος

Κάθε χρόνο το ελληνικό κράτος προσλαμβάνει σε πολλαπλές φάσεις όπως ονομάζονται χιλιάδες εκπαιδευτικούς ως αναπληρωτές για να καλύψουν τα κενά στην πρωτοβάθμια και δευτεροβάθμια εκπαίδευση. Ο τρόπος με τον οποίο γίνονται αυτές οι προσλήψεις είναι μέσα από πολλαπλές ανακοινώσεις μέσα στη χρονιά και την έκδοση αρχείων excel όπου αναγράφεται ποιοι εκπαιδευτικοί θα κατανεμηθούν σε ποιες περιοχές. Το βασικό πρόβλημα που υπάρχει είναι ότι αν και η πληροφορία είναι ανοιχτή, δεν είναι ακριβώς προσβάσιμη. Για να βρει ο χρήστης οποιαδήποτε πληροφορία εκτός του που κατατέθηκε κάποιος συγκεκριμένος εκπαιδευτικός θα πρέπει να κάνει εκτενή έρευνα και να εξετάσει πολλά αρχεία τα οποία μάλιστα βγαίνουν και σε διαφορετικές ημερομηνίες κάθε χρόνοι. Κινείται στα τυφλά, καθώς ο αριθμός των εκπαιδευτικών που προσλέιφθονται αλλά και των εκπαιδευτικών που εισέρχεται στους πίνακες κάθε χρόνο δεν ειναι σταθερος. Αυτή η κατάσταση σε συνδυασμό με την φύση του επαγγέλματος το οποίο πλέον έχει σαν βασικό χαρακτηριστικό την πιθανή μετακίνηση από περιοχή σε περιοχή εντείνει το αίσθημα αβεβαιότητας του εκπαιδευτικού για το αν θα καταταχθεί, που θα καταταχθεί και για το πως θα εξελιχθει η καριερα του μελλοντικα.

Περίληψη

Το θέμα της παρούσας πτυχιακής εργασίας είναι η δημιουργία μιας web εφαρμογής άλλα και ενός ανοιχτού web API τα οποία θα βοηθήσουν στην εξαγωγή χρήσιμης πληροφορίας απο τα δεκάδες αρχεία προσλήψεων αναπληρωτών που έχουν αναρτηθεί τα τελευταία χρόνια και θα αναρτηθούν στο μέλλον. Πιο συγκεκριμένα, μετά την συγκομιδή πολλών αρχείων και την κατάλληλη μετατροπή τους ώστε να γεμίσουν μία βάση δεδομένων δημιουργήθηκε ένα web API και μιά web εφαρμογή με κύριο σκοπό την οπτικοποίηση των δεδομένων αυτών και την εξαγωγή συμπερασμάτων. Σκοπός ήταν η αχανής πληροφορία που δίνεται δημόσια να γίνει όντως προσβάσιμη και με μία απλή αναζήτηση να μπορεί ο κάθε εκπαιδευτικός να αντλήσει συμπεράσματα σχετικά με τις τάσεις που επικρατούν στις προσλήψεις τα τελευταία χρόνια, τους ρυθμούς ανάπτυξης του κάθε κλάδου ή τις πιθανότητες του να προσληφθεί σε κάποια περιοχή. Στόχος ήταν να μειωθεί η αβεβαιότητα που αντιμετωπίζουν οι εκπαιδευτικοί κάθε χρονιά σχετικα με τον μέλλον τους άλλα και να υπάρχει ένα εργαλείο που θα τους βοηθήσει στην λήψη αποφάσεων.

«Web application for the collection, storage, distribution and monitoring of non-permanent teaching staff recruitment data»

«Dimos Sidiropoulos»

Abstract

The topic of this thesis is the creation of a web application and an open web API that will help to extract useful information from the dozens of recruitment files that have been posted in recent years and will be posted in the future. More specifically, after harvesting many files and converting them to fill a database, a web API and a web application were created with the main purpose of visualizing this data and drawing conclusions. The aim was to make the vast information given publicly accessible and with a simple search any teacher could draw conclusions about the trends in recruitment in recent years, the growth rates of each sector or the chances of being recruited in a particular area. The aim was to reduce the uncertainty that teachers face each year about their future but also to have a tool to help them in their decision making.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένεια μου για την στήριξη τους κατα την διάρκεια των σπουδών μου.

Περιεχόμενα

Κατάλογος Εικόνων

Συντομογραφίες

Π.Ε.	Πτυχιακή Εργασία
I/O	Input/Output
A.I	Artificial Intelligence
M.L	Machine Learning
IoT	Internet Of Things
PWA	Progressive Web Apps
CRUD	Create Read Update Delete

Κεφάλαιο 1ο: Εισαγωγή

1.1 Αναπληρωτές Εκπαιδευτικοί Δευτεροβάθμιας και Πρωτοβάθμιας Εκπαίδευσης

Εκπαιδευτικοί κάθε κλάδου και ειδικότητας, που εργάζονται στην πρωτοβάθμια ή στη δευτεροβάθμια εκπαίδευση, καλούνται, είτε με την έναρξη είτε κατά τη διάρκεια του εκάστου σχολικού έτους να εργαστούν ως αναπληρωτές. Την τελευταία δεκαετία και πλέον μάλιστα, λόγω των αλλαγών στα εργασιακά πεπραγμένα, από πλευράς κεντρικής διοίκησης και διαχείρισης, και εξαιτίας της βαθιάς κρίσης που βίωσε και βιώνει η χώρα, ο αριθμός των αναπληρωτών δασκάλων και καθηγητών αυξάνεται συνεχώς. Ο θεσμός, βέβαια, ερευνήθηκε σε βάθος χρόνου, με αναφορές σε αναπληρωτές εκπαιδευτικούς να υπάρχουν σε νομοθετικές ρυθμίσεις από το 1929 έως το 1949. Η εργασιακή, ωστόσο, κατάσταση των αναπληρωτών και οι επαγγελματικές προοπτικές τους καθορίστηκαν με νομοθετήματα από το 1949 έως και τον Ν.3848/2010. Τη δε ιλιγγιώδη αύξηση του αριθμού των αναπληρωτών αποδεικνύουν και οι στατιστικοί δείκτες, σύμφωνα με τους οποίους οι αναπληρωτές εκπαιδευτικοί αποτελούν πλέον το 20% των εκπαιδευτικών που εργάζονται στην Πρωτοβάθμια και το 10% των εκπαιδευτικών που απασχολούνται στη Δευτεροβάθμια εκπαίδευση.

Τι ακριβως είναι όμως οι αναπληρωτές; Αναπληρωτές ονομάζονται οι εκπαιδευτικοί εκείνοι που εργάζονται στις σχολικές μονάδες, ελλείψει μόνιμων δασκάλων (λόγω οργανικού κενού, ασθενείας, εγκυμοσύνης, εκπαιδευτικής άδειας κ.ά.), για συγκεκριμένο και περιορισμένο χρονικό διάστημα. Εργάζονται δε από την ημέρα της πρόσληψής τους (οποτεδήποτε μέσα στο διάστημα της σχολικής χρονιάς παραστεί ανάγκη και κληθούν) έως το πέρας του σχολικού έτους (τέλη Ιουνίου).

Ο αναπληρωτής βιώνει ένα κλίμα αβεβαιότητας που επικρατεί στις τάξεις των εκπαιδευτικών όλων των ειδικοτήτων, κυρίως των νεώτερων, ή/και σε ηλικία ή/και σε εμπειρία. Και η εν λόγω διατύπωση δεν είναι τυχαία, καθώς στη θέση του αναπληρωτή εκπαιδευτικού δε βρίσκονται πια μόνο οι νεόκοποι, αλλά υπηρετούν κάθε χρόνο και εκπαιδευτικοί με πολυετή εμπειρία (δεκαετίας και πλέον), ενδεχομένως με οικογένεια και παιδιά.

Ο αριθμός των αναπληρωτών εκπαιδευτικών, που προσλαμβάνονται κάθε χρόνο στη Γενική και την Ειδική Αγωγή και Εκπαίδευση, πρωτοβάθμια και δευτεροβάθμια, είναι πολύ μεγάλος. Αυξάνεται δε χρόνο με τον χρόνο, λόγω της κρίσης και του οικονομικού και εργασιακού αδιεξόδου που επικρατεί στη χώρα. Αυτοί που συνταξιοδοτούνται, ορισμένοι εξ αυτών εσπευσμένα, για να προλάβουν τυχόν περικοπές στα συντάξιμα εισοδήματά τους, δεν αντικαθίστανται από νεοδιορισθέντες εκπαιδευτικούς. Οι θέσεις τους καλύπτονται από αναπληρωτές, όπως συμβαίνει και με τις νέες θέσεις, που δημιουργούνται λόγω της αύξησης ή της ποικιλίας και της διαφοροποίησης του μαθητικού πληθυσμού στα σχολεία (π.χ. ανάγκες για δημιουργία Τμημάτων Ένταξης και για εκπαιδευτικούς Παράλληλης Στήριξης μαθητών με ειδικές εκπαιδευτικές ανάγκες και ικανότητες, Τμημάτων Υποδοχής για παλιννοστούντες και αλλοδαπούς μαθητές, πρόσφυγες και μετανάστες κ.ά.)[1].

1.2 Διαδικασία πρόσληψης Αναπληρωτών Εκπαιδευτικών

Οι αναπληρωτές εκπαιδευτικοί καλούνται, όπως προαναφέρθηκε, να εργαστούν στην αρχή κάθε νέου σχολικού έτους ή και κατά τη διάρκεια της χρονιάς. Τι προηγείται, όμως, και τι έπεται της ανάληψης υπηρεσίας; Αρχικά, οι εκπαιδευτικοί θα πρέπει ηλεκτρονικά πρώτα να δηλώσουν το ενδιαφέρον τους για πρόσληψη στο σύστημα του ΟΠΣΥΔ, μιας ηλεκτρονικής πλατφόρμας που υπάρχει για αυτόν τον

σκοπό στο Πανελλήνιο Σχολικό Δίκτυο, εντός μιας συγκεκριμένης προθεσμίας, που προκηρύσσεται από το Υπουργείο Παιδείας. Μετά το πέρας λίγων μηνών και όσο η νέα σχολική χρονιά πλησιάζει, οι εκπαιδευτικοί καλούνται να επισκεφτούν εκ νέου τη συγκεκριμένη σελίδα και να δηλώσουν περιοχές προτίμησης εργασίας (Αττική, Θεσσαλία, Δωδεκάνησα κ.τλ.). Η διαδικασία μοιάζει λίγο με το μηχανογραφικό που καλούνται να υποβάλουν οι τελειόφοιτοι του Λυκείου μετά το πέρας των Πανελληνίων. Έπειτα, αναμονή και αγωνία, καθώς οι εκπαιδευτικοί θα πρέπει να είναι σε επιφυλακή για την ανακοίνωση διορισμού. Ως τότε, είναι ένας αριθμός σε ένα λογιστικό φύλλο, σε έναν πίνακα, σε μια λίστα αναμονής για προσωρινό διορισμό.

Η ανακοίνωση των αποτελεσμάτων γίνεται μέσω τεράστιων αρχείων τύπου excel, ένα για κάθε ειδικότητα ή με διαχωρισμό γενικής και ειδικής εκπαίδευσης ή και ακόμα πιο γενικά με διαχωρισμό πρωτοβάθμιας και δευτεροβάθμιας εκπαίδευσης. Μόλις ανακοινωθεί η ημερομηνία διορισμού ενός αριθμού αναπληρωτών, διαδικασία που επαναλαμβάνεται αρκετές φορές μέσα στη σχολική χρονιά, οι συγκεκριμένοι εκπαιδευτικοί θα πρέπει να παραδώσουν κάποια δικαιολογητικά αυτοπροσώπως στη Διεύθυνση Εκπαίδευσης του τόπου που έχουν προσληφθεί εντός τριών ημερών. Να σημειωθεί ότι αυτή η διαδικασία επαναλαμβάνεται κάθε χρόνο, ακόμα και για αναπληρωτές που έχουν υπηρετήσει στην ίδια Διεύθυνση επανειλημμένως. Έπειτα, σε εξίσου πολύ σύντομο διάστημα (ενδέχεται και εντός δύο ημερών) πρέπει να βρεθούν στο σχολείο τους και να αρχίσουν το έργο τους. Μετά δε το πέρας του σχολικού έτους (τέλη Ιουνίου), παρουσιάζονται και πάλι στη Διεύθυνση που τους προσέλαβε, παραλαμβάνουν την απόλυσή τους και πλέον, ως άνεργοι, επαναλαμβάνουν την παραπάνω διαδικασία εκ νέου, προκειμένου να προσληφθούν και την ερχόμενη χρονιά.

1.3 Κίνητρο

Το κίνητρο αυτής της πτυχιακής εργασίας ήταν η οργάνωση των δεδομένων προσληψεων και η οπτικοποίηση τους ώστε να γίνεται πιο εύκολα η κατανόησή τους και η εξαγωγή συμπερασμάτων. Το πρόβλημα που υπάρχει αρχικά είναι ότι το υπουργείο κάθε χρόνο βγάζει μια πληθώρα αρχείων χωρίς κάποιο συγκεκριμένο αριθμό και ανα διάφορες ημερομηνίες μέσα στην χρονιά, κάθε χρονιά. Η πληροφορία αυτή αν και ανοιχτή προς πρόσβαση για όλους δεν είναι κάπου συγκεντρωμένη και οργανωμένη.

Έτσι για την αναζήτηση κάποια πληροφορίας θα πρέπει πολλές φορές να ανοιχτούν πολλά από αυτά τα excel αρχεία. Παραδείγματος χάρη το να δει κάποιος την πολύ χρήσιμη πληροφορία για το ποια είναι τα ελάχιστα μόρια για την πρόσληψη κάποια ειδικότητας εκπαιδευτικού σε μια συγκεκριμένη περιοχή για κάθε χρονιά θα χρειαστεί να ψάξει σε μια πληθώρα excel αρχείων. Πιο συγκεκριμένα, δεν φτάνει απλά ψάξει να βρει τα αρχεία αυτά για κάθε χρονιά και να ψάξει γραμμή-γραμμή για την συγκεκριμένη αυτή ειδικότητα στην συγκεκριμένη περιοχή, αλλα θα πρέπει και να βρει όλα τα excel αρχεία που αναρτήθηκαν εκείνη την χρονιά. Η αναζήτηση αυτή θα είναι απίστευτα χρονοβόρα άλλα και αν δεν γίνει με τεράστια προσοχή, λανθασμένη.

Πολύ σημαντικό για τους ίδιους τους εκπαιδευτικούς είναι και η τάση των προσλήψεων. Πόσοι μουσικοί προσλήφθηκαν στην Κρήτη ανα χρονιά; Υπάρχει αύξηση ή μείωση; Με τι ταχύτητες κινούνται οι προσλήψεις στον τομέα της ειδικής αγωγής; Αυξάνονται ή υπάρχει κορεσμός; Αυτά είναι κάποια από τα πολλά ερωτήματα των εκπαιδευτικών. Ακόμα και τα δεδομένα το ίδιου του εκπαιδευτικού δεν υπάρχουν κάπου μαζεμένα και θα πρέπει να ανατρέχει πάλι σε αυτά τα excel αρχεία για να βρει παλαιότερους του διορισμούς.

Δεν υπάρχει δηλαδή η δυνατότητα κάποιου εκπαιδευτικού να δει κάπου οργανωμένη αυτή την πληροφορία, ώστε να μπορεί να κάνει προβλέψεις για το μέλλον ή γενικά για το μέλλον κάποιας ειδικότητας, γενικά δεν μπορεί να βγάλει συμπεράσματα. Υπάρχει λοιπόν μεγάλη ανάγκη για συγκομιδή, καθαρισμό, οργάνωση και οπτικοποίηση αυτής της πληροφορίας ακριβώς για τους λόγους που παρουσιάστηκαν παραπάνω.

1.4 Συνεισφορά

Η συνεισφορά αυτής της πτυχιακής εργασίας είναι η δημιουργία μιας web εφαρμογής με όνομα eAnaplirotes αλλά και ενός web API προς επίλυση των προβλημάτων που αναφέρθηκαν.

Η εφαρμογή είναι αυτή που οπτικοποιεί τα δεδομένα και χωρίζεται σε δύο κομμάτια. Το πρώτο είναι η ονομαστική αναζήτηση, δηλαδή η αναζήτηση ενός συγκεκριμένου ατόμου με το ονοματεπώνυμό του. Το αποτέλεσμα κάθε τέτοιας αναζήτησης είναι η συγκεντρωτική προβολή του ιστορικού των διορισμών του. Η πληροφορία αυτή παρουσιάζεται συγκεντρωτικά σε ένα πίνακα για κάθε χρονιά που υπάρχουν δεδομένα αλλα και με την χρήση γραφημάτων. Έτσι γίνεται και η οπτικοποίηση της πληροφορίας με τρόπο που γίνεται εύκολα κατανοητή από τον χρήστη και μπορεί έτσι να δει τις τάσεις στην σειρά που ήταν στον πίνακα ανα χρονιά, το πόσο γρήγορα ανεβαίνει, αν ανεβαίνει και δεν είναι στάσιμος, τι επίδραση μπορεί να είχαν στην κατάταξη τα μόρια που πήρε κάθε χρονια και γενικά αν οι αποφάσεις που έλαβε κάθε χρονιά όσον αφορά σε ποια φάση διορίστηκε, σε ποιες περιοχές επέλεξε να διοριστεί ή πως εφοδιάστηκε απο πλευράς πτυχίων ή πιστοποιητικών επηρέασαν την κατάταξή του στους πίνακες. Γράφημα εκτός από την σειρά πίνακα του εκπαιδευτικού υπάρχει και για τα μόρια του ανα χρονιά.

Το δεύτερο κομμάτι είναι η γενική αναζήτηση. Στην γενική αναζήτηση μπορεί να γίνει εισαγωγή τεσσάρων τύπων πληροφορίας, του τύπου (γενικής, ειδικής κλπ.), του κλάδου, της περιοχης τοποθέτησης και την περιοχή διεύθυνσης. Μπορεί να γίνει χρήση ενός τύπου πληροφορίας ή και ακόμα όλων ταυτόχρονα. Η πληροφορία που επιστρέφεται είναι το πλήθος των αναπληρωτών που προσλήφθηκαν. Παραδείγματος χάρη πόσοι εκπαιδευτικοί ειδικής αγωγής προσλήφθηκαν στα Χανιά, πόσοι δάσκαλοι αγγλικών προσλήφθηκαν πανελλαδικά, πόσους αναπληρωτές είχε η δευτεροβαθμια διευθυνση Ξανθης κ.α. Μάλιστα, στην περίπτωση που γίνει χρήση του κλάδου ταυτόχρονα με την περιοχή εμφανίζεται και η πληροφορία για το πόσα ήταν τα ελάχιστα μόρια για την πρόσληψη ενος εκπαιδευτικού του συγκεκριμένου αυτού κλάδου στην συγκεκριμένη περιοχή. Και εδώ τα αποτελέσματα εμφανίζονται και σε μορφή κειμένου μέσα σε έναν συγκεντρωτικό πίνακα αλλά και μέσω διαγραμμάτων

Το API τώρα, που είναι ανοιχτό προς όλους προσφέρει στοχευμένη πληροφορία για όλες τις διευθύνσεις εκπαίδευσης πανελλαδικά, τις περιοχές, τους κλάδους κ.α. Επίσης ικανοποιεί και σύνθετα ερωτήματα όπως αυτά που αναφερθηκαν παραπανω. δηλαδή πληροφορία για τις τάσεις που επικρατούν ανα σχολική χρονιά όσον αφορά τις προσλήψεις και τα ελάχιστα μόρια πρόσληψης.

1.5 Οργάνωση της εργασίας

Η συγκεκριμένη πτυχιακή εργασία χωρίζεται σε έξι κεφάλαια, που αφορούν τις τεχνολογίες που χρησιμοποιήθηκαν, το τρόπο που αντλήθηκαν τα δεδομένα και γέμισαν την βάση δεδομένων, τον σχεδιασμό της εφαρμογής στο back και στο front κομμάτι, την παρουσίαση του τελικού προϊόντος της εφαρμογής και τέλος κάποιες τελικές σκέψεις μετα την υλοποίηση της εφαρμογής αυτής όπως και μελλοντικών επεκτάσεων. Πιο συγκεκριμένα:

Στο δεύτερο κεφάλαιο της Π.Ε γίνει εκτενής ανάλυση των τεχνολογιών που χρησιμοποιήθηκαν για την μετατροπή των αρχείων στην κατάλληλη μορφή και το ανέβασμα τους μετά στην βάση, των τεχνολογιών που χρησιμοποιήθηκαν στην βάση, όπως και στο backend και frontend της εφαρμογής περιγράφοντας εκτενώς τα μειονεκτήματα και τα πλεονεκτήματα κάθε τεχνολογίας όπως και πεδία χρήσης τους.

Στο τρίτο κεφάλαιο, γίνεται αναλυτική περιγραφή όλου του κύκλου ζωής των δεδομένων της εφαρμογής και τον τρόπο που αυτά αποθηκεύονται στη βάση. Πως αντλούνται αυτά τα δεδομένα, ποια είναι η αρχική τους μορφή και ποια προβλήματα συναντήθηκαν στα αρχικά αυτά δεδομένα. Πως αποφασίστηκε τελικά να δημιουργηθουν και πως με την υλοποίηση ενός python script ανέβηκαν τελικά στη βάση. Επίσης αναλύεται ποια είναι η τελική μορφή της βασης, ποιοι είναι οι πίνακες και πως συνδέονται μεταξύ τους.

Στο τέταρτο κεφάλαιο περιγράφεται αναλυτικά η σχεδίαση και η υλοποίηση της εφαρμογής. Ποιες τεχνολογίες χρησιμοποιήθηκαν στο backend και στο frontend κομμάτι της εφαρμογης και πως αξιοποιήθηκαν. Παρατίθενται κομμάτια κώδικα και εξηγούνται ώστε να κατανοηθεί πλήρως το πως λειτουργεί η εφαρμογή, ποιες τακτικές ακολουθήθηκαν και γενικά πως το back και το front συνδέονται μεταξύ τους.

Στο πέμπτο κεφάλαιο παρουσιάζεται το τελικό προϊόν. Ποιο είναι δηλαδή το interface που αλληλεπιδρά ο χρήστης, πως γίνεται η εισαγωγή των δεδομένων και τι αυτός βλέπει. Πιο συγκεκριμένα περιγράφονται τα δύο σενάρια χρήσης της εφαρμογής. Το ένα της ονομαστικής αναζήτησης, όπου με την χρήση κάποιου ονοματεπώνυμο αναζητείται κάποιος εκπαιδευτικός συγκεκριμένα άλλα και υποσενάρια που προκύπτουν όπως αυτό της συνωνυμίας. Και το δεύτερο σενάριο όπου ο χρήστης με την χρήση φίλτρων μπορεί ουσιαστικά να παράξει πληροφορία για εκατοντάδες ερωτήματα που αφορά το πλήθος των προσλήψεων άλλα και τα ελάχιστα μόρια πρόσληψης ενός εκπαιδευτικού ενός συγκεκριμένου κλάδου σε μια συγκεκριμένη περιοχή.

Τέλος, στο τελευταίο κεφάλαιο προκύπτουν και περιγράφονται κάποια συμπεράσματα που αφορούν την γενική κατάσταση και τον τρόπο με τον οποίο γίνεται σήμερα η πρόσληψη ενός εκπαιδευτικού, τι προσφέρει αυτή η εφαρμογή πάνω σε αυτό και κάποιες σκέψεις σχετικά με την όλη υλοποίηση της εφαρμογής. Επίσης προτείνονται και κάποιες μελλοντικές επεκτάσεις οι οποίες θα δώσουν στην εφαρμογή μεγαλύτερη αξία και λειτουργικότητα.

Κεφάλαιο 2ο: Τεχνολογίες

2.1 Εισαγωγή

Στην παρούσα πτυχιακή λόγω της φύσης της χρησιμοποιήθηκε μια πληθώρα τεχνολογιών. Πιο συγκεκριμένα Python για την δημιουργία ενός script που θα παίρνει τα δεδομένα των excel και θα γεμίζει μια βάση δεδομένων και MySQL για την διαχείριση αυτής της βάσης δεδομένων. Node.js μαζί με το framework της Express για την δημιουργία του API και γενικά του back-end. Για την δημιουργία του front-end χρησιμοποιήθηκε Vue.js, ένα framework βασισμένο στη javascript και η βιβλιοθήκη Chart.js για την οπτικοποίηση των δεδομένων. Στην συνέχεια παρουσιάζονται αναλυτικά οι τεχνολογίες αυτές.

2.2 Python

Η Python είναι μια ευέλικτη, open-source και υψηλού επιπέδου γλώσσα προγραμματισμού που κυκλοφόρησε για πρώτη φορά το 1991 από τον Guido van Rossum. Από την ίδρυσή της, η Python έχει αυξήσει τη δημοτικότητά της και έχει γίνει μία από τις πιο διαδεδομένες γλώσσες στον κόσμο. Από το 2023, η Python συνεχίζει να αποτελεί βασικό εργαλείο για προγραμματιστές, επιστήμονες δεδομένων, προγραμματιστές διαδικτύου και άλλους.

Ένα από τα πιο αξιοσημείωτα χαρακτηριστικά της Python είναι το πόσο ευανάγνωστη είναι. Το συντακτικό της είναι εύκολα κατανοητό και η χρήση της εσοχής για τον ορισμό μπλοκ κώδικα την καθιστά ιδανική επιλογή για αρχάριους. Το καθαρό, ευανάγνωστο από τον άνθρωπο συντακτικό επιτρέπει στους προγραμματιστές να υλοποιήσουν τις ιδέες τους σε λιγότερες γραμμές κώδικα σε σύγκριση με άλλες γλώσσες όπως η Java ή η C++. Υποστηρίζει ένα ευρύ φάσμα παραδειγμάτων προγραμματισμού, συμπεριλαμβανομένου του διαδικαστικού, του αντικειμενοστραφούς και του λειτουργικού προγραμματισμού. Είναι επίσης μια γλώσσα δυναμικού τύπου, πράγμα που σημαίνει ότι οι προγραμματιστές δεν χρειάζεται να δηλώνουν τον τύπο μιας μεταβλητής πριν τη χρησιμοποιήσουν. Αυτό το χαρακτηριστικό την καθιστά εξαιρετικά ευέλικτη, αλλά μπορεί επίσης να οδηγήσει σε σφάλματα αν ο προγραμματιστής δεν είναι προσεκτικός. Η εκτεταμένη τυπική βιβλιοθήκη της Python προσφέρει ένα ευρύ φάσμα ενοτήτων και πακέτων για διάφορες εργασίες, όπως ο χειρισμός αρχείων, η επεξεργασία δεδομένων, η ανάπτυξη ιστοσελίδων και πολλά άλλα[2], [3].

Η Python χρησιμοποιείται ευρέως για την ανάπτυξη ιστοσελίδων. Τα frameworks Django και Flask είναι τα δύο πιο δημοφιλή που βασίζονται στην Python και επιτρέπουν στους προγραμματιστές να δημιουργούν εφαρμογές ιστού με ευκολία. Το Django ακολουθεί την "plug and play" φιλοσοφία παρέχοντας ένα ευρύ φάσμα λειτουργιών out-of-the-box. Αντίθετα, το Flask είναι ένα μικρο framework που επιτρέπει στους προγραμματιστές να δημιουργούν ελαφριές εφαρμογές ιστού[4].

Η Python έχει επίσης βρει ισχυρά ερείσματα στον τομέα της επιστήμης των δεδομένων. Βιβλιοθήκες όπως οι NumPy, Pandas και Matplotlib καθιστούν την Python μια εξαιρετική επιλογή για αριθμητική ανάλυση, επεξεργασία δεδομένων και οπτικοποίηση. Η βιβλιοθήκη SciPy επεκτείνει τις δυνατότητες της NumPy, προσφέροντας πρόσθετα εργαλεία για επιστημονικούς υπολογισμούς. Οι βιβλιοθήκες μηχανικής μάθησης της Python, όπως η scikit-learn και η TensorFlow, την καθιστούν επίσης και μια ισχυρή γλώσσα για προγνωστικά μοντέλα και την τεχνητή νοημοσύνη.

Είναι και μια εξαιρετική γλώσσα για αυτοματοποίηση. Η εκτεταμένη στάνταρ βιβλιοθήκη και τα modules τρίτων καθιστούν εύκολη την αυτοματοποίηση εργασιών όπως η διαχείριση αρχείων, η

απόξεση ιστού (web scraping) και η αλληλεπίδραση με API. Η απλότητα και η ευκολία χρήσης της Python την καθιστούν ιδανική επιλογή για την αυτοματοποίηση επαναλαμβανόμενων εργασιών, απελευθερώνοντας χρόνο για πιο παραγωγική εργασία.

Η κοινότητα της Python παίζει σημαντικό ρόλο στην επιτυχία της γλώσσας. Το Python Software Foundation (PSF) διαχειρίζεται την ανάπτυξη της γλώσσας και προωθεί τη χρήση της. Η κοινότητα της Python συμβάλλει στην ανάπτυξή της δημιουργώντας νέες βιβλιοθήκες, οργανώνοντας εκδηλώσεις και παρέχοντας υποστήριξη σε νέους χρήστες μέσω φόρουμ και λιστών αλληλογραφίας. Αυτή η ενεργή κοινότητα διασφαλίζει ότι η Python συνεχίζει να εξελίσσεται και να παραμένει επίκαιρη στον συνεχώς μεταβαλλόμενο κόσμο της ανάπτυξης λογισμικού.

Χρησιμοποιείται ευρέως στον κόσμο των επιστημονικών υπολογιστών. Βιβλιοθήκες όπως η SciPy, η NumPy και η Matplotlib έχουν διευκολύνει τους επιστήμονες και τους μηχανικούς να χρησιμοποιούν την Python για αριθμητικούς και συμβολικούς υπολογισμούς, καθώς και για την οπτικοποίηση δεδομένων. Αυτές οι βιβλιοθήκες έχουν μετατρέψει την Python σε ένα ισχυρό εργαλείο για την επιστημονική έρευνα, επιτρέποντας στους ερευνητές να επικεντρωθούν στην εργασία τους χωρίς να χρειάζεται να ανησυχούν για τις πολυπλοκότητες των γλωσσών χαμηλότερου επιπέδου.

Ένας άλλος τομέας όπου η Python έχει κερδίσει δημοτικότητα είναι ο τομέας της τεχνητής νοημοσύνης και της μηχανικής μάθησης. Βιβλιοθήκες όπως η TensorFlow, η Keras και η PyTorch έχουν καταστήσει δυνατό για τους προγραμματιστές να δημιουργούν και να εκπαιδεύουν σύνθετα μοντέλα μηχανικής μάθησης με ευκολία. Αυτές οι βιβλιοθήκες παρέχουν μια διεπαφή υψηλού επιπέδου στις υποκείμενες μαθηματικές πράξεις, διευκολύνοντας τους προγραμματιστές να δημιουργούν μοντέλα χωρίς να χρειάζεται να κατανοούν τις λεπτομέρειες χαμηλού επιπέδου.

Η Python είναι επίσης γνωστή για την εκτεταμένη υποστήριξή της για την επεξεργασία φυσικής γλώσσας (NLP). Βιβλιοθήκες όπως η NLTK (Natural Language Toolkit) και η spaCy παρέχουν εργαλεία για τη δημιουργία συμβόλων (tokenization), την επισήμανση μέρους του λόγου (part-of-speech tagging), την αναγνώριση ονοματικών οντοτήτων (named entity recognition) και πολλά άλλα. Αυτές οι βιβλιοθήκες έχουν διευκολύνει τους προγραμματιστές να δημιουργούν εφαρμογές που μπορούν να κατανοούν και να επεξεργάζονται την ανθρώπινη γλώσσα.

Το web scraping είναι ένας άλλος τομέας στον οποίο η Python μεγαλουργεί. Βιβλιοθήκες όπως η Beautiful Soup και η Scrapy διευκολύνουν την εξαγωγή δεδομένων από ιστότοπους και ιστοσελίδες. Αυτές οι βιβλιοθήκες επιτρέπουν στους προγραμματιστές να δημιουργούν προσαρμοσμένους web scrapers που μπορούν να συλλέγουν αυτόματα δεδομένα από τον ιστό, διευκολύνοντας τη συλλογή μεγάλων ποσοτήτων δεδομένων για ανάλυση ή άλλους σκοπούς.

Η ευελιξία της Python επεκτείνεται και στον κόσμο της ανάπτυξης παιχνιδιών. Βιβλιοθήκες όπως η Pygame και η Panda3D παρέχουν εργαλεία για τη δημιουργία 2D και 3D παιχνιδιών στην Python. Αυτές οι βιβλιοθήκες προσφέρουν μια διεπαφή υψηλού επιπέδου στις υποκείμενες μηχανές γραφικών και φυσικής, διευκολύνοντας τους προγραμματιστές να δημιουργούν παιχνίδια χωρίς να χρειάζεται να κατανοούν τις λεπτομέρειες χαμηλού επιπέδου.

Η ευκολία χρήσης και η ευελιξία της την έχουν καταστήσει επίσης δημοφιλή επιλογή για τη διδασκαλία του προγραμματισμού. Η απλή σύνταξη και η αναγνωσιμότητα της Python την καθιστούν εξαιρετική γλώσσα για αρχάριους. Εκπαιδευτικές πλατφόρμες όπως το Codecademy, το Coursera και το edX προσφέρουν μαθήματα στην Python, βοηθώντας τους ανθρώπους να μάθουν έννοιες και δεξιότητες προγραμματισμού.

Η υποστήριξη που υπάρχει για ανάπτυξη σε πολλαπλές πλατφόρμες είναι ένας άλλος λόγος για τη δημοτικότητά της. Ο κώδικας της Python μπορεί να εκτελεστεί σε Windows, macOS και Linux χωρίς να χρειάζεται κάποια τροποποίηση. Αυτή η υποστήριξη πολλαπλών πλατφορμών διευκολύνει τους προγραμματιστές να δημιουργούν εφαρμογές που μπορούν να τρέξουν σε πολλαπλά λειτουργικά συστήματα χωρίς να χρειάζεται να ξαναγράψουν κώδικα.

Όπως όμως και κάθε γλώσσα προγραμματισμού έτσι και η Python έχει τους περιορισμούς της. Λόγω της δυναμικής τυποποίησης και της διερμηνευμένης φύσης της, η Python μπορεί να είναι πιο αργή από τις μεταγλωττισμένες γλώσσες όπως η C ή η Java. Ωστόσο, οι συμβιβασμοί στην απόδοση είναι συχνά αποδεκτοί για πολλές εφαρμογές, λαμβάνοντας υπόψη τα οφέλη παραγωγικότητας που προσφέρει η Python.

Συνοψίζοντας, η ευελιξία, η ευκολία χρήσης και οι εκτεταμένες βιβλιοθήκες της Python την καθιστούν ένα ισχυρό εργαλείο για ένα ευρύ φάσμα εφαρμογών. Η ενεργή κοινότητα της Python και η υποστήριξη από οργανισμούς όπως το Python Software Foundation διασφαλίζουν ότι η γλώσσα συνεχίζει να εξελίσσεται και να παραμένει σημαντική στον κόσμο της ανάπτυξης λογισμικού. Η αύξηση της δημοτικότητας της Python δεν δείχνει σημάδια επιβράδυνσης και είναι πιθανό να παραμείνει ένα πολύτιμο εργαλείο για τους προγραμματιστές για τα επόμενα χρόνια.

2.3 Node.js

Η Node.js είναι ένα ανοιχτού κώδικα, διαπλατφορμικό περιβάλλον εκτέλεσης που επιτρέπει στους προγραμματιστές να δημιουργούν κλιμακούμενες δικτυακές εφαρμογές. Χρησιμοποιεί τη JavaScript, μια από τις πιο δημοφιλείς γλώσσες προγραμματισμού, καθιστώντας την μια ιδιαίτερα προσιτή πλατφόρμα για τους προγραμματιστές ιστού. Το Node.js εισήχθη το 2009 από τον Ryan Dahl, ο οποίος είχε ως στόχο τη δημιουργία ιστότοπων σε πραγματικό χρόνο με δυνατότητα push, μια τεχνική που έχει γίνει αναπόσπαστο μέρος των σύγχρονων εφαρμογών ιστού.

Όσον αφορά την αρχιτεκτονική της Node.js, στον πυρήνα της, αξιοποιεί τη μηχανή V8 JavaScript της Google, την ίδια μηχανή που τροφοδοτεί το πρόγραμμα περιήγησης Chrome. Η μηχανή V8 λαμβάνει κώδικα JavaScript και τον μεταγλωττίζει απευθείας σε κώδικα μηχανής, με αποτέλεσμα την εκτέλεση υψηλής απόδοσης. Ωστόσο, αυτό που διαφοροποιεί τη Node.js από ένα παραδοσιακό περιβάλλον προγράμματος περιήγησης ιστού είναι το μοντέλο εισόδου/εξόδου που καθοδηγείται από συμβάντα

Σε ένα συμβατικό περιβάλλον διακομιστή πολλαπλών threads, κάθε εισερχόμενη αίτηση πελάτη διεκπεραιώνεται από ένα ξεχωριστό thread. Καθώς αυξάνεται ο αριθμός των αιτήσεων, το σύστημα χρειάζεται να δημιουργήσει περισσότερα threads, με αποτέλεσμα να αυξάνεται η κατανάλωση μνήμης και CPU. Αντίθετα, στη Node.js χρησιμοποιείται μια αρχιτεκτονική με ένα μόνο thread, καθοδηγούμενη από συμβάντα. Αυτό σημαίνει ότι ένα μόνο κύριο thread χειρίζεται όλα τα αιτήματα πελατών καταχωρώντας συμβάντα και τις αντίστοιχες ανακλήσεις τους. Όταν συμβαίνει ένα συμβάν, ενεργοποιείται η κατάλληλη συνάρτηση επανάκλησης, η οποία εκτελείται στο ίδιο κύριο thread. Αυτή η προσέγγιση μειώνει το κόστος δημιουργίας και διαχείρισης πολλαπλών threads, με αποτέλεσμα ένα πιο αποδοτικό και επεκτάσιμο σύστημα.

Ένα από τα πιο ισχυρά χαρακτηριστικά της Node.js είναι το μοντέλο μη μπλοκαρισμένου I/O. Σε ένα μπλοκαρισμένο μοντέλο εισόδου/εξόδου (I/O), ο διακομιστής περιμένει να ολοκληρωθεί κάθε λειτουργία προτού προχωρήσει στην επόμενη. Αντίθετα, ένα μοντέλο μη μπλοκαρισμένου I/O επιτρέπει στον διακομιστή να χειρίζεται πολλαπλές λειτουργίες ταυτόχρονα.

Στη Node.js, όταν ξεκινά μια λειτουργία I/O, όπως η ανάγνωση από ένα αρχείο ή η αναζήτηση σε μια βάση δεδομένων, το κύριο thread συνεχίζει την εκτέλεση άλλου κώδικα χωρίς να περιμένει την ολοκλήρωση της λειτουργίας. Μόλις ολοκληρωθεί η λειτουργία I/O, εκπέμπεται ένα συμβάν και εκτελείται η αντίστοιχη συνάρτηση ανάκλησης. Αυτή η προσέγγιση επιτρέπει στη Node.js να διαχειρίζεται μεγάλο αριθμό ταυτόχρονων συνδέσεων με ελάχιστη χρήση πόρων[5], [6].

Πολύ σημαντικό χαρακτηριστικό της Node.js είναι ότι διαθέτει έναν ενσωματωμένο διαχειριστή πακέτων που ονομάζεται NPM (Node Package Manager), ο οποίος επιτρέπει στους προγραμματιστές να εγκαθιστούν, να μοιράζονται και να διαχειρίζονται εύκολα πακέτα JavaScript ανοικτού κώδικα. Το NPM έχει γίνει ένα από τα μεγαλύτερα μητρώα πακέτων λογισμικού στον κόσμο, παρέχοντας πρόσβαση σε χιλιάδες επαναχρησιμοποιήσιμες ενότητες που μπορούν εύκολα να ενσωματωθούν σε μια εφαρμογή Node.js[7].

Χρησιμοποιώντας τα πακέτα NPM, οι προγραμματιστές μπορούν να δημιουργήσουν modular, συντηρήσιμες και επεκτάσιμες εφαρμογές. Κάθε πακέτο έχει σχεδιαστεί για να επιλύει ένα συγκεκριμένο πρόβλημα ή να παρέχει μια συγκεκριμένη λειτουργικότητα, επιτρέποντας στους προγραμματιστές να συναρμολογούν σύνθετες εφαρμογές από μικρά, επαναχρησιμοποιήσιμα δομικά στοιχεία.

Ένα από τα βασικά χαρακτηριστικά που διαφοροποιούν το Node.js από άλλες πλατφόρμες πλευράς διακομιστή είναι η αρχιτεκτονική του με γνώμονα τα συμβάντα. Σε αυτή την αρχιτεκτονική, ένας πομπός συμβάντων εκπέμπει συμβάντα που ενεργοποιούν λειτουργίες ακροατών όταν πληρούνται ορισμένες συνθήκες. Η Node.js διαθέτει ένα ενσωματωμένο module, το 'events', το οποίο διευκολύνει τη δημιουργία προσαρμοσμένων αρχιτεκτονικών με γνώμονα τα γεγονότα. Αυτό το module επιτρέπει στους προγραμματιστές να δημιουργούν εκπομπούς συμβάντων, να καταχωρούν ακροατές και να χειρίζονται συμβάντα με έναν εξαιρετικά προσαρμόσιμο τρόπο.

Ο προγραμματισμός με γνώμονα τα συμβάντα διευκολύνει τον χειρισμό ασύγχρονων λειτουργιών, όπως τα αιτήματα I/O, με την καταχώριση ανακλήσεων που εκτελούνται όταν συμβεί ένα συμβάν. Αυτή η προσέγγιση έχει ως αποτέλεσμα πιο ευέλικτες εφαρμογές, καθώς ο διακομιστής μπορεί να συνεχίσει την επεξεργασία άλλων εργασιών χωρίς να περιμένει την ολοκλήρωση μακροχρόνιων λειτουργιών.

Η Node.js έχει γίνει δημοφιλής επιλογή για ένα ευρύ φάσμα εφαρμογών λόγω της απόδοσης, της επεκτασιμότητας και της ευκολίας ανάπτυξης. Παρακάτω παρουσιάζονται ορισμένες από τις περιπτώσεις χρήσης στις οποίες το Node.js προτιμάται:

Εφαρμογές πραγματικού χρόνου: Η Node.js είναι κατάλληλη για εφαρμογές πραγματικού χρόνου, όπως διαδικτυακά παιχνίδια, εφαρμογές συνομιλίας και πλατφόρμες ζωντανής ροής. Η μη μπλοκαρισμένη I/O και η αρχιτεκτονική του που βασίζεται σε συμβάντα καθιστούν εύκολη τη διαχείριση μεγάλου αριθμού ταυτόχρονων συνδέσεων, εξασφαλίζοντας μια ομαλή και ευέλικτη εμπειρία χρήστη.

APIs και Microservices: Η Node.js είναι μια δημοφιλής επιλογή για τη δημιουργία APIs και microservices λόγω της ελαφριάς φύσης του και της ικανότητάς του να διαχειρίζεται μεγάλο όγκο αιτημάτων. Ο modular σχεδιασμός της, που υποστηρίζεται από το τεράστιο οικοσύστημα NPM, επιτρέπει στους προγραμματιστές να κατασκευάζουν και να συντηρούν σύνθετες εφαρμογές αναλύοντάς τες σε μικρότερες, πιο διαχειρίσιμες υπηρεσίες.

Αρχιτεκτονικές χωρίς διακομιστή: Η υπολογιστική χωρίς διακομιστή είναι μια αρχιτεκτονική προσέγγιση όπου ο πάροχος νέφους διαχειρίζεται την υποδομή διακομιστών, κλιμακώνοντας αυτόματα τους πόρους με βάση τις ανάγκες της εφαρμογής. Η Node.js είναι μια δημοφιλής επιλογή για εφαρμογές χωρίς διακομιστή λόγω της αποδοτικής χρήσης πόρων και της ικανότητάς της να διαχειρίζεται μεγάλο αριθμό ταυτόχρονων συνδέσεων.

Συσκευές Internet of Things (IoT): Αποτελεί κατάλληλη επιλογή για εφαρμογές IoT λόγω της ελαφριάς και αποδοτικής αρχιτεκτονικής του. Μπορεί να εκτελεστεί σε συσκευές με περιορισμένους πόρους και να χειριστεί πολλαπλές ταυτόχρονες συνδέσεις, γεγονός που το καθιστά εξαιρετική επιλογή για έργα IoT που περιλαμβάνουν συλλογή και επεξεργασία δεδομένων αισθητήρων.

Πολλές γνωστές εταιρείες έχουν υιοθετήσει τη Node.js για διάφορες περιπτώσεις χρήσης, αποδεικνύοντας την ευελιξία και τις δυνατότητές του. Παραδείγματος χάρη το Netflix τη χρησιμοποιεί για τον χειρισμό διεπαφών του χρήστη και τη βελτίωση της απόδοσης των εφαρμογών της. Η Paypal την υιοθέτησε για να βελτιώσει τη διαδικασία ανάπτυξης και να βελτιώσει την απόκριση και την απόδοση των διαδικτυακών εφαρμογών της. Το backend της εφαρμογής για κινητά της LinkedIn κατασκευάστηκε με τη χρήση Node.js, με αποτέλεσμα τη βελτίωση των επιδόσεων και της επεκτασιμότητας. Η Uber χρησιμοποιεί Node.js για το τεράστιο σύστημα αντιστοίχισης, επιτρέποντάς της να διαχειρίζεται μεγάλο όγκο αιτήσεων για διαδρομές και να εξασφαλίζει απρόσκοπτη εμπειρία χρήστη.

Το οικοσύστημα της Node.js τροφοδοτείται από μια ενεργή κοινότητα συνεργατών και προγραμματιστών. Από την ίδρυσή της έχει προσελκύσει μεγάλο αριθμό προγραμματιστών που έχουν δημιουργήσει πληθώρα πακέτων και βιβλιοθηκών ανοιχτού κώδικα, τα οποία είναι διαθέσιμα μέσω του μητρώου NPM. Αυτή η ενεργή κοινότητα όχι μόνο συνεισφέρει στον πυρήνα του Node.js αλλά παρέχει επίσης υποστήριξη, πόρους και εργαλεία που βελτιώνουν τη διαδικασία ανάπτυξης. Η κοινότητα της Node.js ασχολείται με τη συνεχή βελτίωση, αντιμετωπίζοντας ζητήματα, συμβάλλοντας σε συζητήσεις και βελτιώνοντας τα χαρακτηριστικά και τις επιδόσεις της πλατφόρμας.

Ενώ η Node.js προσφέρει πολλά πλεονεκτήματα για την ανάπτυξη ιστοσελίδων, έχει επίσης ορισμένους περιορισμούς και προκλήσεις:

Μοντέλο μονής ροής: Η αρχιτεκτονική ενός thread της Node.js μπορεί να είναι δίκοπο μαχαίρι. Παρόλο που επιτρέπει τον αποτελεσματικό χειρισμό ασύγχρονων λειτουργιών και μειώνει την επιβάρυνση από τη διαχείριση πολλαπλών thread, μπορεί επίσης να αποτελέσει περιορισμό για εργασίες που δεσμεύουν την ΚΜΕ. Για εφαρμογές που απαιτούν βαρείς υπολογισμούς, το μοντέλο με ένα thread μπορεί να γίνει σημείο συμφόρησης, καθώς μπορεί να μπλοκάρει άλλα εισερχόμενα αιτήματα.

Callback Hell: Σε προηγούμενες εκδόσεις της Node.js, η διαχείριση ασύγχρονου κώδικα μπορούσε να οδηγήσει σε μια κατάσταση γνωστή ως "callback hell" ή "pyramid of doom", όπου οι εμφωλευμένες κλήσεις καθιστούσαν τον κώδικα λιγότερο ευανάγνωστο και δυσκολότερο να συντηρηθεί. Ωστόσο, με την εισαγωγή των Promises, του async/await και άλλων προτύπων ασύγχρονου προγραμματισμού, το ζήτημα αυτό έχει σε μεγάλο βαθμό μετριαστεί.

Νέα και ασταθή modules: Το τεράστιο οικοσύστημα NPM παρέχει πληθώρα πακέτων ανοικτού κώδικα για διάφορους σκοπούς. Ωστόσο, ορισμένα από αυτά τα modules μπορεί να είναι σχετικά νέα ή ασταθης, οδηγώντας ενδεχομένως σε προβλήματα σε περιβάλλοντα παραγωγής. Οι προγραμματιστές πρέπει να αξιολογούν προσεκτικά και να επιλέγουν πακέτα που συντηρούνται καλά και έχουν σταθερό ιστορικό.

Το μέλλον του Node.js φαίνεται πολλά υποσχόμενο, καθώς η ενεργή κοινότητά του συνεχίζει να συμβάλλει στην ανάπτυξη και τη βελτίωσή του. Οι συνεχείς εξελίξεις στη JavaScript και στη μηχανή V8 θα οδηγήσουν πιθανότατα σε ακόμη καλύτερες επιδόσεις και χαρακτηριστικά για τη Node.js. Η αυξανόμενη δημοτικότητα του serverless computing και των αρχιτεκτονικών microservices την τοποθετεί ως έναν ισχυρό διεκδικητή στο σύγχρονο τοπίο του web development.

Καθώς το web development συνεχίζει να εξελίσσεται, η Node.js θα παραμείνει πιθανότατα ένα ζωτικής σημασίας εργαλείο για την κατασκευή κλιμακούμενων και αποδοτικών εφαρμογών ιστού. Η ενεργή κοινότητά του, το ισχυρό οικοσύστημα και η προσαρμόσιμη αρχιτεκτονική του το καθιστούν μια αξιόπιστη και ισχυρή επιλογή για τους προγραμματιστές παγκοσμίως.

Συμπερασματικά, η Node.js έφερε επανάσταση στην ανάπτυξη ιστοσελίδων εισάγοντας ένα μοντέλο I/O που καθοδηγείται από συμβάντα και δεν μπλοκάρει, το οποίο επιτρέπει την ανάπτυξη γρήγορων και κλιμακούμενων δικτυακών εφαρμογών. Η αποδοτική αρχιτεκτονική του, σε συνδυασμό με το τεράστιο οικοσύστημα NPM, το έχει καταστήσει δημοφιλή επιλογή για ένα ευρύ φάσμα εφαρμογών, από συστήματα πραγματικού χρόνου έως API και microservicces. Η υιοθέτηση της Node.js από μεγάλες εταιρείες αναδεικνύει την ευελιξία και τις δυνατότητές του, εδραιώνοντας τη θέση της ως ένα ισχυρό και απαραίτητο εργαλείο στη σύγχρονη ανάπτυξη ιστοσελίδων.

2.4 Express

Η Express.js, που συνήθως αναφέρεται ως Express, είναι ένα ελαφρύ, ευέλικτο και μινιμαλιστικό framework εφαρμογών ιστού για τη Node.js. Δημιουργήθηκε από τον TJ Holowaychuk το 2010 και έχει γίνει ένα από τα πιο δημοφιλή framework για τη δημιουργία εφαρμογών ιστού και APIs με τη Node.js. Η Express παρέχει ένα ισχυρό σύνολο χαρακτηριστικών που επιτρέπουν στους προγραμματιστές να δημιουργούν γρήγορα και εύκολα εφαρμογές ιστού και APIs, ενώ παράλληλα επιτρέπει εκτενες customization και επεκτασιμότητα.

Απλότητα και μινιμαλισμός: Η Express έχει σχεδιαστεί με γνώμονα την απλότητα και τον μινιμαλισμό. Παρέχει ένα λεπτό στρώμα χαρακτηριστικών πάνω από τη Node.js, επιτρέποντας στους προγραμματιστές να δημιουργούν γρήγορα εφαρμογές ιστού χωρίς την ανάγκη πολύπλοκων ρυθμίσεων. Η μινιμαλιστική προσέγγιση του framework σημαίνει ότι οι προγραμματιστές μπορούν εύκολα να προσθέσουν μόνο τα στοιχεία που χρειάζονται, με αποτέλεσμα να δημιουργούν ελαφριές και αποδοτικές εφαρμογές.

Middleware: Η βασική λειτουργικότητα της Express περιστρέφεται γύρω από την έννοια του middleware. Οι λειτουργίες middleware χρησιμοποιούνται για τον χειρισμό διαφόρων πτυχών του κύκλου αίτησης-απόκρισης, όπως η ανάλυση της αίτησης, ο έλεγχος ταυτότητας και η μορφοποίηση της απόκρισης. Οι προγραμματιστές μπορούν να χρησιμοποιήσουν ενσωματωμένο middleware, middleware τρίτων από το μητρώο NPM ή να δημιουργήσουν προσαρμοσμένες λειτουργίες middleware προσαρμοσμένες στις συγκεκριμένες ανάγκες τους.

Δρομολόγηση: Η Express παρέχει ισχυρά χαρακτηριστικά δρομολόγησης που επιτρέπουν στους προγραμματιστές να καθορίζουν τον τρόπο με τον οποίο η εφαρμογή θα πρέπει να ανταποκρίνεται στα αιτήματα του πελάτη με βάση τη ζητούμενη διεύθυνση URL και τη μέθοδο HTTP. Με τις δυνατότητες δρομολόγησης της, οι προγραμματιστές μπορούν να δημιουργούν RESTful APIs, να χειρίζονται δυναμικές παραμέτρους URL και να οργανώνουν τις διαδρομές της εφαρμογής τους με δομημένο και κλιμακούμενο τρόπο.

Απόδοση: Ο μινιμαλιστικός σχεδιασμός και η αποτελεσματική επεξεργασία ενδιάμεσου λογισμικού της Express την καθιστούν ένα framework υψηλών επιδόσεων. Δεδομένου ότι είναι χτισμένο πάνω στη Node.js, η Express κληρονομεί το μοντέλο I/O που βασίζεται σε συμβάντα και δεν μπλοκάρει, το οποίο του επιτρέπει να χειρίζεται μεγάλο αριθμό ταυτόχρονων συνδέσεων με ελάχιστη χρήση πόρων.

Επεκτασιμότητα: Η Express είναι εξαιρετικά επεκτάσιμη και προσαρμόσιμη. Οι προγραμματιστές μπορούν εύκολα να ενσωματώσουν βιβλιοθήκες και εργαλεία τρίτων, να δημιουργήσουν προσαρμοσμένο ενδιάμεσο λογισμικό ή να χρησιμοποιήσουν μηχανές προτύπων όπως οι Handlebars, Pug ή EJS για την απόδοση δυναμικού περιεχομένου HTML.

Μία από τις κύριες περιπτώσεις χρήσης της Express είναι η δημιουργία RESTful APIs (Διεπαφές Προγραμματισμού Εφαρμογών Μεταφοράς Αναπαραστατικής Κατάστασης). Τα RESTful APIs επιτρέπουν την επικοινωνία μεταξύ διαφορετικών τμημάτων μιας διαδικτυακής εφαρμογής ή μεταξύ διαφορετικών εφαρμογών συνολικά. Χρησιμοποιώντας μεθόδους HTTP (GET, POST, PUT, DELETE κλπ.), οι προγραμματιστές μπορούν να εκτελούν λειτουργίες CRUD (Create, Read, Update, Delete) σε πόρους που αναπαρίστανται από διευθύνσεις URL.

Τα middleware functions είναι μια θεμελιώδης έννοια στην Express. Πρόκειται για functions που έχουν πρόσβαση στο αντικείμενο αίτησης (req), στο αντικείμενο απόκρισης (res) και στο επόμενο middleware function στον κύκλο αίτησης-απόκρισης της εφαρμογής. Αυτά τα functions μπορούν να εκτελέσουν οποιονδήποτε κώδικα, να κάνουν αλλαγές στα αντικείμενα request και response, να τερματίσουν τον κύκλο request-response ή να καλέσουν την επόμενη συνάρτηση middleware στη στοίβα. Τα middleware functions χρησιμοποιούνται για διάφορους σκοπούς, όπως η καταγραφή, ο έλεγχος ταυτότητας, ο χειρισμός σφαλμάτων ή η ανάλυση request bodies.

Η Express υποστηρίζει ένα ευρύ φάσμα από template engines που μπορούν να χρησιμοποιηθούν για την απόδοση δυναμικού περιεχομένου HTML στην πλευρά του διακομιστή. Τα template engines επιτρέπουν στους προγραμματιστές να ορίζουν HTML templates με placeholders για δυναμικό περιεχόμενο, τα οποία μπορούν να αντικατασταθούν με πραγματικά δεδομένα κατά το χρόνο εκτέλεσης. Ορισμένα δημοφιλείς template engine που υποστηρίζονται από την Express περιλαμβάνουν το Pug (παλαιότερα γνωστό ως Jade), το Handlebars και το EJS (Embedded JavaScript).

Η Express έχει γίνει το de facto framework για την κατασκευή εφαρμογών ιστού και APIs με τη Node.js. Η ευελιξία, η επεκτασιμότητα και η ευκολία χρήσης της τη καθιστούν μια δημοφιλή επιλογή μεταξύ των προγραμματιστών. Πολλές γνωστές εταιρείες και πλατφόρμες χρησιμοποιούν την Express στα τεχνολογικά τους stack, όπως η IBM, η Uber και η Medium.

Η Express είναι επίσης μια δημοφιλής επιλογή για τη δημιουργία microservices. Τα microservices είναι μικρές, ανεξάρτητες υπηρεσίες που συνεργάζονται για να σχηματίσουν μια μεγαλύτερη εφαρμογή. Κάθε microservice μπορεί να αναπτυχθεί και να κλιμακωθεί ανεξάρτητα. Με τη χρήση της Express, οι προγραμματιστές μπορούν εύκολα να δημιουργήσουν και να διαχειριστούν μικρουπηρεσίες σε περιβάλλον Node.js.

Συμπερασματικά, η Express.js είναι ένα ευέλικτο και ισχυρό framework εφαρμογών ιστού για τη Node.js. Παρέχει μια απλή και μινιμαλιστική προσέγγιση για τη δημιουργία εφαρμογών ιστού και API, καθιστώντας το μια δημοφιλή επιλογή μεταξύ των προγραμματιστών. Με χαρακτηριστικά όπως το middleware, τη δρομολόγηση και την υποστήριξη για διάφορα template engines, η Express επιτρέπει στους προγραμματιστές να δημιουργούν γρήγορα κλιμακούμενες και αποδοτικές εφαρμογές ιστού. Η επεκτασιμότητα και η συμβατότητα της Express με το οικοσύστημα Node.js το καθιστούν

εξαιρετική επιλογή για τη σύγχρονη ανάπτυξη ιστού. Η ενεργή κοινότητά του και η συνεχής ανάπτυξή του διασφαλίζουν ότι θα παραμείνει ένα απαραίτητο εργαλείο για τους προγραμματιστές και τα επόμενα χρόνια[8].

2.5 MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS) ανοικτού κώδικα που χρησιμοποιεί τη Δομημένη Γλώσσα Ερωτήσεων (SQL) για τη διαχείριση και τον χειρισμό δεδομένων. Από την ίδρυσή του το 1995 από μια σουηδική εταιρεία που ονομάζεται MySQL AB, έχει γίνει ένα από τα πιο δημοφιλή και ευρέως χρησιμοποιούμενα συστήματα βάσεων δεδομένων στον κόσμο. Το όνομα MySQL είναι ένας συνδυασμός του "My", του ονόματος της κόρης του συνιδρυτή Michael Widenius, και του "SQL", της συντομογραφίας της Structured Query Language.

Ένα σχεσιακό σύστημα βάσεων δεδομένων είναι ένας τύπος συστήματος βάσεων δεδομένων που αποθηκεύει δεδομένα σε πίνακες με προκαθορισμένες στήλες και γραμμές. Κάθε γραμμή σε έναν πίνακα αντιπροσωπεύει μια μοναδική εγγραφή, ενώ κάθε στήλη αντιπροσωπεύει ένα χαρακτηριστικό της εγγραφής. Οι σχέσεις μεταξύ των πινάκων μπορούν να δημιουργηθούν μέσω της χρήσης πρωτεύοντος και ξένου κλειδιού. Η γλώσσα SQL χρησιμοποιείται για τη δημιουργία, την τροποποίηση και την ανάκτηση δεδομένων σε σχεσιακές βάσεις δεδομένων.

Η MySQL φημίζεται για τις επιδόσεις, την αξιοπιστία και την ευκολία χρήσης της. Μερικά από τα αξιοσημείωτα χαρακτηριστικά της MySQL περιλαμβάνουν:

Συμμόρφωση ACID: Η MySQL υποστηρίζει τις ιδιότητες ACID (Atomicity, Consistency, Isolation, Durability), οι οποίες είναι απαραίτητες για την αξιόπιστη επεξεργασία συναλλαγών. Οι συναλλαγές είναι ατομικές, που σημαίνει ότι αντιμετωπίζονται ως ενιαία μονάδα. Όλες οι λειτουργίες εντός μιας συναλλαγής είτε ολοκληρώνονται επιτυχώς είτε ανατρέπονται πλήρως.

Αντιγραφή: Η MySQL υποστηρίζει την αντιγραφή master-slave, η οποία επιτρέπει την αντιγραφή δεδομένων από έναν διακομιστή (master) σε έναν άλλο (slave). Η αντιγραφή μπορεί να χρησιμοποιηθεί για εξισορρόπηση φορτίου, δημιουργία αντιγράφων ασφαλείας και κλιμάκωση των λειτουργιών ανάγνωσης σε πολλούς διακομιστές.

Κατάτμηση: Η MySQL υποστηρίζει κατάτμηση πινάκων, η οποία διαιρεί έναν πίνακα σε μικρότερα, πιο διαχειρίσιμα κομμάτια που ονομάζονται κατατμήσεις. Η κατάτμηση μπορεί να βελτιώσει την απόδοση και να απλοποιήσει τη διαχείριση μεγάλων πινάκων.

Μηχανή αποθήκευσης InnoDB: Χρησιμοποιείται από προεπιλογή η μηχανή αποθήκευσης InnoDB, η οποία υποστηρίζει συναλλαγές συμβατές με ACID, ξένα κλειδιά και αναζήτηση πλήρους κειμένου. Η InnoDB διαθέτει επίσης δυνατότητες αποκατάστασης από crash και υποστήριξη για online backup.

Ασφάλεια: Η MySQL παρέχει ισχυρά χαρακτηριστικά ασφαλείας, όπως κρυπτογράφηση δεδομένων, ασφαλείς συνδέσεις και έλεγχο πρόσβασης βάσει ρόλων. Υποστηρίζει επίσης πρόσθετα για έλεγχο ταυτότητας, λογιστικό έλεγχο και επικύρωση κωδικού πρόσβασης.

Υποστήριξη πολλαπλών πλατφορμών: Μπορεί να τρέξει σε διάφορα λειτουργικά συστήματα, συμπεριλαμβανομένων των Windows, macOS, Linux και Unix.

Επεκτασιμότητα: Η MySQL μπορεί να διαχειριστεί μεγάλες ποσότητες δεδομένων και ταυτόχρονους χρήστες. Έχει σχεδιαστεί για να είναι επεκτάσιμη τόσο κάθετα (προσθέτοντας περισσότερους πόρους σε έναν μόνο διακομιστή) όσο και οριζόντια (προσθέτοντας περισσότερους διακομιστές).

Υποστήριξη για SQL και NoSQL: Η MySQL υποστηρίζει τόσο μοντέλα δεδομένων SQL όσο και NoSQL. Παρέχει έναν τύπο δεδομένων JSON για την αποθήκευση και την αναζήτηση εγγράφων JSON, καθιστώντας την κατάλληλη για εφαρμογές που απαιτούν τόσο δομημένα όσο και ημιδομημένα δεδομένα.

Η MySQL χρησιμοποιείται σε ένα ευρύ φάσμα εφαρμογών, όπως ανάπτυξη ιστοσελίδων, αποθήκευση δεδομένων, ηλεκτρονικό εμπόριο, συστήματα διαχείρισης περιεχομένου και άλλα. Αποτελεί δημοφιλή επιλογή για εφαρμογές ιστού λόγω των επιδόσεων, της ευκολίας χρήσης και της συμβατότητάς της με δημοφιλείς γλώσσες ανάπτυξης ιστού, όπως η PHP, η Java, η Javascript και η Python. Χρησιμοποιείται επίσης από πολλές μεγάλες εταιρείες, όπως η Google, το Facebook και το Twitter, για τη διαχείριση τεράστιου όγκου δεδομένων. Η MySQL διατίθεται τόσο σε κοινοτικές όσο και σε εμπορικές εκδόσεις. Η κοινοτική έκδοση είναι δωρεάν και ανοικτού κώδικα, ενώ η εμπορική έκδοση, γνωστή ως MySQL Enterprise Edition, προσφέρει πρόσθετες δυνατότητες και υποστήριξη έναντι αμοιβής. Η κοινότητα της MySQL είναι ενεργή και παρέχει πόρους, τεκμηρίωση και υποστήριξη για τους χρήστες. Η αρχιτεκτονική της είναι ένα πολυεπίπεδο σύστημα, το οποίο αποτελείται από διάφορα στοιχεία που συνεργάζονται για την επεξεργασία, την αποθήκευση και την ανάκτηση δεδομένων. Ορισμένα από τα βασικά συστατικά στοιχεία της αρχιτεκτονικής της MySQL είναι τα εξής:

Μοντέλο πελάτη-εξυπηρετητή: Η MySQL χρησιμοποιεί ένα μοντέλο πελάτη-εξυπηρετητή, όπου ο διακομιστής διαχειρίζεται τη βάση δεδομένων και επεξεργάζεται τα ερωτήματα SQL, ενώ οι πελάτες συνδέονται στον διακομιστή για να ζητήσουν και να ανακτήσουν δεδομένα. Οι πελάτες μπορεί να είναι εφαρμογές, εργαλεία γραμμής εντολών ή άλλα συστήματα διαχείρισης βάσεων δεδομένων.

Διαχείριση συνδέσεων: Η MySQL διαχειρίζεται τις συνδέσεις πελατών μέσω ενός επιπέδου διαχείρισης συνδέσεων και ελέγχου ταυτότητας. Όταν ένας πελάτης συνδέεται στο διακομιστή MySQL, δημιουργείται ένα νήμα σύνδεσης που διαχειρίζεται τον έλεγχο ταυτότητας, την ασφάλεια και την επικοινωνία με τον πελάτη.

Διεπαφή SQL: Το στρώμα διεπαφής SQL λαμβάνει ερωτήματα SQL από τους πελάτες, αναλύει τα ερωτήματα και δημιουργεί ένα σχέδιο εκτέλεσης. Βελτιστοποιεί το σχέδιο εκτέλεσης για απόδοση και το αποστέλλει στη μηχανή αποθήκευσης για εκτέλεση.

Μηχανή αποθήκευσης: Η μηχανή αποθήκευσης είναι υπεύθυνη για τη διαχείριση της φυσικής αποθήκευσης και ανάκτησης δεδομένων. Η MySQL υποστηρίζει πολλαπλές μηχανές αποθήκευσης, καθεμία με τα δικά της χαρακτηριστικά και δυνατότητες. Η InnoDB είναι η προεπιλεγμένη μηχανή αποθήκευσης, αλλά υπάρχουν και άλλες, όπως η MyISAM, η MEMORY και η BLACKHOLE.

Ενεργοποιήσιμες μηχανές αποθήκευσης: Η pluggable αρχιτεκτονική μηχανών αποθήκευσης της MySQL επιτρέπει στους χρήστες να επιλέξουν την καταλληλότερη μηχανή αποθήκευσης για τις συγκεκριμένες ανάγκες τους. Αυτή η ευελιξία επιτρέπει στη MySQL να χειρίζεται διαφορετικούς τύπους φόρτων εργασίας και να βελτιστοποιεί τις επιδόσεις.

Η MySQL χρησιμοποιείται ευρέως για διάφορους σκοπούς, όπως:

Εφαρμογές ιστού: Η MySQL είναι μια δημοφιλής επιλογή για εφαρμογές ιστού, συμπεριλαμβανομένων των πλατφορμών ηλεκτρονικού εμπορίου, των συστημάτων διαχείρισης περιεχομένου και των ιστότοπων κοινωνικής δικτύωσης. Χρησιμοποιείται συχνά σε συνδυασμό με γλώσσες ανάπτυξης ιστού, όπως η PHP, η Java και η Python.

Data Warehousing: Μπορεί να χρησιμοποιηθεί για την αποθήκευση μεγάλου όγκου δεδομένων για data warehousing, ανάλυση και υποβολή εκθέσεων. Επίσης μπορεί να διαχειριστεί τεράστιους όγκους δεδομένων και παρέχει εργαλεία για μετασχηματισμό, συγκέντρωση και ανάλυση δεδομένων.

Ενσωματωμένες βάσεις δεδομένων: Είναι κατάλληλη για ενσωματωμένες βάσεις δεδομένων σε επιτραπέζιες εφαρμογές, εφαρμογές για κινητά και συσκευές IoT. Προσφέρει μια ελαφριά και ευέλικτη λύση για τοπική αποθήκευση και διαχείριση δεδομένων.

Λύσεις υψηλής διαθεσιμότητας: Η MySQL μπορεί να χρησιμοποιηθεί για τη δημιουργία λύσεων υψηλής διαθεσιμότητας και failover για κρίσιμες εφαρμογές. Υποστηρίζει αντιγραφή, ομαδοποίηση και εξισορρόπηση φορτίου, ώστε να διασφαλίζεται η αδιάλειπτη πρόσβαση στα δεδομένα.

Η ρύθμιση επιδόσεων στη MySQL περιλαμβάνει τη βελτιστοποίηση διαφόρων παραγόντων για τη βελτίωση της ταχύτητας και της αποδοτικότητας των λειτουργιών της βάσης δεδομένων. Ορισμένες από τις βασικές συμβουλές ρύθμισης των επιδόσεων για τη MySQL περιλαμβάνουν:

Βελτιστοποίηση ερωτημάτων: Με την χρήση της εντολής EXPLAIN υπάρχει δυνατότητα ανάλυσης των σχεδίων εκτέλεσης ερωτημάτων και για τον εντοπισμό σημείων συμφόρησης. Επανεγγραφή ή αναδιαμόρφωση των αναποτελεσματικών ερωτημάτων για βελτίωση της απόδοσης.

Χρήση δεικτών: Δημιουργία ευρετηρίων σε στήλες που ζητούνται συχνά για την επιτάχυνση της ανάκτηση δεδομένων.Η υπερβολική ευρετηρίαση είναι μέθοδος προς αποφυγή, καθώς μπορεί να επιβραδύνει τις λειτουργίες εισαγωγής και ενημέρωσης.

Συντονισμός παραμέτρων διακομιστή: Προσαρμογή των παραμέτρων του διακομιστή MySQL, όπως τα μεγέθη ρυθμιστικού διαύλου, τα μεγέθη προσωρινής μνήμης και τα όρια σύνδεσης, για την βελτιστοποίηση της απόδοσης. Παρακολούθηση των μετρήσεων απόδοσης του διακομιστή για την εντόπιση σημείων που χρειάζονται βελτίωση.

Βελτιστοποίηση μηχανών αποθήκευσης: Επιλογή σωστών μηχανών αποθήκευσης για συγκεκριμένες περιπτώσεις χρήσης. Συντονισμός των ρυθμίσεων της μηχανής αποθήκευσης, όπως το μέγεθος της δεξαμενής απομονωτών InnoDB και το μέγεθος του απομονωτή κλειδιών MyISAM, για την βελτιστοποίηση στην πρόσβαση των δεδομένων.

Τακτική συντήρηση: Εκτέλεση τακτικών εργασιών συντήρησης, όπως βελτιστοποίηση πινάκων, ενημέρωση στατιστικών στοιχείων και ανασυγκρότηση του συστήματος αρχείων, για την διατήρηση της ομαλής λειτουργία της βάσης δεδομένων.

Παρακολούθηση επιδόσεων: Χρήση των εργαλείων παρακολούθησης όπως το MySQL Performance Schema, το MySQL Enterprise Monitor και των λύσεων παρακολούθησης τρίτων για την παρακολούθηση των μετρήσεων απόδοσης και τον εντοπισμό προβλημάτων.

Η MySQL αντιμετωπίζει ανταγωνισμό από διάφορα άλλα συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων. Μερικοί από τους βασικούς ανταγωνιστές είναι οι εξής:

PostgreSQL: Ένα RDBMS ανοικτού κώδικα που προσφέρει χαρακτηριστικά όπως η κληρονομικότητα πινάκων, η υποστήριξη πινάκων και η προηγμένη ευρετηρίαση. Η PostgreSQL είναι γνωστή για την ευελιξία της, την επεκτασιμοτητα της και την τήρηση των προτύπων SQL.

Βάση δεδομένων Oracle: Η Oracle Database είναι γνωστή για τα ισχυρά χαρακτηριστικά της, τις υψηλές επιδόσεις και την επεκτασιμότητά της. Χρησιμοποιείται συνήθως σε μεγάλες επιχειρήσεις και κρίσιμες εφαρμογές.

Microsoft SQL Server: Ένα εμπορικό RDBMS που αναπτύχθηκε από τη Microsoft. Ο SQL Server είναι δημοφιλής στους χρήστες των Windows και προσφέρει χαρακτηριστικά όπως η ενσωμάτωση με τα εργαλεία BI της Microsoft, ισχυρές δυνατότητες καταγραφής και προηγμένες αναλύσεις.

IBM Db2: Ένα εμπορικό RDBMS που αναπτύχθηκε από την IBM. Το Db2 είναι γνωστό για τις υψηλές επιδόσεις, την αξιοπιστία και την υποστήριξη προηγμένων αναλύσεων και μηχανικής μάθησης.

MariaDB: Ένα RDBMS ανοικτού κώδικα που δημιουργήθηκε ως διακλάδωση της MySQL από τους αρχικούς προγραμματιστές της MySQL. Η MariaDB έχει ως στόχο να διατηρήσει τη συμβατότητα με τη MySQL, προσθέτοντας παράλληλα νέα χαρακτηριστικά και βελτιώσεις.

Η MySQL είναι ένας εξέχων παίκτης στην αγορά RDBMS εδώ και δεκαετίες και η δημοτικότητά της είναι πιθανό να συνεχιστεί. Ορισμένες τάσεις που μπορεί να διαμορφώσουν το μέλλον της MySQL είναι οι εξής:

Ενσωμάτωση του νέφους: Με την αυξανόμενη στροφή προς το cloud computing, η ενσωμάτωση της MySQL με υπηρεσίες και παρόχους cloud θα γίνει πιο κρίσιμη. Η MySQL προσφέρει ήδη υπηρεσίες που βασίζονται στο cloud, όπως το MySQL Database Service και το MySQL Cluster, οι οποίες θα γίνουν πιο σημαντικές καθώς όλο και περισσότεροι οργανισμοί θα υιοθετούν το cloud computing.

Τεχνητή νοημοσύνη και μηχανική μάθηση: Καθώς η τεχνητή νοημοσύνη και η μηχανική μάθηση γίνονται όλο και πιο διαδεδομένες, η MySQL θα πρέπει να ενσωματωθεί με εργαλεία και πλαίσια AI/ML, παρέχοντας υποστήριξη για προηγμένες αναλύσεις, αναγνώριση προτύπων και προγνωστική μοντελοποίηση.

IoT και Big Data: Η άνοδος του Διαδικτύου των Πραγμάτων (IoT) και των Big Data θα απαιτήσει από τη MySQL να διαχειρίζεται τεράστιες ποσότητες δεδομένων που παράγονται από συνδεδεμένες συσκευές. Η MySQL θα πρέπει να βελτιστοποιήσει τις επιδόσεις, την επεκτασιμότητα και την αποθήκευση για να ανταποκριθεί στις απαιτήσεις των εφαρμογών IoT και μεγάλων δεδομένων.

Συμπερασματικά, η MySQL είναι ένα ευέλικτο και ισχυρό RDBMS που βρίσκεται στην πρώτη γραμμή του τοπίου διαχείρισης βάσεων δεδομένων εδώ και πολλά χρόνια. Οι εκτεταμένες δυνατότητες, οι επιδόσεις, η αξιοπιστία και η ευκολία χρήσης της την καθιστούν προτιμώμενη επιλογή για ένα ευρύ φάσμα εφαρμογών. Ο χαρακτήρας ανοικτού κώδικα και η ενεργή κοινότητα της MySQL συμβάλλουν περαιτέρω στη δημοτικότητά της. Καθώς η τεχνολογία εξελίσσεται και αναδύονται νέες τάσεις, η MySQL θα συνεχίσει να προσαρμόζεται και να καινοτομεί για να ανταποκρίνεται στις μεταβαλλόμενες ανάγκες της αγοράς βάσεων δεδομένων[9], [10].

2.6 Javascript

Η JavaScript είναι μια υψηλού επιπέδου, διερμηνευμένη και αντικειμενοστραφής γλώσσα προγραμματισμού που χρησιμοποιείται κυρίως για την ανάπτυξη ιστοσελίδων. Αρχικά αναπτύχθηκε από τη Netscape ως ένας τρόπος δημιουργίας διαδραστικών ιστοσελίδων, έκτοτε έχει γίνει θεμελιώδες μέρος του σύγχρονου ιστού και υποστηρίζεται από όλους τους μεγάλους browsers. Η ευελιξία της JavaScript, η ευκολία χρήσης και το ευρύ φάσμα χαρακτηριστικών της την καθιστούν τη γλώσσα που επιλέγουν οι προγραμματιστές για τη δημιουργία δυναμικών και διαδραστικών ιστοσελίδων. Δημιουργήθηκε το 1995 από τον Brendan Eich, έναν μηχανικό της Netscape, ο οποίος ανέλαβε να δημιουργήσει μια γλώσσα σεναρίων για τον ιστό που θα έκανε τις ιστοσελίδες πιο δυναμικές και

διαδραστικές. Η γλώσσα ονομάστηκε αρχικά Mocha, μετονομάστηκε σε LiveScript και τελικά ονομάστηκε JavaScript σε μια συμφωνία συνεργασίας μεταξύ της Netscape και της Sun Microsystems, οι οποίες προωθούσαν τη γλώσσα Java εκείνη την εποχή. Παρά τις ομοιότητες στα ονόματά τους, η Java και η JavaScript είναι εντελώς ξεχωριστές γλώσσες με διαφορετική σύνταξη και χρήση. Κυκλοφόρησε αρχικά ως μέρος του Netscape Navigator 2.0, του πρώτου προγράμματος περιήγησης που υποστήριζε τη γλώσσα[11], [12]. Γρήγορα απέκτησε δημοτικότητα μεταξύ των προγραμματιστών ιστού και, το 1997, τυποποιήθηκε από την Ευρωπαϊκή Ένωση Κατασκευαστών Υπολογιστών (ECMA) ως ECMAScript[13]. Έκτοτε, η JavaScript συνέχισε να εξελίσσεται, με τις επόμενες εκδόσεις της ECMAScript να προσθέτουν νέα χαρακτηριστικά και βελτιώσεις.

Η δημοτικότητα της JavaScript μπορεί να αποδοθεί στα βασικά χαρακτηριστικά της που επιτρέπουν στους προγραμματιστές να δημιουργούν ιδιαίτερα διαδραστικούς και δυναμικούς ιστότοπους. Ορισμένα από αυτά τα χαρακτηριστικά περιλαμβάνουν:

Αλληλεπιδραστικότητα: Η JavaScript επιτρέπει στους προγραμματιστές να δημιουργούν διαδραστικά στοιχεία σε μια ιστοσελίδα, όπως κουμπιά, φόρμες και κινούμενες εικόνες, τα οποία μπορούν να ανταποκρίνονται στην είσοδο του χρήστη. Αυτή η δυνατότητα βελτιώνει την εμπειρία του χρήστη και επιτρέπει στους προγραμματιστές να δημιουργούν πιο ελκυστικές και φιλικές προς τον χρήστη ιστοσελίδες.

Ασύγχρονος προγραμματισμός: Με χαρακτηριστικά όπως τα callbacks, τα promises και το async/await, η JavaScript επιτρέπει τον ασύγχρονο προγραμματισμό, επιτρέποντας την ταυτόχρονη πραγματοποίηση πολλαπλών λειτουργιών χωρίς να περιμένουν να τελειώσει η μία πριν ξεκινήσουν οι επόμενες. Αυτό έχει ως αποτέλεσμα ταχύτερες και αποδοτικότερες ιστοσελίδες, καθώς οι προγραμματιστές μπορούν να βελτιστοποιήσουν τη φόρτωση των πόρων και την εκτέλεση των εργασιών.

Χειρισμός DOM: Επιτρέπει στους προγραμματιστές να ενημερώνουν δυναμικά το περιεχόμενο, τη δομή και το στυλ μιας ιστοσελίδας χωρίς να απαιτείται επαναφόρτωση της σελίδας. Αυτό το χαρακτηριστικό είναι ζωτικής σημασίας για τη δημιουργία ευέλικτων εφαρμογών ιστού που μπορούν να προσαρμόζονται στις εισροές των χρηστών και στις μεταβαλλόμενες συνθήκες[14].

Αντικειμενοστραφής προγραμματισμός (OOP): Η JavaScript υποστηρίζει αντικειμενοστραφή προγραμματισμό, ένα παράδειγμα που επιτρέπει στους προγραμματιστές να μοντελοποιούν αντικείμενα του πραγματικού κόσμου και τις αλληλεπιδράσεις τους. Με τη χρήση κλάσεων, αντικειμένων και κληρονομικότητας, οι προγραμματιστές μπορούν να δημιουργήσουν αρθρωτό και επαναχρησιμοποιήσιμο κώδικα που είναι ευκολότερο να συντηρηθεί και να επεκταθεί.

Ευρύ οικοσύστημα: Η JavaScript διαθέτει ένα ευρύ οικοσύστημα βιβλιοθηκών, framework και εργαλείων που διευκολύνουν τους προγραμματιστές να δημιουργούν σύνθετες εφαρμογές ιστού. Δημοφιλείς βιβλιοθήκες και frameworks όπως το jQuery, η React, η Vue.js και η Angular έχουν απλοποιήσει τις κοινές εργασίες και έχουν βοηθήσει στην ταχεία ανάπτυξη εφαρμογών ιστού.

Με την πάροδο του χρόνου, η JavaScript έχει γίνει απαραίτητο εργαλείο για τους προγραμματιστές ιστού. Η γλώσσα έχει εξελιχθεί από απλά σενάρια για την προσθήκη διαδραστικότητας σε ιστοσελίδες σε μια ισχυρή γλώσσα που μπορεί να υποστηρίξει σύνθετες εφαρμογές ιστού.

Front-end Development: Είναι υπεύθυνη για τη δημιουργία της διεπαφής χρήστη και της εμπειρίας του χρήστη μιας ιστοσελίδας. Οι προγραμματιστές χρησιμοποιούν τη JavaScript για να χειρίζονται τις αλληλεπιδράσεις των χρηστών, να δημιουργούν animations και να παρέχουν δυναμικό περιεχόμενο. Δημοφιλή front-end frameworks όπως η React, η Angular και η Vue.js έχουν δώσει τη δυνατότητα

στους προγραμματιστές να δημιουργούν πιο αποτελεσματικά πολύπλοκες και ευέλικτες διεπαφές χρήστη.

Back-end Development: Με την έλευση του Node.js, ενός περιβάλλοντος εκτέλεσης που επιτρέπει στους προγραμματιστές να εκτελούν JavaScript στην πλευρά του διακομιστή, η JavaScript έχει γίνει επίσης μια δημοφιλής γλώσσα για την ανάπτυξη ιστοσελίδων back-end. Η Node.js χρησιμοποιεί ένα μοντέλο εισόδου/εξόδου που καθοδηγείται από συμβάντα και δεν μπλοκάρει, καθιστώντας το ιδιαίτερα αποδοτικό για το χειρισμό πολλαπλών ταυτόχρονων συνδέσεων. Οι προγραμματιστές μπορούν πλέον να χρησιμοποιούν την ίδια γλώσσα τόσο για front-end όσο και για back-end development, βελτιώνοντας τη διαδικασία ανάπτυξης.

Full-stack Development: Συνδυάζοντας τόσο την ανάπτυξη front-end όσο και την ανάπτυξη back-end, η JavaScript χρησιμοποιείται στο full-stack development για τη δημιουργία ολόκληρων εφαρμογών ιστού. Οι προγραμματιστές που χρησιμοποιούν το MEAN (MongoDB, Express.js, Angular, Node.js) ή το MERN (MongoDB, Express.js, React, Node.js) stack μπορούν να αξιοποιήσουν τη JavaScript σε ολόκληρη τη διαδικασία ανάπτυξης, διευκολύνοντας τη δημιουργία συνεκτικών και καλά ενσωματωμένων εφαρμογών.

Συμβατότητα πολλαπλών πλατφορμών: JavaScript υποστηρίζεται από όλα τα μεγάλα προγράμματα περιήγησης, καθιστώντας την μια καθολική γλώσσα για την ανάπτυξη ιστοσελίδων. Οι προγραμματιστές μπορούν να γράψουν κώδικα μία φορά και να τον εκτελέσουν σε πολλαπλές πλατφόρμες, μειώνοντας την προσπάθεια που απαιτείται για τη δημιουργία εφαρμογών ιστού πολλαπλών πλατφορμών.

Υποστήριξη από την κοινότητα: Με την ευρεία υιοθέτησή της, η JavaScript διαθέτει μια μεγάλη και ενεργή κοινότητα προγραμματιστών που συμβάλλουν στο οικοσύστημα της γλώσσας. Υπάρχουν αμέτρητοι πόροι, σεμινάρια και φόρουμ όπου οι προγραμματιστές μπορούν να αναζητήσουν βοήθεια, να μοιραστούν γνώσεις και να συνεργαστούν σε έργα.

Απόδοση: Η JavaScript είναι μια διερμηνευμένη γλώσσα, πράγμα που σημαίνει ότι μπορεί να εκτελεστεί απευθείας από το πρόγραμμα περιήγησης χωρίς την ανάγκη μεταγλώττισης. Αυτό την καθιστά ταχύτερη από ορισμένες άλλες γλώσσες και επιτρέπει στους προγραμματιστές να βλέπουν τα άμεσα αποτελέσματα των αλλαγών στον κώδικά τους.

Ευελιξία: Είναι μια ευέλικτη γλώσσα που μπορεί να χρησιμοποιηθεί για ένα ευρύ φάσμα εφαρμογών. Οι προγραμματιστές μπορούν να τη χρησιμοποιήσουν για απλές εργασίες όπως η επικύρωση φορμών ή για σύνθετες εφαρμογές όπως τα διαδικτυακά παιχνίδια και η επεξεργασία δεδομένων σε πραγματικό χρόνο.

Παρά τα πολλά πλεονεκτήματά της, η JavaScript δεν είναι απαλλαγμένη από τις προκλήσεις και τις επικρίσεις της. Ορισμένες από τις συνήθεις επικρίσεις περιλαμβάνουν:

Ζητήματα ασφάλειας: Η JavaScript εκτελείται στην πλευρά του πελάτη, γεγονός που την καθιστά ευάλωτη σε ευπάθειες ασφαλείας, όπως επιθέσεις cross-site scripting (XSS). Οι προγραμματιστές πρέπει να επαγρυπνούν για να διασφαλίσουν ότι ο κώδικάς τους είναι ασφαλής και ακολουθεί τις βέλτιστες πρακτικές[15].

Ασυνέπειες απόδοσης: Λόγω των διαφορών στις υλοποιήσεις των προγραμμάτων περιήγησης, η JavaScript μπορεί μερικές φορές να συμπεριφέρεται με ασυνέπεια σε διαφορετικά προγράμματα περιήγησης. Οι προγραμματιστές πρέπει συχνά να γράφουν πρόσθετο κώδικα ή να χρησιμοποιούν polyfills για να εξασφαλίσουν συνεπή συμπεριφορά.

Καμπύλη εκμάθησης: Ενώ η JavaScript είναι σχετικά εύκολη για να ξεκινήσει κάποιος, έχει μια απότομη καμπύλη εκμάθησης καθώς οι προγραμματιστές εμβαθύνουν σε πιο προηγμένα θέματα όπως ο ασύγχρονος προγραμματισμός, τα κλεισίματα και η πρωτοτυπική κληρονομικότητα.

Η άνοδος της JavaScript στην κορυφή έχει επηρεάσει σημαντικά τον κλάδο της τεχνολογίας. Η συμβολή της εκτείνεται πέρα από την ανάπτυξη ιστού και έχει ανοίξει το δρόμο για διάφορες τεχνολογικές εξελίξεις.

Ανάπτυξη κινητών συσκευών: Frameworks στη JavaScript όπως το React Native και το PhoneGap έχουν δώσει τη δυνατότητα στους προγραμματιστές να δημιουργούν εφαρμογές για κινητά χρησιμοποιώντας την ίδια βάση κώδικα τόσο για πλατφόρμες iOS όσο και για Android. Αυτό έχει φέρει επανάσταση στην ανάπτυξη κινητών συσκευών μειώνοντας τον χρόνο, την προσπάθεια και το κόστος ανάπτυξης.

Αρχιτεκτονική χωρίς διακομιστή: Η JavaScript και η event-driven φύση της την έχει καταστήσει σε μια δημοφιλή επιλογή για την υπολογιστική χωρίς διακομιστή. Οι προγραμματιστές μπορούν να δημιουργήσουν εφαρμογές που αποτελούνται από microservices χωρίς να διαχειρίζονται την υποκείμενη υποδομή, με αποτέλεσμα την εξοικονόμηση κόστους, την επεκτασιμότητα και την ταχύτερη διάθεση στην αγορά.

Διαδίκτυο των πραγμάτων (IoT): Η ελαφριά και ασύγχρονη φύση της την έχει καταστήσει κατάλληλη για συσκευές IoT με περιορισμένους υπολογιστικούς πόρους. Το Node.js, ειδικότερα, χρησιμοποιείται ευρέως για την κατασκευή εφαρμογών IoT λόγω του μη μπλοκαρίσματος εισόδου/εξόδου και της υποστήριξης επεξεργασίας δεδομένων σε πραγματικό χρόνο.

Web Assembly (WASM): Η κυριαρχία της JavaScript στην ανάπτυξη ιστοσελίδων οδήγησε στην ανάπτυξη του WebAssembly, μιας δυαδικής μορφής εντολών που επιτρέπει την εκτέλεση κώδικα υψηλής απόδοσης σε προγράμματα περιήγησης στο διαδίκτυο. Αυτό επιτρέπει στους προγραμματιστές να εκτελούν κώδικα γραμμένο σε γλώσσες όπως η C, η C++ και η Rust στον ιστό, παρέχοντας μια εναλλακτική λύση στη JavaScript για υπολογιστικά απαιτητικές εργασίες.

Μηχανική μάθηση και τεχνητή νοημοσύνη: Βιβλιοθήκες JavaScript όπως οι TensorFlow.js και Brain.js έχουν φέρει τη μηχανική μάθηση και την τεχνητή νοημοσύνη στο πρόγραμμα περιήγησης. Οι προγραμματιστές μπορούν πλέον να δημιουργούν και να αναπτύσσουν μοντέλα μηχανικής μάθησης απευθείας στο πρόγραμμα περιήγησης, ανοίγοντας νέες δυνατότητες για διαδραστικές και εξατομικευμένες εμπειρίες ιστού.

Η ραγδαία εξέλιξη και η ευρεία υιοθέτηση της JavaScript υποδηλώνουν ότι θα συνεχίσει να αποτελεί κυρίαρχη δύναμη στην τεχνολογική βιομηχανία. Η γλώσσα εξελίσσεται συνεχώς, με την επιτροπή ECMAScript να κυκλοφορεί τακτικά νέες εκδόσεις με πρόσθετα χαρακτηριστικά και βελτιώσεις. Η αυξανόμενη δημοτικότητα των Progressive Web Apps (PWAs), οι οποίες προσφέρουν εμπειρίες που μοιάζουν με native εφαρμογές στον web, είναι πιθανό να εδραιώσει περαιτέρω τη θέση της JavaScript. Καθώς όλο και περισσότερες επιχειρήσεις και προγραμματιστές αγκαλιάζουν τις PWAs, η JavaScript θα διαδραματίσει καθοριστικό ρόλο στην παροχή απρόσκοπτων και ελκυστικών εμπειριών για τους χρήστες[16]. Επιπλέον, το αυξανόμενο ενδιαφέρον για το Web3, ένα αποκεντρωμένο διαδίκτυο που τροφοδοτείται από την τεχνολογία blockchain, παρουσιάζει νέες ευκαιρίες για τους προγραμματιστές JavaScript. Καθώς οι εφαρμογές Web3 κερδίζουν έδαφος, ο ρόλος της JavaScript στην κατασκευή αποκεντρωμένων εφαρμογών (DApps) και στην αλληλεπίδραση με δίκτυα blockchain θα γίνεται όλο και πιο σημαντικός[17], [18].

Η JavaScript έχει εξελιχθεί από μια απλή γλώσσα σεναρίων σε μια ισχυρή και ευέλικτη γλώσσα προγραμματισμού που έχει μεταμορφώσει την τεχνολογική βιομηχανία. Ο αντίκτυπός της επεκτείνεται πέρα από την ανάπτυξη ιστοσελίδων, επηρεάζοντας την ανάπτυξη κινητών συσκευών, το serverless computing, το IoT και πολλά άλλα. Με τη συνεχή εξέλιξή της, την ισχυρή υποστήριξη της κοινότητας και την αυξανόμενη σημασία της στις αναδυόμενες τεχνολογίες, η JavaScript είναι έτοιμη να παραμείνει θεμελιώδες μέρος του τεχνολογικού τοπίου για τα επόμενα χρόνια[19], [20].

2.7 Vue.Js

Η Vue.js είναι ένα δημοφιλές framework της JavaScript για την κατασκευή σύγχρονων εφαρμογών ιστού. Δημιουργήθηκε από τον Evan You το 2014. Έχει γίνει γρήγορα ένα από τα πιο ευρέως χρησιμοποιούμενα εργαλεία frontend λόγω της απλότητας, της αντιδραστικότητας και της ευελιξίας του. Η κύρια εστίασή του είναι στο "επίπεδο προβολής" μιας εφαρμογής, το οποίο ασχολείται με την οπτική αναπαράσταση των δεδομένων. Συνδυάζει τη δηλωτική απόδοση με μια αρχιτεκτονική βασισμένη σε συστατικά, καθιστώντας εύκολη τη δημιουργία σύνθετων διεπαφών χρήστη που είναι αρθρωτές και συντηρήσιμες[21].

Ένα από τα καθοριστικά χαρακτηριστικά της Vue.js είναι το σύστημα αντιδραστικότητας(reactivity system). Τα αντικείμενα δεδομένων μέσα σε μια εφαρμογή είναι "αντιδραστικά", πράγμα που σημαίνει ότι όταν τα δεδομένα αλλάζουν, η προβολή (διεπαφή χρήστη) ενημερώνεται αυτόματα για να αντικατοπτρίζει αυτές τις αλλαγές. Η Vue.js χρησιμοποιεί ένα εικονικό DOM (Document Object Model) για την αποτελεσματική ενημέρωση μόνο των τμημάτων του DOM που πρέπει να αλλάξουν. Αυτό επιτρέπει μια ομαλή και ευέλικτη εμπειρία χρήστη, ακόμη και σε εφαρμογές με σύνθετες και δυναμικές διεπαφές χρήστη[22].

Η Vue.js χρησιμοποιεί μια δηλωτική προσέγγιση για το render του view. Οι προγραμματιστές μπορούν να δηλώσουν την επιθυμητή κατάσταση της προβολής χρησιμοποιώντας μια πρότυπη σύνταξη, η οποία περιλαμβάνει HTML, JavaScript και ειδικές οδηγίες για τη Vue. Στη συνέχεια, η Vue.js αποδίδει αυτόματα την προβολή με βάση το πρότυπο και τα δεδομένα της εφαρμογής. Αυτή η προσέγγιση απλοποιεί τον κώδικα, τον καθιστά πιο ευανάγνωστο και αφαιρεί τις πολυπλοκότητες του άμεσου χειρισμού του DOM. Προωθεί μια αρχιτεκτονική βασισμένη σε components, όπου μια εφαρμογή χτίζεται χρησιμοποιώντας επαναχρησιμοποιόντας αυτά. Ένα component είναι μια αυτοτελής μονάδα που ενθυλακώνει ένα μέρος της διεπαφής χρήστη, συμπεριλαμβανομένης της σήμανσης, των δεδομένων και της συμπεριφοράς του. Τα components μπορούν να φωλιάζουν μέσα σε άλλα components για τη δημιουργία μιας ιεραρχικής δομής. Αυτή η αρθρωτή προσέγγιση καθιστά εύκολη τη διαχείριση της πολυπλοκότητας μεγάλων εφαρμογών, καθώς κάθε συστατικό μπορεί να αναπτυχθεί και να συντηρηθεί ανεξάρτητα.

Η Vue.js παρέχει ένα σύνολο από directives που επιτρέπουν στους προγραμματιστές να αποδίδουν συμπεριφορά σε στοιχεία DOM. Οι οδηγίες έχουν ως πρόθεμα τη λέξη-κλειδί "v-" και μπορούν να χρησιμοποιηθούν για τη δέσμευση δεδομένων σε στοιχεία, το χειρισμό της εισόδου του χρήστη, τον έλεγχο της ροής της απεικόνισης και πολλά άλλα. Για παράδειγμα, η οδηγία "v-model" παρέχει αμφίδρομη δέσμευση δεδομένων, επιτρέποντας την αυτόματη αντανάκλαση της εισόδου του χρήστη στα δεδομένα της εφαρμογής και το αντίστροφο.

Προσφέρει computed properties και watchers για να βοηθήσει τους προγραμματιστές να διαχειριστούν dependencies και παρενέργειες στις εφαρμογές τους. Τα computed properties είναι παράγωγες τιμές που ενημερώνονται αυτόματα όταν αλλάζουν τα dependencies τους. Οι watchers,

από την άλλη πλευρά, επιτρέπουν στους προγραμματιστές να εκτελούν προσαρμοσμένη λογική σε απόκριση σε αλλαγές στα δεδομένα. Και τα δύο χαρακτηριστικά διευκολύνουν τη δημιουργία αποτελεσματικών και διαδραστικών εφαρμογών.

Η διεπαφή γραμμής εντολών Vue (CLI) είναι ένα ισχυρό εργαλείο που απλοποιεί τη διαδικασία εγκατάστασης και διαχείρισης έργων Vue.js. Παρέχει ένα σύνολο από templates και ρυθμίσεων για διαφορετικούς τύπους έργων, συμπεριλαμβανομένων single-page applications, progressive web apps και server-rendered apps. Το Vue CLI ενσωματώνεται επίσης με δημοφιλή εργαλεία όπως το Babel, το Webpack και το ESLint, καθιστώντας εύκολη την προσαρμογή της διαδικασίας κατασκευής και την επιβολή προτύπων ποιότητας κώδικα.

Το Vue Router είναι η επίσημη βιβλιοθήκη δρομολόγησης για εφαρμογές Vue.js. Επιτρέπει στους προγραμματιστές να ορίζουν διαδρομές και να πλοηγούνται μεταξύ διαφορετικών προβολών μέσα σε μια εφαρμογή μιας σελίδας. Το Vue Router υποστηρίζει χαρακτηριστικά όπως τα nested routes, παραμέτρους διαδρομής και navigation guards, καθιστώντας εύκολη τη δημιουργία σύνθετων και δυναμικών εφαρμογών με ομαλή εμπειρία χρήσης.

Η Vuex είναι μια βιβλιοθήκη διαχείρισης κατάστασης για εφαρμογές Vue.js. Παρέχει έναν κεντρικό αποθηκευτικό χώρο για τη διαχείριση της κατάστασης, των ενεργειών και των μεταλλάξεων της εφαρμογής. Το Vuex διευκολύνει τη διαχείριση κοινής κατάστασης σε πολλαπλά στοιχεία, την παρακολούθηση των αλλαγών στην κατάσταση και την υλοποίηση σύνθετης λογικής και πλευρικών επιδράσεων. Ενσωματώνεται επίσης με το reactivity system της Vue, εξασφαλίζοντας ότι η προβολή ενημερώνεται αυτόματα όταν αλλάζει η κατάσταση.

Το Vue DevTools είναι μια επέκταση του προγράμματος περιήγησης που παρέχει μια σειρά εργαλείων για την αποσφαλμάτωση και τη δημιουργία προφίλ εφαρμογών Vue.js. Επιτρέπει στους προγραμματιστές να επιθεωρούν τα στοιχεία, τα δεδομένα και τα συμβάντα της εφαρμογής, καθώς και να εντοπίζουν τα σημεία συμφόρησης της απόδοσης και να βελτιστοποιούν την απόδοση της εφαρμογής.

Χρησιμοποιείται ευρέως για τη δημιουργία εφαρμογών μιας σελίδας (SPA), όπου ολόκληρη η εφαρμογή φορτώνεται μία φορά και οι επακόλουθες αλληλεπιδράσεις διεκπεραιώνονται χωρίς επαναφόρτωση της σελίδας[23]. Η Vue.js παρέχει ένα ισχυρό σύνολο εργαλείων για τη διαχείριση της κατάστασης της εφαρμογής, το χειρισμό της εισόδου του χρήστη και τη δυναμική απόδοση της προβολής. Αυτό καθιστά εύκολη τη δημιουργία ευέλικτων και διαδραστικών SPAs που παρέχουν απρόσκοπτη εμπειρία χρήστη.

Η Vue.js είναι επίσης μια δημοφιλής επιλογή για την κατασκευή προοδευτικών εφαρμογών ιστού (PWA), οι οποίες είναι εφαρμογές ιστού που παρέχουν μια εμπειρία που μοιάζει με native στο διαδίκτυο. Οι PWAs μπορούν να εγκατασταθούν στη συσκευή του χρήστη, να λειτουργούν εκτός σύνδεσης και να παρέχουν χαρακτηριστικά όπως push notifications και συγχρονισμό στο παρασκήνιο. Η Vue.js διευκολύνει την κατασκευή PWAs που είναι γρήγορες, αξιόπιστες και ελκυστικές.

Υποστηρίζει server-side rendering, όπου η αρχική προβολή της εφαρμογής απεικονίζεται στον διακομιστή και αποστέλλεται στον πελάτη ως HTML. Αυτό βελτιώνει τις επιδόσεις της εφαρμογής, τη βελτιστοποίηση μηχανών αναζήτησης και την εμπειρία χρήστη, ειδικά σε αργά δίκτυα. Παρέχει ένα σύνολο εργαλείων για τη διαχείριση της κατάστασης της εφαρμογής και το render του view τόσο στον πελάτη όσο και στον διακομιστή.

Ένα από τα σημαντικότερα πλεονεκτήματα της Vue.js είναι η ευκολία εκμάθησής του. Έχει σχεδιαστεί με γνώμονα την απλότητα και η βασική βιβλιοθήκη της επικεντρώνεται μόνο στο επίπεδο

προβολής. Ακόμα και προγραμματιστές με βασικές γνώσεις HTML, CSS και JavaScript μπορούν να ξεκινήσουν την κατασκευή εφαρμογών στο Vue.js με σχετική ευκολία.

Προσφέρει υψηλό επίπεδο ευελιξίας. Μπορεί εύκολα να ενσωματωθεί σε έργα που χρησιμοποιούν άλλες βιβλιοθήκες JavaScript και η modular αρχιτεκτονική του επιτρέπει στους προγραμματιστές να επιλέγουν μόνο τα κομμάτια που χρειάζονται.Μπορεί να χρησιμοποιηθεί για την ανάπτυξη μικρών διαδραστικών τμημάτων ενός έργου, καθώς και για την κατασκευή σύνθετων single-page applications.

Η Vue.js χρησιμοποιεί ένα εικονικό DOM, το οποίο βελτιστοποιεί την ενημέρωση και την απόδοση των στοιχείων στη διεπαφή χρήστη. Αυτό σημαίνει ότι μόνο τα τμήματα της σελίδας που πρέπει να αλλάξουν αναδημιουργούνται εκ νέου, καθιστώντας τις εφαρμογές Vue.js γρήγορες και αποδοτικές. Επιπλέον, παρέχει εργαλεία παρακολούθησης των επιδόσεων, βοηθώντας τους προγραμματιστές να βελτιστοποιήσουν τις εφαρμογές τους.

Διαθέτει μια ισχυρή και ενεργή κοινότητα που συμβάλλει συνεχώς στην ανάπτυξή του. Αυτό περιλαμβάνει τακτικές ενημερώσεις, διορθώσεις και μια μεγάλη ποικιλία από plugins και εργαλεία που δημιουργούνται από την κοινότητα. Η ζωηρή κοινότητα εξασφαλίζει ότι οι προγραμματιστές έχουν πρόσβαση σε πληθώρα πόρων, συμπεριλαμβανομένων σεμιναρίων, φόρουμ και χρήσιμης τεκμηρίωσης.

Παρόλο που το Vue.js διαθέτει μια αναπτυσσόμενη κοινότητα και έναν αυξανόμενο αριθμό διαθέσιμων πόρων, εξακολουθεί να υστερεί σε σχέση με πιο καθιερωμένα framework όπως η React ή η Angular όσον αφορά το εύρος και το βάθος των tutorials, των μαθημάτων και των third-party εργαλείων[24].

Για πολύ μικρά έργα, η χρήση της Vue.js μπορεί να επιφέρει μεγαλύτερη επιβάρυνση από ό,τι είναι απαραίτητο. Το framework παρέχει πολλά χαρακτηριστικά που μπορεί να μην είναι απαραίτητα για απλά έργα και η εγκατάστασή του μπορεί να απαιτήσει περισσότερο χρόνο από ό,τι η απλή χρήση της vanilla JavaScript.

Σε ορισμένες περιπτώσεις, η ενσωμάτωση της Vue.js σε μεγάλα έργα, ειδικά σε εκείνα που δεν έχουν κατασκευαστεί με Vue.js από την αρχή, μπορεί να αποτελέσει πρόκληση. Η διαδικασία μπορεί να απαιτεί την επανεγγραφή τμημάτων του κώδικα ή την αντιμετώπιση ζητημάτων συμβατότητας με άλλες βιβλιοθήκες και frameworks.

Η Vue.js παρέχει mixins ως έναν ευέλικτο τρόπο διανομής επαναχρησιμοποιήσιμων λειτουργιών για τα συστατικά της Vue. Ένα αντικείμενο mixin μπορεί να περιέχει οποιεσδήποτε επιλογές component. Όταν ένα component χρησιμοποιεί ένα mixin, όλες οι επιλογές στο mixin θα "αναμιχθούν" στις επιλογές του ίδιου του component.

Επίσης επιτρέπει τον ορισμό φίλτρων που μπορούν να χρησιμοποιηθούν για την εφαρμογή κοινής μορφοποίησης κειμένου. Τα φίλτρα μπορούν να χρησιμοποιηθούν σε δύο σημεία: παρεμβολές mustache και εκφράσεις v-bind. Τα φίλτρα πρέπει να προσαρτώνται στο τέλος της έκφρασης JavaScript, που υποδηλώνεται με το σύμβολο "pipe".

Η Vue.js διαθέτει ένα ενσωματωμένο σύστημα transitions που επιτρέπει να εφαρμόζονται αυτόματα εφέ μετάβασης όταν στοιχεία εισάγονται ή αφαιρούνται από το DOM. Η Vue.js παρέχει διάφορα hooks για τον έλεγχο των χρονισμών των animation/transition και επίσης ενσωματώνεται καλά με CSS animations και βιβλιοθήκες animation τρίτων.

Υποστηρίζει server side rendering, επιτρέποντάς να γίνοται render τα components σε HTML strings στον server και να τα στέλνετε απευθείας στον browser. Αυτό επιτρέπει καλύτερες επιδόσεις και SEO, καθώς η σελίδα μπορεί να ευρετηριαστεί από τις μηχανές αναζήτησης χωρίς την εκτέλεση JavaScript.

Το Vue Native είναι ένα framework για τη δημιουργία native εφαρμογών για κινητά cross-platform με χρήση της Vue.js. Μεταγλωττίζεται σε ένα πρότυπο React Native, το οποίο μπορεί στη συνέχεια να προσαρμοστεί περαιτέρω χρησιμοποιώντας τη σύνταξη του React Native.

Η Vue.js είναι εξαιρετική για τη δημιουργία δυναμικών διεπαφών ιστού, ειδικά όταν χρειάζεστε ένα διαδραστικό UI με στοιχεία που έχουν μεταβαλλόμενα δεδομένα. Τα παραδείγματα περιλαμβάνουν πίνακες οργάνων, οπτικοποιήσεις δεδομένων και διαδραστικές φόρμες.

Η Vue.js είναι κατάλληλη για SPAs όπου η πλειοψηφία της αλληλεπίδρασης με τον χρήστη συμβαίνει σε μία μόνο ιστοσελίδα, με κλήσεις AJAX που χρησιμοποιούνται για τη φόρτωση δεδομένων ανάλογα με τις ανάγκες.

Οι προγραμματιστές επίσης μπορούν να δημιουργήσουν PWAs που προσφέρουν μια εμπειρία που μοιάζει με εφαρμογή για κινητά. Αυτές είναι γρήγορες, αξιόπιστες και μπορούν να λειτουργήσουν εκτός σύνδεσης ή σε δίκτυα χαμηλής ποιότητας.

Η Vue.js μπορεί να χρησιμοποιηθεί για τη δημιουργία δυναμικών εμπειριών αγορών με διαδραστικές αναζητήσεις προϊόντων, φίλτρα και ενημερώσεις καλαθιού αγορών σε πραγματικό χρόνο.

Μια άλλη χρήση είναι με το WebSocket για την ανάπτυξη εφαρμογών πραγματικού χρόνου, όπως εφαρμογές συνομιλίας, ζωντανές αθλητικές ενημερώσεις και αναλύσεις σε πραγματικό χρόνο.

Η Vue.js μπορεί να χρησιμοποιηθεί για την κατασκευή του front end των πλατφορμών CMS, παρέχοντας μια ευέλικτη και διαδραστική εμπειρία χρήστη.

Με το Vue Native, η Vue.js μπορεί να χρησιμοποιηθεί για τη δημιουργία εφαρμογών κινητών συσκευών cross-platform για Android και iOS.

Η Vue.js είναι ένα ευέλικτο και ισχυρό framework για την κατασκευή διεπαφών χρήστη. Το reactivity system, το δηλωτικο rendering και η αρχιτεκτονική που βασίζεται σε components καθιστούν εύκολη τη δημιουργία σύνθετων και δυναμικών εφαρμογών που είναι modular και συντηρήσιμες. Το οικοσύστημα της παρέχει ένα σύνολο εργαλείων για τη διαχείριση της κατάστασης της εφαρμογής, της δρομολόγησης και της διαδικασίας κατασκευής, καθώς και για την αποσφαλμάτωση και τη δημιουργία προφίλ της εφαρμογής. Η Vue.js χρησιμοποιείται ευρέως για την κατασκευή εφαρμογών μιας σελίδας, προοδευτικών εφαρμογών ιστού και εφαρμογών απόδοσης από την πλευρά του διακομιστή, παρέχοντας μια απρόσκοπτη και ελκυστική εμπειρία χρήστη. Με την απλότητα, την ευελιξία και την ενεργή κοινότητά του, το Vue.js συνεχίζει να αποτελεί δημοφιλή επιλογή για την ανάπτυξη frontend.

2.8 Chart.Js

Το Chart.js είναι μια βιβλιοθήκη της JavaScript, ανοικτού κώδικα που χρησιμοποιείται για τη δημιουργία δυναμικών και διαδραστικών απεικονίσεων δεδομένων με τη μορφή διαγραμμάτων και γραφικών παραστάσεων. Η βιβλιοθήκη παρέχει ένα ευρύ φάσμα επιλογών προσαρμογής και υποστηρίζει διάφορους τύπους διαγραμμάτων, όπως γραμμή, ράβδος, πίτα, διασπορά και ραντάρ. Αναπτύχθηκε από τον Nick Downie και κυκλοφόρησε αρχικά τον Μάρτιο του 2013, το Chart.js έγινε

γρήγορα μια δημοφιλής επιλογή μεταξύ των προγραμματιστών λόγω της απλότητας, της ευελιξίας και της ικανότητάς του να παράγει οπτικά ελκυστικά διαγράμματα.

Το Chart.js προσφέρει ένα ευρύ φάσμα χαρακτηριστικών που το καθιστούν ένα ισχυρό εργαλείο για την οπτικοποίηση δεδομένων. Τα χαρακτηριστικά αυτά περιλαμβάνουν:

Μεγάλο εύρος τύπων διαγραμμάτων: Το Chart.js υποστηρίζει μια ποικιλία τύπων διαγραμμάτων, όπως γραμμικά διαγράμματα, ραβδογράμματα, κυκλικά διαγράμματα, διαγράμματα διασποράς, διαγράμματα ραντάρ και διαγράμματα φούσκας. Αυτή η ευελιξία επιτρέπει στους προγραμματιστές να επιλέγουν τον κατάλληλο τύπο διαγράμματος για την αποτελεσματική αναπαράσταση των δεδομένων τους.

Customization: Το Chart.js παρέχει πολυάριθμες επιλογές προσαρμογής για άξονες, ετικέτες, tooltips, animations και πολλά άλλα. Οι προγραμματιστές μπορούν να διαμορφώσουν και να μορφοποιήσουν τα διαγράμματά τους ώστε να ταιριάζουν με το σχεδιασμό και την επωνυμία του ιστότοπού τους.

Responsive Design: Τα διαγράμματα που δημιουργούνται με το Chart.js ανταποκρίνονται από προεπιλογή. Η βιβλιοθήκη προσαρμόζει αυτόματα τις διαστάσεις του διαγράμματος με βάση το μέγεθος του container, διασφαλίζοντας ότι το διάγραμμα φαίνεται καλά σε διαφορετικά μεγέθη οθόνης και συσκευές.

Διαδραστικότητα: Υποστηρίζει διάφορα διαδραστικά χαρακτηριστικά, όπως tooltips, hover effects και click events. Οι χρήστες μπορούν να αλληλεπιδρούν με τα διαγράμματα για να λαμβάνουν πρόσθετες πληροφορίες ή να ενεργοποιούν συγκεκριμένες ενέργειες.

Απόδοση: Το Chart.js είναι βελτιστοποιημένο για απόδοση, εξασφαλίζοντας ομαλές κινούμενες εικόνες και γρήγορη απόδοση των διαγραμμάτων ακόμη και με μεγάλα σύνολα δεδομένων.

Ενσωμάτωση με Front-end Frameworks: Μεγάλο πλεονέκτημα ότι είναι συμβατό με δημοφιλή front-end frameworks όπως η Angular, η React και η Vue.js. Οι προγραμματιστές μπορούν εύκολα να ενσωματώσουν το Chart.js στις υπάρχουσες διαδικτυακές εφαρμογές τους που έχουν κατασκευαστεί με αυτά τα frameworks.

Για να δημιουργήσει ένα γράφημα χρησιμοποιώντας το Chart.js, ένας προγραμματιστής πρέπει να συμπεριλάβει τη βιβλιοθήκη Chart.js στην ιστοσελίδα του και να δημιουργήσει ένα στοιχείο canvas στο αρχείο HTML. Το στοιχείο canvas χρησιμεύει ως δοχείο για το γράφημα. Στη συνέχεια, ο προγραμματιστής γράφει κώδικα JavaScript για να καθορίσει τη διαμόρφωση, τα δεδομένα και τις επιλογές του γραφήματος. Τέλος, δημιουργούν ένα instance του γραφήματος χρησιμοποιώντας τον Chart constructor και του μεταβιβάζουν το configuration.

Αν και το Chart.js είναι μια ισχυρή βιβλιοθήκη για τη δημιουργία διαγραμμάτων, ενδέχεται να μην είναι η καλύτερη επιλογή για κάθε έργο. Ορισμένοι περιορισμοί περιλαμβάνουν:

Πολύπλοκες οπτικοποιήσεις: Chart.js δεν έχει σχεδιαστεί για πολύπλοκες απεικονίσεις δεδομένων, όπως διαγράμματα δικτύου, γεωγραφικοί χάρτες ή δενδροειδή διαγράμματα. Για αυτές τις περιπτώσεις χρήσης, το D3.js ή το Highcharts μπορεί να είναι πιο κατάλληλα.

Τρισδιάστατα διαγράμματα: Το Chart.js δεν υποστηρίζει τρισδιάστατα διαγράμματα. Βιβλιοθήκες όπως η Three.js ή η Plotly.js είναι καταλληλότερες για τρισδιάστατες απεικονίσεις.

Server-Side Rendering: Το Chart.js βασίζεται στο στοιχείο HTML5 canvas, καθιστώντας την απόδοση των διαγραμμάτων στην πλευρά του διακομιστή πρόκληση. Για την απόδοση από την πλευρά του

διακομιστή, βιβλιοθήκες όπως η Matplotlib (Python) ή η ggplot2 (R) μπορεί να είναι καλύτερες επιλογές.

Το Chart.js διαθέτει ένα πλούσιο οικοσύστημα από πρόσθετα που δημιουργήθηκαν τόσο από την core ομάδα όσο και από την κοινότητα. Οι προγραμματιστές μπορούν να χρησιμοποιήσουν αυτά τα πρόσθετα για να βελτιώσουν τα διαγράμματά τους ή ακόμη και να δημιουργήσουν προσαρμοσμένα πρόσθετα προσαρμοσμένα στις συγκεκριμένες ανάγκες τους. Επωφελείται από μια ζωντανή κοινότητα προγραμματιστών και συνεργατών που συμμετέχουν ενεργά στην ανάπτυξη της βιβλιοθήκης. Το αποθετήριο GitHub έχει μεγάλο αριθμό αστέρων, διχασμών και συνεισφερόντων, γεγονός που υποδηλώνει το έντονο ενδιαφέρον της κοινότητας για το έργο. Η ενεργή κοινότητα όχι μόνο βοηθά στη βελτίωση και τη συντήρηση της βιβλιοθήκης, αλλά παρέχει επίσης υποστήριξη σε άλλους προγραμματιστές μέσω φόρουμ, συζητήσεων και διαδικτυακών πλατφορμών.

Το ολοκληρωμένο documentation του Chart.js είναι ένα ακόμη δυνατό του σημείο. Ο επίσημος ιστότοπος προσφέρει λεπτομερείς οδηγούς και σεμινάρια για το πώς να ξεκινήσετε, να προσαρμόσετε τα διαγράμματα και να χρησιμοποιήσετε plugins. Περιλαμβάνει επίσης μια διαδραστική ενότητα επίδειξης όπου οι χρήστες μπορούν να εξερευνήσουν διαφορετικούς τύπους διαγραμμάτων και επιλογές προσαρμογής. Αυτή η εμπεριστατωμένη τεκμηρίωση βοηθά τους προγραμματιστές να μάθουν γρήγορα και να εφαρμόσουν το Chart.js στα έργα τους.

Η ευελιξία και οι επιλογές προσαρμογής του Chart.js το καθιστούν κατάλληλο για διάφορες περιπτώσεις χρήσης. Οι επιχειρήσεις μπορούν να το χρησιμοποιήσουν για τη δημιουργία ταμπλό για την παρακολούθηση των πωλήσεων, του μάρκετινγκ και των λειτουργικών μετρήσεων. Οι επιστήμονες δεδομένων και οι αναλυτές μπορούν να το χρησιμοποιήσουν για την οπτικοποίηση δεδομένων και τον εντοπισμό μοτίβων, τάσεων και ακραίων τιμών. Οι προγραμματιστές μπορούν να ενσωματώσουν το Chart.js σε εφαρμογές ιστού και να παρέχουν στους χρήστες διαδραστικές απεικονίσεις δεδομένων, βελτιώνοντας το engagement των χρηστών και παρέχοντας πληροφορίες.:

Marketing Analytics: Παρακολούθηση και οπτικοποίηση μετρήσεων όπως η επισκεψιμότητα του ιστότοπου, η απόκτηση χρηστών και τα ποσοστά μετατροπής.

Χρηματοοικονομική ανάλυση: Εμφανίση οικονομικών δεδομένων όπως τιμές μετοχών, έσοδα και έξοδα σε διαγράμματα για καλύτερη κατανόηση και λήψη αποφάσεων.

Υγειονομική περίθαλψη: Οπτικοποίηση δεδομένων ασθενών, συμπεριλαμβανομένων ζωτικών σημείων, εργαστηριακών αποτελεσμάτων και ιατρικού ιστορικού.

Εκπαίδευση: Χρήση διαγραμμάτων για να αναπαράσταση επιδόσεων, της φοίτηση και την πρόοδο των μαθητών με την πάροδο του χρόνου.

Εφαρμογές IoT: Εμφάνιση δεδομένων σε πραγματικό χρόνο από συνδεδεμένες συσκευές, αισθητήρες και συστήματα.

Συνοψίζοντας, το Chart.js είναι μια ευέλικτη, εύχρηστη και ισχυρή βιβλιοθήκη JavaScript για τη δημιουργία διαδραστικών διαγραμμάτων και οπτικοποιησης δεδομένων. Υποστηρίζει διάφορους τύπους διαγραμμάτων, επιλογές για customization και plugins, καθιστώντας το κατάλληλο για ένα ευρύ φάσμα εφαρμογών. Η ενεργή κοινότητα και το ολοκληρωμένο documentation ενισχύουν περαιτέρω την ελαστικότητά της. Παρόλο που μπορεί να μην είναι ιδανική για σύνθετες ή τρισδιάστατες απεικονίσεις, αποτελεί εξαιρετική επιλογή για τις περισσότερες ανάγκες απεικόνισης δεδομένων στον ιστό[25].

Κεφάλαιο 3ο: Ανάκτηση δεδομένων και Δημιουργία της Βάσης Δεδομένων

3.1 Πρωτογενή δεδομένα

Τα πρωτογενή δεδομένα είναι αυτά που ανακοινώνονται από του Υπουργείο παιδείας κάθε χρόνο. Γίνεται δημοσίευση των αρχείων αυτών τουλάχιστον τρεις φορές την χρονιά και με πολλαπλές ανακοινώσεις που μπορεί να διαχωρίζονται ανα επίπεδο εκπαίδευσης (πρωτοβάθμια, δευτεροβάθμια) ή και τύπου εκπαίδευσης (ειδικής, τυπικής). Αυτή η πληθώρα ανακοινώσεων αρχείων και μάλιστα όχι σε κάποιες στάνταρ ημερομηνίες είναι που κάνει και πολύ δύσκολη την «συγκομιδή» τους.

Ένα παράδειγμα ενός τέτοιου αρχείου είναι το παρακάτω.

A	B	C	D	E	F	G	H	I	J	K
Α/Α	Α/Α ΡΟΗ	ΤΥΠΟΣ	ΕΠΩΝΥΜΟ	ΟΝΟΜΑ	ΠΑΤΡΩΝΥΜΟ	ΚΛΑΔΟΣ	ΕΙΔΙΚΟΤΗΤΑ	ΠΙΝΑΚΑΣ	ΣΕΙΡΑ ΠΙΝΑΚΑ	ΜΟΡΙΑ ΠΙΝΑΚΑ
3	25	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΓΑΛΑΝΗ	ΑΓΓΕΛΙΚΗ	ΣΩΤΗΡΙΟΣ	ΠΕ05	ΠΕ05	Α	1017	40.2
4	26	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΚΑΡΑΚΑΣΗ	ΑΡΓΥΡΩ	ΠΕΡΙΚΛΗΣ	ΠΕ05	ΠΕ05	Α	1028	39.93
5	28	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΧΡΥΣΑΝΘΑΚΟΠΟΥΛΟΥ	ΜΑΡΙΑ	ΙΩΑΝΝΗΣ	ΠΕ05	ΠΕ05	Α	1065	38.63
6	29	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΣΤΟΥΜΠΟΥ	ΕΛΕΥΘΕΡΙΑ	ΗΛΙΑΣ	ΠΕ05	ΠΕ05	Α	1149	35.58
7	30	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΚΟΛΙΟΥΛΗ	ΣΟΦΙΑ	ΝΙΚΟΛΑΟΣ	ΠΕ05	ΠΕ05	Α	1195	34.7
8	31	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΚΑΚΛΙΔΑΚΗ	ΑΓΑΠΗ	ΕΜΜΑΝΟΥΗΛ	ΠΕ05	ΠΕ05	Α	1271	33
9	1	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΜΠΙΣΑ	ΜΑΡΙΑ	ΣΩΤΗΡΙΟΣ	ΠΕ06	ΠΕ06	Α	1214	55.8
10	2	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΠΑΠΑΛΕΞΗ	ΜΑΡΙΑ	ΔΗΜΗΤΡΙΟΣ	ΠΕ06	ΠΕ06	Α	1323	53.25
11	3	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΣΤΡΑΤΙΔΟΥ	ΒΑΣΙΛΙΚΗ	ΑΛΕΞΑΝΔΡΟΣ	ΠΕ06	ΠΕ06	Α	1803	46.38
12	4	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΤΣΑΓΚΟΥΡΝΟΥ	ΠΑΝΑΓΙΩΤΑ	ΠΑΝΑΓΙΩΤΗΣ	ΠΕ06	ΠΕ06	Α	2044	43.3
13	5	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΦΥΛΑΚΤΟΥ	ΜΑΡΙΑ	ΘΩΜΑΣ	ΠΕ06	ΠΕ06	Α	2073	42.88
14	6	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΚΑΛΑΓΚΙΑ	ΜΑΡΙΑ	ΑΝΤΩΝΙΟΣ	ΠΕ06	ΠΕ06	Α	2148	41.83
15	7	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΜΕΧΜΕΤ	ΣΙΜΠΕΛ	ΧΑΣΑΝ	ΠΕ06	ΠΕ06	Α	2316	39.4
16	8	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΝΙΚΟΛΑΟΥ	ΒΑΡΒΑΡΑ	ΔΙΟΝΥΣΙΟΣ	ΠΕ06	ΠΕ06	Α	2608	33.43
17	9	ΝΗΠΙΑΓΩΓΕΙΟ	ΜΥΛΩΝΑ	ΜΑΡΙΑ	ΔΗΜΗΤΡΙΟΣ	ΠΕ06	ΠΕ06	Α	2880	29.9
18	10	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΓΙΑΝΝΟΥΛΗ	ΓΕΩΡΓΙΑ	ΙΩΑΝΝΗΣ	ΠΕ06	ΠΕ06	Α	2892	29.85
19	12	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΒΑΡΒΙΤΣΙΩΤΗ	ΑΓΓΕΛΙΚΗ	ΑΓΓΕΛΟΣ	ΠΕ06	ΠΕ06	Α	3022	28.6
20	13	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΛΑΝΔΡΟΥ	ΑΙΚΑΤΕΡΙΝΗ-ΕΜΜΑΝΟΥΕΛΑ	ΔΗΜΗΤΡΙΟΣ	ΠΕ06	ΠΕ06	Α	3144	27.78
21	14	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΑΘΑΝΑΣΟΠΟΥΛΟΥ	ΓΕΩΡΓΙΑ ΜΑΡΙΑ	ΑΘΑΝΑΣΙΟΣ	ΠΕ06	ΠΕ06	Α	3164	27.63
22	15	ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ	ΣΠΗΛΙΟΠΟΥΛΟΣ	ΑΝΤΩΝΙΟΣ	ΚΩΝΣΤΑΝΤΙΝΟΣ	ΠΕ06	ΠΕ06	Α	3165	27.63

ΓΕΝΙΚΗ ΑΘΜΙΑ

Εικόνα 3.1: Αρχείο πρόσληψης αναπληρωτών

L	M	N	O
ΠΕΡΙΟΧΗ ΤΟΠΟΘΕΤΗΣΗΣ	ΩΡΑΡΙΟ	Δ/ΝΣΗ ΕΚΠ/ΣΗΣ	ΠΕΡΙΦΕΡΕΙΑ
ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΡΕΘΥΜΝΟΥ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΡΕΘΥΜΝΟΥ	ΚΡΗΤΗΣ
ΙΩΑΝΝΙΝΩΝ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΙΩΑΝΝΙΝΩΝ	ΗΠΕΙΡΟΥ
ΚΑΡΔΙΤΣΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΚΑΡΔΙΤΣΑΣ	ΘΕΣΣΑΛΙΑΣ
Β΄ ΘΕΣΣΑΛΟΝΙΚΗΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. Β΄ ΘΕΣΣΑΛΟΝΙΚΗΣ	ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
Β΄ ΑΘΗΝΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. Β΄ ΑΘΗΝΑΣ	ΑΤΤΙΚΗΣ
Δ΄ ΑΘΗΝΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. Δ΄ ΑΘΗΝΑΣ	ΑΤΤΙΚΗΣ
Α΄ ΧΙΟΥ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΧΙΟΥ	ΒΟΡΕΙΟΥ ΑΙΓΑΙΟΥ
ΡΟΔΟΠΗΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΡΟΔΟΠΗΣ	ΑΝΑΤΟΛΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ ΚΑΙ ΘΡΑΚΗΣ
ΞΑΝΘΗΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΞΑΝΘΗΣ	ΑΝΑΤΟΛΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ ΚΑΙ ΘΡΑΚΗΣ
ΤΡΙΚΑΛΩΝ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΤΡΙΚΑΛΩΝ	ΘΕΣΣΑΛΙΑΣ
ΖΑΚΥΝΘΟΥ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΖΑΚΥΝΘΟΥ	ΙΟΝΙΩΝ ΝΗΣΩΝ
ΚΟΡΙΝΘΙΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΚΟΡΙΝΘΙΑΣ	ΠΕΛΟΠΟΝΝΗΣΟΥ
Α΄ ΣΑΜΟΥ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΣΑΜΟΥ	ΒΟΡΕΙΟΥ ΑΙΓΑΙΟΥ
ΑΧΑΪΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΧΑΪΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΑΧΑΪΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΧΑΪΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΧΑΛΚΙΔΙΚΗΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΧΑΛΚΙΔΙΚΗΣ	ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
Β΄ ΘΕΣΣΑΛΟΝΙΚΗΣ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. Β΄ ΘΕΣΣΑΛΟΝΙΚΗΣ	ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΑΧΑΪΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΧΑΪΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΜΕΣΣΗΝΙΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΜΕΣΣΗΝΙΑΣ	ΠΕΛΟΠΟΝΝΗΣΟΥ
ΤΡΙΚΑΛΩΝ (Π.Ε.) - Μειωμένου Ωραρίου	ΑΜΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΤΡΙΚΑΛΩΝ	ΘΕΣΣΑΛΙΑΣ
ΑΧΑΪΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΑΧΑΪΑΣ	ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΜΕΣΣΗΝΙΑΣ (Π.Ε.)	ΑΠΩ	ΔΙΕΥΘΥΝΣΗ Π.Ε. ΜΕΣΣΗΝΙΑΣ	ΠΕΛΟΠΟΝΝΗΣΟΥ

Εικόνα 3.2: Αρχείο πρόσληψης αναπληρωτών

Αυτό είναι ένα από τα πιο πρόσφατα δημοσιευμένα αρχεία, είναι οι προλήψεις γενικής πρωτοβάθμια εκπαίδευσης του 2023. Οι στήλες που υπάρχουν στο αρχείο αυτό είναι οι Α/Α, Α/Α Ροής, Τύπος, Επώνυμο, Όνομα, Πατρώνυμο, Κλάδος, Ειδικότητα, Πίνακας, Σειρά Πίνακα, Μόρια Πίνακα, Περιοχή Τοποθέτησης, Ωράριο, Διεύθυνση Εκπαίδευσης και Περιφέρεια.

3.2 Προεπεξεργασία δεδομένων

Ένα μεγάλο πρόβλημα που εντοπίζεται στα αρχεία αυτά είναι ότι έχουν ανόμοια πεδία μεταξύ τους, αυτή ήταν και μια απο τις μεγαλύτερες δυσκολίες κατα την υλοποίηση της εφαρμογής. Παραδείγματος χάρη ένα αρχείο μπορεί να έχει το μητρώνυμο ενός εκπαιδευτικού και ένα άλλο όχι, κάποιο αρχείο να έχει μία στήλη με το ωράριο του κάθε εκπαιδευτικού και ένα άλλο όχι. Υπάρχει δηλαδή διαφορετικός αριθμός στηλών σε πολλά από τα excel αυτά αρχεία.

Οι διαφορές όμως δεν αφορούν μόνο τον αριθμό των στηλών άλλα και την ονοματολογία σε κάποιο αρχείο, ο κλάδος μπορεί να αναφέρεται σαν ειδικότητα, σε πολλά αρχεία μάλιστα υπάρχουν και οι δύο στήλες δηλαδή μια στήλη για την ειδικότητα και μία για τον κλάδο με ακριβώς τις ίδιες τιμές.

Επίσης διαφορές υπάρχουν και στις ονομασίες με την πάροδο του χρόνου, παραδείγματος χάρη, τα Κεδδύ (κέντρα διαφοροδιάγνωσης, διάγνωσης και υποστήριξης) μετονομάστηκαν την χρονιά 2019-2020 σε Κεσύ (Κέντρα Εκπαιδευτικής και Συμβουλευτικής Υποστήριξης) και αργότερα την χρονιά 2021-2022 μετανομάστηκαν σε ΚΕΔΑΣΥ (Κέντρα Διάγνωσης Αξιολόγησης Συμβουλευτικής και Υποστήριξης) ενώ υπάρχουν και αναφορές του τυπου ΚΕΔΑΣΥ - ΚΕΣΥ σε κάποια αρχεία. Δηλαδή η ακριβώς ίδια έννοια με τέσσερις διαφορετικές ονομασίες. Με τον χρόνο επίσης άλλαξαν και οι ονομασίες των ειδικοτήτων, δηλαδή το ΠΕ80 μετονομάστηκε και απο το 2019 και μετα πλέον λέγεται ΣΕ32.

Μάλιστα το format των αρχείων δεν αλλάζει μόνο από χρονιά σε χρονιά αλλα και από αρχείο σε αρχείο μέσα στην ίδια χρονιά. Για την αποθήκευση αυτών των αρχείων σε μια βάση δεδομένων θα πρέπει να έρθουν όλα αυτά τα αρχεία σε μια ομοιόμορφη μορφή. Να καθαριστούν από κάποια περιττή πληροφορία, να «συγχρονιστούν» εννιολογικά και να έρθουν σε κάποια συγκεκριμένη μορφή ώστε να βολεύουν την ίδια την εφαρμογή.

Έτσι, λόγω της φύσης των αρχείων, για την δημιουργία ενός στάνταρ φορμάτ για όλα τα αρχεία, ώστε να εισαχθούν αργότερα στην βάση, τα σημαντικότερα κριτήρια ήταν η επιλογή στηλών που βρίσκονται αν όχι σε όλα στα περισσότερα αρχεία, στην επιλογή της πιο χρήσιμης πληροφορίας, στην αναδιαμόρφωση κάποιας λιγότερης σημαντικής πληροφορίας σε κάποια πιο χρήσιμη και την απλοποίηση της πληροφορίας όπου χρειαζόταν για να είναι καταλληλότερη για το interface της web εφαρμογής. Η γενική ιδέα ήταν η προσαρμογή της πληροφορίας των αρχείων στην μορφή που πρέπει να είναι για την αποθηκευμένη στην βάση και όχι στην διαμόρφωση της βάσης με βάση την αρχική πληροφορία, ακολουθήθηκε αυτό το μονοπάτι λόγω της συνολικά κακής μορφής της αρχικής πληροφορίας αυτή για τους λόγους που αναφέρθηκαν παραπάνω.

Πιο αναλυτικά οι στήλες του προσαρμοσμένου excel που «ανεβαίνει» αργότερα στην βάση είναι οι εξής:

- Τύπος: Η στήλη τύπος είναι μια στήλη η οποία υπάρχει σε όλα τα αρχεία και η οποία αναδιαμορφώθηκε ωστε να καλύπτει τις ανάγκες της εφαρμογής. Πιο συγκεκριμένα, λόγω της πληθώρας των τιμών που είχε αυτή η στήλη, όπως μουσικά σχολεία, γενική παιδεία, braille, νοηματική,δασκαλος σε ειδική μονάδα, νηπιαγωγείο, ΔΥΕΠ, ΤΥ-ΖΕΠ και δεκάδες άλλα, αποφασίστηκε η χρήση τριών τύπων πεδίων για την ανακεφαλαίωση όλων των παραπάνω. Αυτά τα παιδεία είναι το Μουσικά, Γενικής και Ειδικής. Το πεδίο Μουσικά καλύπτει οτιδήποτε έχει να κάνει με εκπαιδευτικούς που διδάσκουν μουσική σε όλα τα επίπεδα της εκπαίδευσης. Το πεδίο Γενικής καλύπτει οποιαδήποτε ρόλο δασκάλου, σε οποιαδήποτε βαθμίδα εκπαίδευσης που ασχολείται με την διδασκαλία σε σχολεία γενικής/τυπικής εκπαίδευσης και γενικά σε μαθητές που δεν εντάσσονται στον τομέα ειδικής εκπαίδευσης. Τελος το πεδίο της Ειδικής καλύπτει οποιονδήποτε ρόλο έχει να κάνει με την ενασχόληση του εκπαιδευτικού με κάποια παιδί που κατατάσσεται σε ειδική κατηγορία, είτε αυτή είναι η διδασκαλία με την χρήση της braille και της νοηματικής ή πιο γενικα με την παράλληλη στήριξη.
- Επώνυμο, Όνομα, Πατρώνυμο: Οι τρεις αυτές στήλες είναι αναγκαίες όχι μονο για την εμφάνιση ενός εκπαιδευτικού σαν οντότητα στην εφαρμογή αλλά και γενικά για την ταυτοποίηση του εκπαιδευτικού, καθώς δεν υπάρχει κάποιο άλλο αναγνωριστικό πεδίο στα excel αρχεία όπως τον αριθμό μητρώου ή κάποιο άλλο προσωπικό μοναδικό αναγνωριστικό όπως για παράδειγμα το ΑΜΚΑ. Αν και σε κάποια αρχεία υπήρχε η στήλη του Μητρώνυμου, κάτι που αν επιλέγοταν τελικά θα ενίσχυε ακόμα περισσότερο στην ταυτοποίηση ενός εκπαιδευτικου τα αρχεία αυτά ήταν ελάχιστα και έτσι η πληροφορία συνολικα δεν ήταν επαρκής για να ενταχθεί στο τελικό format.
- Κλάδος: Η στήλη του κλαδου ήταν ακόμη μία στήλη που υπάρχει σε όλα τα δημοσιευμένα αρχεία. όπως προαναφερθηκε πολλά αρχεία είχαν στήλες με ονομασίες Κλάδος και Ειδικότητα που περιείχαν τα ίδια δεδομένα. Ετσι, η επιλογή της στήλης του Κλάδου ήταν στην πραγματικότητα μια ονοματική προτίμηση, ενώ η στήλη της ειδικότητας διαγράφτηκε. Στη στήλη του Κλάδου έγινε αλλαγές στους κωδικούς οι οποίοι άλλαξαν(διατηρόντας την ίδια σημασιολογία) ώστε να συμφωνούν όλα τα αρχεία μεταξύ τους.

- Σειρά Πίνακα, Μόρια Πίνακα: Και οι δύο στήλες περιέχουν σημαντική πληροφορία για την έκδοση συμπερασμάτων από τον εκάστοτε εκπαιδευτικό. Επίσης είναι κατάλληλες για οπτικοποίηση μετέπειτα στην εφαρμογή.
- Περιοχή Τοποθέτησης: Σημαντική πληροφορία που υπάρχει σε όλα τα αρχεία. Έγινε επεξεργασία των δεδομένων στα αρχεία ώστε να διαγραφούν ειδικοί χαρακτήρες όπως οι τόνοι οι οποίοι μπορεί να προκαλούσαν πρόβλημα αργότερα στην αποθήκευση στη βάση. Επίσης διαγράφηκε ο τύπος της διεύθυνσης εκπαίδευσης που υπήρχε (Π.Ε, Δ.Ε) και προστέθηκε σε άλλη στήλη. Τελος, έγινε μια συνολική απλοποίηση της πληροφορίας σε κάποιες ονομασίες.
- Διεύθυνση Εκπαίδευσης; Και εδώ έγιναν παρόμοιες τροποποιήσεις όπως και στην περιοχή τοποθέτησης. Αρχικά αφαιρέθηκαν και πάλι όλοι οι ειδικοί χαρακτήρες προς αποφυγή δημιουργίας προβλήματος κατα την αποθήκευση τους. Η βαθμίδα της διεύθυνσης είχε πολλές διαφορετικές μορφές στα αρχεία, εν τέλει επιλέχθηκε η μορφή του Π.Ε για πρωτοβάθμια εκπαίδευση και του Δ.Ε για την δευτεροβάθμια και τροποποιήθηκαν και όλα τα ονόματα των περιοχών ώστε να είναι στο ίδιο μοτίβο. Παραδείγματος χάρη στις περιπτώσεις περιοχών όπως αυτές της Αθήνας, της Θεσσαλονίκης κ.α οι οποίες χωρίζονται σε τμήματα, πάντα η ονοματολογία του τμήματος (Α,Β,Γ,Δ) βρίσκεται μπροστά. Παρόμοια και εδώ υπήρχαν πολλές διαφορετικές μορφές στα αρχεία και επιλέχθηκε αυτή που κρίθηκε πιο κατάλληλη και απλή.
- Ημερομηνία, Έτος: Στήλες με σημαντική χρονολογική πληροφορία. Αρχικά η στήλη της Ημερομηνίας περιέχει την πληροφορία του πότε έγινε η δημοσιοποίηση του αρχείου (π.χ 2022-11-01) και του Έτος εμπεριέχει την χρονιά πρόσληψης (π.χ 2022-2023). Αυτές είναι δύο στήλες οι οποίες δεν υπήρχαν σε κανένα αρχικό αρχείο και προστέθηκαν λόγω της σημαντικότητα τους για την εξαγωγή δεδομένων.
- Σχόλια: Η στήλη Σχόλια περιέχει δύο τύπους πληροφορίας. Αρχικά εμπεριέχει τον τύπο του ωραρίου, πλήρες ή μειωμένου που εκφράζεται με τις συμβολοσειρές ΠΩ και ΜΩ αντίστοιχα. Η δεύτερη πληροφορία είναι αυτή του μουσικού οργάνου που διδάσκει ο εκπαιδευτικός σε περίπτωση που είναι μουσικός. Αυτή η στήλη δημιουργήθηκε με συνδυασμό δυο άλλων στηλών οι οποίες δεν ήταν αρκετά σημαντικές από μόνες τους αλλα μπορούν κατα περίπτωση να παρέχουν σημαντική πληροφορία.

3.3 Φόρτωση δεδομένων στην βάση

Αφού λοιπόν έρθουν τα αρχεία στην επιθυμητή μορφή επόμενο βήμα είναι η φόρτωση αυτών των δεδομένων στην βάση. Η φόρτωση των δεδομένων στην βάση γίνεται με χρήση ενός python script και η βάση που χρησιμοποιείται είναι μια βάση MySQL. Αυτή είναι η διαδικασία φόρτωσης των δεδομένων βήμα-βήμα:

```
1  import mysql.connector
2  from mysql.connector import Error
3  import pandas as pd
4  from glob import glob
5  import numpy as np
```

Εικόνα 3.3: Βιβλιοθήκες και εργαλεία του python script

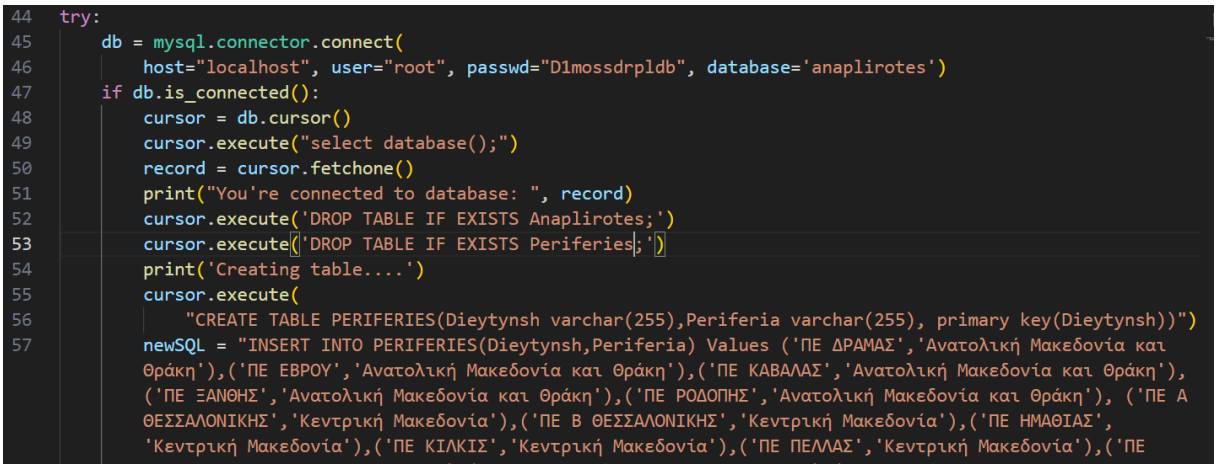
Αρχικά γίνεται η εισαγωγή όλων των εργαλείων και των βιβλιοθηκών. Πιο αναλυτικά:

- Η mysql.connector είναι ένα module το οποίο χρησιμοποιείται για την συνδεση της εφαρμογης με τη MySQL βάση. Με την χρήση του επιτρέπεται η εκτέλεση εντολών SQL και η διαχείριση λειτουργιών στη βάση.
- Το Error της mysql.mysql.connector εισάγεται ώστε να πιαστούν και να διαχειριστούν διάφορες εξαιρέσεις που σχετίζονται με την MySQL.
- Το Pandas είναι μια δημοφιλης βιβλιοθήκη ανάλυσης δεδομένων στην python που παρέχει ευέλικτες δομές δεδομένων. Είναι ιδιαίτερα γνωστή για το αντικείμενο Dataframe που χρησιμοποιείται εδώ και για οποιαδήποτε λειτουργία έχει να κάνει με csv αρχεία
- Η συνάρτηση glob χρησιμοποιείται για την ανάκτηση αρχείων που ταιριάζουν στο ίδιο μοτίβο, παραδείγματος χάρη όλα τα csv αρχεία.
- Η numpy είναι ένα από τα πιο διάσημα πακέτα στην python και χρησιμοποιείται στην εφαρμογή για τον χειρισμό των δεδομένων.

```
8   source = r'C:\Users\dimos\OneDrive\Documents\Projects\AnaplirotesCSV/'
9
10  import_files = sorted(glob(source + "*.csv"))
11  filecount = len(import_files)
12  print(filecount)
13
14  if filecount < 1:
15      print('No files found in the filepath:' + source)
16  else:
17      df = pd.concat((pd.read_csv(file)
18                      for file in import_files), ignore_index=True)
19
20      df = df.replace({np.nan: None})
```

Εικόνα 3.4: Διαχείριση αρχείων

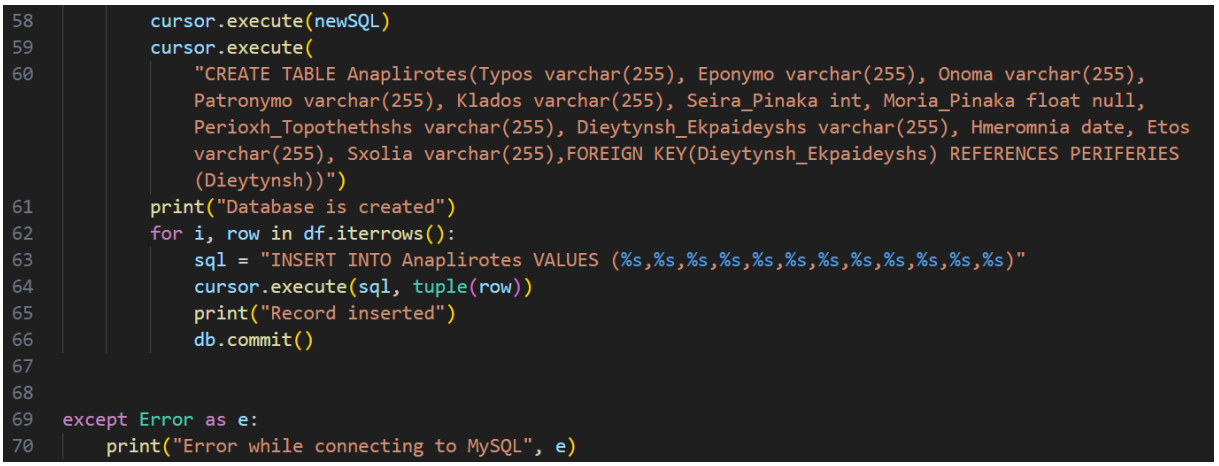
Αρχικά όλα τα csv αρχεία ομαδοποιούνται σε έναν φάκελο. Χρησιμοποιώντας μετά την συνάρτηση glob ανακτάτε μια λίστα με όλα τα αρχεία στον κατάλογο source. Η χρήση της συνάρτησης sorted() εξασφαλίζει ότι τα αρχεία βρίσκονται σε αλφαβητική σειρά με βάση τα ονόματα των αρχείων τους. Στην συνέχεια με την χρήση της συνάρτησης len() αν ο αριθμός των αρχείων είναι μεγαλύτερος του μηδενός γίνεται συνένωση όλων των αρχείων csv σε ένα ενιαίο Dataframe.Το ignore_index=True εξασφαλίζει ότι το DataFrame που προκύπτει θα έχει συνεχή δείκτη (δηλαδή δεν θα μεταφέρει τις τιμές του δείκτη από τα επιμέρους αρχεία .csv). Μετά τη συνένωση των αρχείων σε ένα ενιαίο DataFrame, ο κώδικας αντικαθιστά στη συνέχεια τυχόν τιμές NaN (Not a Number) στο DataFrame με None. Αυτό γίνεται για να εξασφαλιστεί η συμβατότητα κατά την εισαγωγή των δεδομένων με την MySQL, καθώς δεν αναγνωρίζει το NaN αλλά μπορεί να λειτουργήσει με το NULL (το οποίο στην Python αναπαρίσταται από το None).



```
try:
    db = mysql.connector.connect(
        host="localhost", user="root", passwd="D1mossdrpldb", database='anaplirotes')
    if db.is_connected():
        cursor = db.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)
        cursor.execute('DROP TABLE IF EXISTS Anaplirotes;')
        cursor.execute('DROP TABLE IF EXISTS Periferies;')
        print('Creating table....')
        cursor.execute(
            "CREATE TABLE PERIFERIES(Dieytynsh varchar(255),Periferia varchar(255), primary key(Dieytynsh))")
        newSQL = "INSERT INTO PERIFERIES(Dieytynsh,Periferia) Values ('ΠΕ ΔΡΑΜΑΣ','Ανατολική Μακεδονία και
        Θράκη'),('ΠΕ ΕΒΡΟΥ','Ανατολική Μακεδονία και Θράκη'),('ΠΕ ΚΑΒΑΛΑΣ','Ανατολική Μακεδονία και Θράκη'),
        ('ΠΕ ΞΑΝΘΗΣ','Ανατολική Μακεδονία και Θράκη'),('ΠΕ ΡΟΔΟΠΗΣ','Ανατολική Μακεδονία και Θράκη'), ('ΠΕ Α
        ΘΕΣΣΑΛΟΝΙΚΗΣ','Κεντρική Μακεδονία'),('ΠΕ Β ΘΕΣΣΑΛΟΝΙΚΗΣ','Κεντρική Μακεδονία'),('ΠΕ ΗΜΑΘΙΑΣ',
        'Κεντρική Μακεδονία'),('ΠΕ ΚΙΛΚΙΣ','Κεντρική Μακεδονία'),('ΠΕ ΠΕΛΛΑΣ','Κεντρική Μακεδονία'),('ΠΕ
```

Εικόνα 3.5: Δημιουργία πίνακα για τις περιφέρειες και εισαγωγή δεδομένων

Αφού δημιουργηθεί ένα Dataframe με όλα τα αρχεία επόμενο βήμα είναι η σύνδεση με την βάση. Χρησιμοποιώντας τη βιβλιοθήκη mysql.connector εγκαθιστάτε μια σύνδεση με την βάση δεδομένων . Η βάση δεδομένων βρίσκεται σε τοπικό μηχάνημα (localhost) και συνδέεται χρησιμοποιώντας το όνομα χρήστη root, τον κωδικό πρόσβασης και επιλέγοντας τη βάση δεδομένων. Εάν η σύνδεση με τη βάση είναι επιτυχής, τότε θα δημιουργηθεί ένας δρομέας (cursor) βάσης δεδομένων, ο οποίος επιτρέπει την εκτέλεση εντολών SQL. Το συγκεκριμένο script θα τρέχει κάθε φορα που ανανεώνεται η βάση με νέα δεδομένα, δηλαδή κάθε φορά που δημοσιεύονται νέα αρχεία από το υπουργείο, έτσι για να μην χρειάζεται να ρίχνονται οι πίνακες χειροκίνητα ελέγχεται και διαγράφονται (DROP) τυχόν υπάρχοντες με ονόματα Anaplirotes και Periferies για να διασφαλιστεί ότι δεν υπάρχουν συγκρούσεις. Η βάση περιέχει δύο πίνακες, ένα πίνακας με όνομα Anaplirotes και ένας Periferies. Ο στόχος ήταν, για να μην υπάρχει συνεχής επανάληψη της περιφέρειας στον πίνακα Anaplirotes να δημιουργηθεί ένας δεύτερος πίνακας που ονομάζεται Periferies και η στήλη της Διεύθυνσης Εκπαίδευσης στον πίνακα με τους αναπληρωτές να είναι ξένο κλειδί που θα κανει reference την στήλη Dieytynsh στον πίνακα Perifereies. Έτσι, δημιουργείται ένας νέος πίνακας με όνομα Periferies με δύο στήλες: την Dieytynsh και την Periferia. Η στήλη Dieytynsh ορίζεται ως πρωτεύον κλειδί, που σημαίνει ότι κάθε τιμή σε αυτή τη στήλη είναι μοναδική και στην συνέχεια γεμίζει ο πίνακας από όλες τις πιθανές διευθύνσεις εκπαίδευσης και της περιφέρειας που ανήκει η καθεμια.



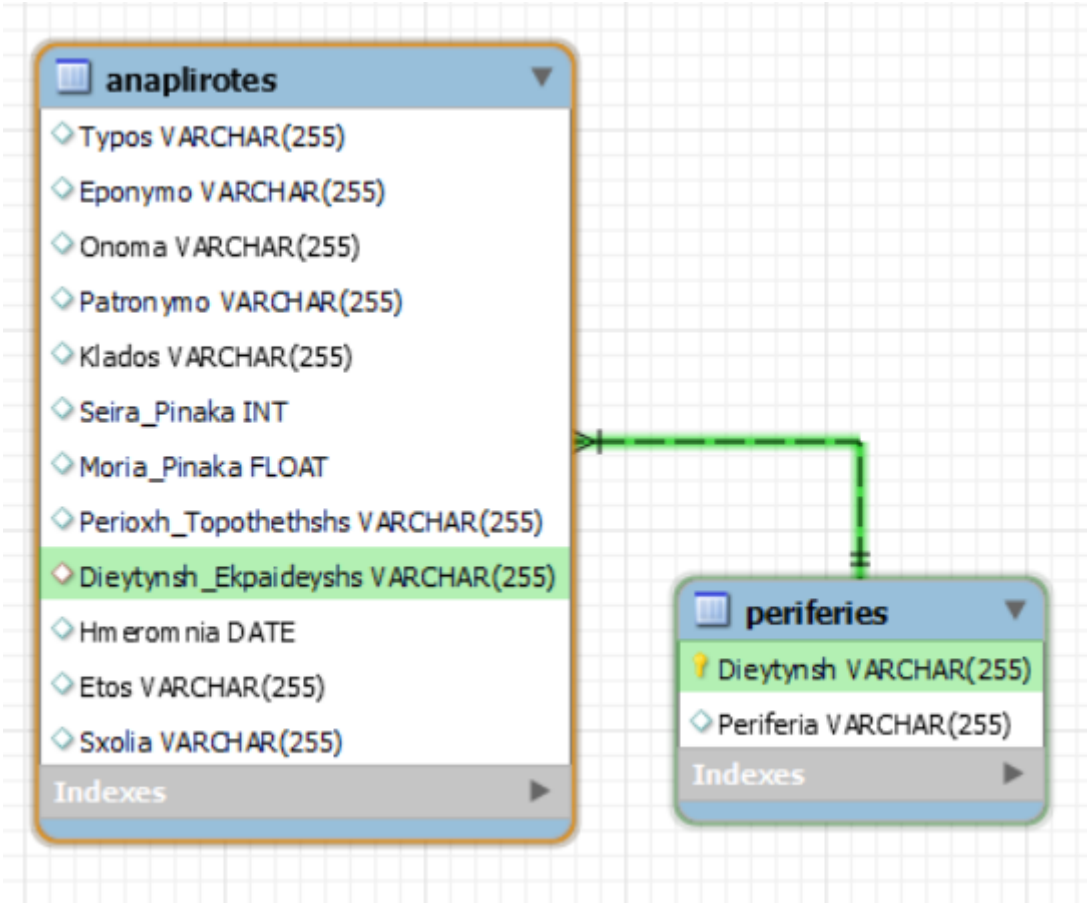
```
        cursor.execute(newSQL)
        cursor.execute(
            "CREATE TABLE Anaplirotes(Typos varchar(255), Eponymo varchar(255), Onoma varchar(255),
            Patronymo varchar(255), Klados varchar(255), Seira_Pinaka int, Moria_Pinaka float null,
            Perioxh_Topothethshs varchar(255), Dieytynsh_Ekpaideyshs varchar(255), Hmeromnia date, Etos
            varchar(255), Sxolia varchar(255),FOREIGN KEY(Dieytynsh_Ekpaideyshs) REFERENCES PERIFERIES
            (Dieytynsh))")
        print("Database is created")
        for i, row in df.iterrows():
            sql = "INSERT INTO Anaplirotes VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"
            cursor.execute(sql, tuple(row))
            print("Record inserted")
            db.commit()


except Error as e:
    print("Error while connecting to MySQL", e)
```

Εικόνα 3.6: Δημιουργία του πίνακα των αναπληρωτών και εισαγωγή δεδομένων

Αφού με το function execute() εκτελεστούν οι εντολές SQL και γεμίσει ο πίνακας Periferies στην συνέχεια δημιουργείται και αυτός των αναπληρωτών. Τέλος, για να εκτελεστούν όλες αυτές οι εντολές SQL και να απεικονιστούν οι αλλαγές στη βάση δεδομένων, το script θα πρέπει επίσης να δεσμεύσει τις αλλαγές χρησιμοποιώντας την db.commit() και να κλείσει τη σύνδεση όταν τελειώσει. Χρησιμοποιείται στο τέλος και το Error της mysql.connector το οποίο διαχειρίζεται τυχόν εξαιρέσεις που μπορεί να προκύψουν κατα την σύνδεση με την βάση.

3.4 Βάση Δεδομένων

Αφού λοιπόν τροποποιηθούν τα αρχικά excel αρχεία στην κατάλληλη δομή, μετατραπούν σε csv και τρέξει το παραπάνω script της python γεμίζει η βάση. Η βάση στην τελική της μορφή περιέχει δύο πίνακες. Αρχικά ο πίνακας των αναπληρωτών (anaplirotes), με τα πεδία να είναι ο τύπος εκπαίδευσης (string), το επώνυμο (string), το πατρώνυμο(string), ο κλάδος (string), η σειρά πίνακα (int), τα μόρια πίνακα (float), η περιοχή τοποθέτησης (string), η διεύθυνση εκπαίδευσης (string), η ημερομηνία (date), το έτος (string) και τα σχόλια (string). Ο δεύτερος πίνακας για τις περιφέρειες (periferies) αποτελείται από δύο πεδία την διεύθυνση (string) και την περιφέρεια (string). Υπενθυμίζεται ότι οι δύο πίνακες συνδέονται μεταξύ τους με την διεύθυνση εκπαίδευσης, η οποία είναι ξένο κλειδί στον πίνακα με τους αναπληρωτές και με όνομα πεδίου Dieytyndh_Ekpaideyshs κάνει reference το κύριο κλειδί στον πίνακα των περιφερειων το πεδίο Dieythynsh. Στο παρακάτω ER διάγραμμα απεικονίζονται οι πίνακες, τα πεδία και η μεταξύ τους σχέση.



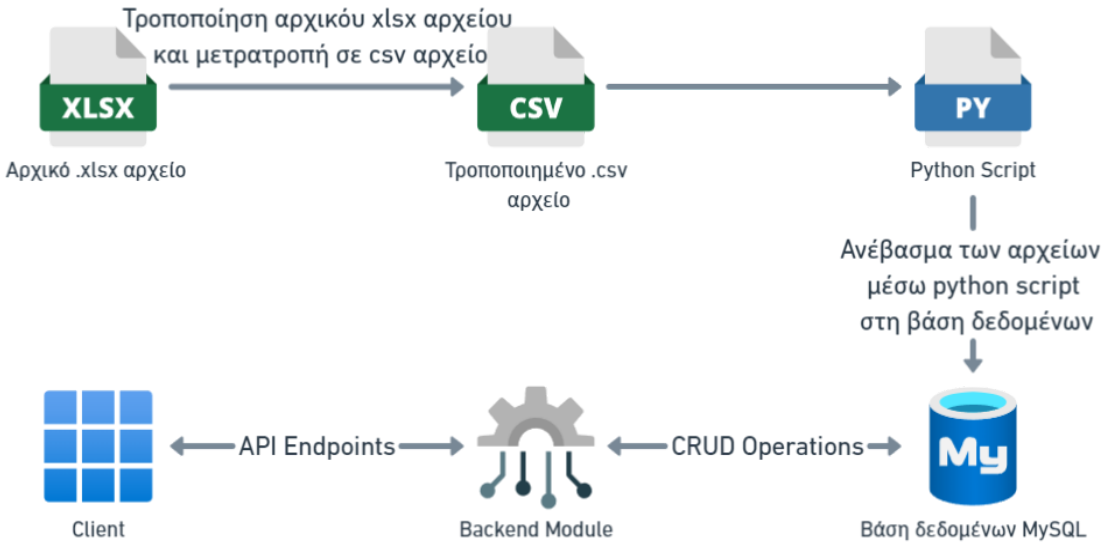
Εικόνα 3.7: Διάγραμμα ER της βάσης

3.5 Οδηγίες για την μετατροπή ενός αρχείου στην επιθυμητή για την βαση μορφη

Όπως αναφέρθηκε και παραπάνω πρέπει να γίνουν κάποιες αλλαγές στο αρχείο για να έρθει αυτό στην κατάλληλη μορφή και να ανέβει αργότερα στην βάση. Πιο συγκεκριμένα τώρα, αν και τα αρχεία πολλές φορές είναι διαφορετικά μεταξύ τους, τα βήματα είναι πάνω-κάτω τα ίδια. Αρχικά διαγράφεται από κάθε αρχείο η στήλη Α/Α και Α/Α ροή καθώς δεν έχουν κάποια αξία για την εφαρμογή και τον χρήστη. Στην συνέχεια, στην στήλη Τύπος όπως προαναφερθηκε οι αποδεκτές τιμές είναι τρεις, “ΓΕΝΙΚΗΣ”, “ΕΙΔΙΚΗΣ” και “ΜΟΥΣΙΚΑ”, οπότε γίνεται μετατροπή όλων των τιμών που είναι υποκατηγορίες των συγκεκριμένων σε αυτές τις τρεις. Οι στήλες που έχουν να κάνουν με ονοματολογίες μένουν όπως είναι καθώς είναι οι μοναδικές στήλες για την ταυτοποίηση του εκπαιδευτικού, αυτές είναι το επώνυμο, το όνομα και το πατρώνυμο. Το μητρώνυμο αν και σημαντική πληροφορία διαγράφεται καθώς υπάρχει σε πολύ μικρο ποσοστό των συνολικών αρχείων. Οι επόμενες δύο στήλες είναι αυτές του κλάδου και της ειδικότητας. Αυτές οι δύο στήλες σχεδόν σε όλα τα αρχεία περιέχουν τις ίδιες τιμές, έτσι διαγράφεται η στήλη της ειδικότητας και κρατιέται αυτή του κλάδου. Η στήλη Πίνακας διαγράφεται καθώς δεν περιέχει αξιοποιήσιμη πληροφορία. Οι στήλες Μόρια Πίνακα και Σειρά Πίνακα κρατιούνται αυτούσιες. Η στήλη Περιοχή Τοποθέτησης αν και κρατιέται δέχεται αλλαγές. Διαγράφεται οποιαδήποτε αναφορά υπάρχει στο ωράριο, η βαθμίδα της εκπαίδευσης (πρωτοβάθμια, δευτεροβάθμια) άλλα και οποιοσδήποτε ειδικός χαρακτήρας όπως οι τόνοι. Σε κάποια αρχεία εκτος απο την αναφορά του ωραρίου στην περιοχή τοποθέτησης υπάρχει και ξεχωριστή στήλη μόνο για το ωράριο η οποία διαγράφεται και αυτή καθώς το ωράριο θα χρησιμοποιηθεί σε άλλη στήλη αργότερα. Στην στήλη Διεύθυνση Εκπαίδευσης γίνονται επίσης αρκετές αλλαγές. Αρχικά διαγράφεται η λέξη «Διεύθυνση» από κάθε κελί. Επίσης διαγράφονται όλες οι τελείες ώστε η βαθμίδα εκπαίδευσης να έχει τελικά την μορφη «ΠΕ» ή «ΔΕ» η οποία μπαίνει πάντα πρώτη στο string (π.χ ΠΕ ΛΑΡΙΣΑΣ) . Τελος διαγράφονται όλοι οι τόνοι και οι διευθύνσεις εκπαίδευσης οι οποίες σπάνε σε τμήματα μετατρέπονται έτσι ώστε πάντα το τμήμα να μπαίνει πριν από την περιοχή (π.χ ΠΕ Α ΘΕΣΣΑΛΟΝΙΚΗΣ). Η στήλη της Περιφέρειας διαγράφεται καθώς η πληροφορίας της υπάρχει σε ξεχωριστό πίνακα. Οι έξτρα στήλες που προστίθενται είναι αυτή της Ημερομηνίας η οποία περιέχει την ημερομηνία που ανακοινώθηκαν οι προσλήψεις και εκδόθηκε το αρχείο (π.χ 2022-17-01), του Έτους που αναγράφεται το έτος που αφορά το συγκεκριμένο αρχείο (π.χ 2021-2022) και η στήλη με όνομα Σχόλια. Στα σχόλια υπάρχει πάντα σαν πρώτη τιμή το ωράριο με τις ονομασίες «ΑΜΩ» στην περίπτωση της μερικής απασχόλησης και «ΑΠΩ» στην περίπτωση της πλήρης απασχόλησης το οποίο σε περίπτωση που ο εκπαιδευτικός είναι μουσικός ακολουθείται από τον χαρακτήρα κόμμα (,) και το μουσικό όργανο το οποίο παίζει (π.χ Κοντραμπάσο).

Κεφάλαιο 4ο: Σχεδίαση και υλοποίηση της εφαρμογής eAnaplirotes

4.1 Κύκλος ζωης των αρχείων και αρχιτεκτονική εφαρμογής



Εικόνα 4.1: Κύκλος ζωης των αρχείων και αρχιτεκτονική εφαρμογής

Παραπάνω εμφανίζεται συνοπτικά ο κύκλος ζωης των αρχείων και η αρχιτεκτονική της εφαρμογής. Το αρχικό .xlsx αρχείο που δίνεται τροποποιείται κατάλληλα ώστε να ανέβει με την επιθυμητή μορφή στη βάση δεδομένων αφού πρώτα μετατραπεί σε αρχείο .csv. Εν συνεχεία, με την χρήση του python script τα csv αυτά ανεβαίνουν στην βάση στην οποία μέσω του Backend γίνονται διάφορα CRUD operations. Χρησιμοποιώντας το Backend μέσω των endpoints ο client τελικά ανακτά και οπτικοποιεί τα δεδομένα.

4.2 Υλοποίηση του Back-End

Το API έχει κατασκευαστεί χρησιμοποιώντας την Node.js η οποία έχει αναλυθεί με λεπτομέρεια στο δεύτερο κεφάλαιο και την Express η οποία είναι ένα framework για Node.js. Για την βάση δεδομένων χρησιμοποιήθηκε η MySQL. Σε αυτό το υποκεφάλαιο θα αναλυθεί η δημιουργία του Backend/API.

```
1  const express = require("express");
2  const mysql = require("mysql");
3  const cors = require("cors");
4  require('dotenv').config();
```

Εικόνα 4.2: Βιβλιοθήκες και εργαλεία του backend

Το αρχικό setup απαιτεί την την εισαγωγή τριών πολύ σημαντικών dependencies:

- Express: Η Exprees είναι το framework που θα χρησιμοποιηθεί για την κατασκευή αυτού του API
- MySQL: Το "mysql" είναι ένα πακέτο που επιτρέπει στη Node.js να διασυνδεθεί με τις βάσεις δεδομένων MySQL.
- CORS: Το CORS (Cross-Origin Resource Sharing) είναι ένα χαρακτηριστικό ασφαλείας που υλοποιείται από τα προγράμματα περιήγησης ιστού για τον έλεγχο των request που υποβάλλονται από διαφορετικές πηγές προέλευσης. Συμπεριλαμβάνοντας το ενδιάμεσο λογισμικό cors, υπάρχει η δυνατότητα να οριστούν κανόνες σχετικά με το ποια origins, μέθοδοι και headers επιτρέπονται για την πρόσβαση σε πόρους στο διακομιστή.

Στο τέλος φορτώνεται η μεταβλητή που περιέχει τον κωδικό πρόσβασης για την βάσης και είναι αποθηκευμένη στο αρχείο .env στο process.env, απαιτείται το πακέτο dotenv και καλείται η μέθοδος config().

```
6   const app = express();
7   const PORT = 5173;
8
9   const connection = mysql.createConnection({
10    host: "localhost",
11    user: "root",
12    password: process.env.DB_PASSWORD,
13    database: "anaplirotes",
14  });
15
16  connection.connect((error) => {
17    if (error) throw error;
18    console.log("Successfully connected to the database");
19  });
```

Εικόνα 4.3: Σύνδεση του API με την βάση δεδομένων

Στην συνέχεια και με την εντολή const app = express(), η express() καλείται για να δημιουργήσει μια νέα εφαρμογή Express. Το αντικείμενο app που επιστρέφεται έχει μεθόδους για τη δρομολόγηση αιτημάτων HTTP, τη διαμόρφωση ενδιάμεσου λογισμικού και για το render των views, μεταξύ άλλων εργασιών. Αυτό το αντικείμενο θα είναι το κύριο δομικό στοιχείο για το setup των endpoints.

Μια σταθερά PORT ορίζεται σε 5173. Αυτός ο αριθμός θύρας χρησιμοποιείται για να πει στην εφαρμογή Express πού να ακούει τις εισερχόμενες αιτήσεις HTTP. Οι αριθμοί θυρών είναι απαραίτητοι για τη δρομολόγηση της κυκλοφορίας του δικτύου.

Η συνάρτηση mysql.createConnection χρησιμοποιείται για τη δημιουργία μιας σύνδεσης με τη βάση δεδομένων MySQL. Το αντικείμενο που περνάει στη createConnection έχει τις ακόλουθες ιδιότητες:

- Host: Το όνομα κεντρικού υπολογιστή της βάσης δεδομένων στην οποία συνδέεστε. Σε αυτή την περίπτωση, είναι το localhost.

- User: Το όνομα χρήστη που χρησιμοποιείται για τον έλεγχο ταυτότητας έναντι της βάσης δεδομένων. Εδώ, είναι ο χρήστης root.
- Password: Ο κωδικός πρόσβασης που σχετίζεται με το όνομα χρήστη για τον έλεγχο ταυτότητας. Είναι καλή πρακτική να μην δίνεται hard-coded ο κωδικός πρόσβασης. Για το σκοπό αυτό χρησιμοποιείται μια μεταβλητή περιβάλλοντος.
- Database: Η συγκεκριμένη βάση δεδομένων με την οποία θα πραγματοποιηθεί η σύνδεση.

Το αντικείμενο connection που επιστρέφεται από αυτή τη μέθοδο ενθυλακώνει τη λειτουργικότητα για την αναζήτηση της βάσης δεδομένων, μεταξύ άλλων εργασιών.

Μια σταθερά PORT ορίζεται σε 5173. Αυτός ο αριθμός θύρας χρησιμοποιείται για να πει στην εφαρμογή Express πού να ακούει τις εισερχόμενες αιτήσεις HTTP. Οι αριθμοί θυρών είναι απαραίτητοι για τη δρομολόγηση της κυκλοφορίας του δικτύου.

Η μέθοδος connect καλείται στο αντικείμενο connection για την αρχικοποίηση της σύνδεσης με τη βάση δεδομένων MySQL. Αυτή η μέθοδος λαμβάνει ως όρισμα μια συνάρτηση επανάκλησης, η οποία θα εκτελεστεί μετά την προσπάθεια δημιουργίας της σύνδεσης. Εάν η σύνδεση αποτύχει, το αντικείμενο error θα συμπληρωθεί και η δήλωση throw error θα τερματίσει την εφαρμογή, εμφανίζοντας το σφάλμα. Εάν από την άλλη η σύνδεση είναι επιτυχής, ένα αρχείο καταγραφής της κονσόλας θα το επιβεβαιώσει με το μήνυμα "Επιτυχής σύνδεση με τη βάση δεδομένων".

```
22  app.use(cors());
23  const bodyParser = require("body-parser");
24  app.use(bodyParser.json());
```

Εικόνα 4.4: Διαμόρφωση του Middleware

Μετά την εισαγωγή της ενότητας cors νωρίτερα στον κώδικα, το middleware υλοποιείται τώρα χρησιμοποιώντας τη συνάρτηση app.use(). Σε αυτή την περίπτωση, η συνάρτηση cors() καλείται με τις προεπιλεγμένες ρυθμίσεις της.

Η συνάρτηση app.use() είναι ο τρόπος με τον οποίο η Express προσθέτει middleware στη στοίβα εφαρμογών. Το middleware είναι συναρτήσεις που έχουν πρόσβαση στα αντικείμενα αίτησης και απόκρισης και στην επόμενη συνάρτηση στον κύκλο αίτησης-απόκρισης της εφαρμογής.

Με την κλήση της app.use(cors()), ουσιαστικά λέει στην εφαρμογή Express να χρησιμοποιήσει το CORS middleware με την προεπιλεγμένη διαμόρφωσή του. Αυτό σημαίνει ότι οποιοσδήποτε πελάτης μπορεί να έχει πρόσβαση σε πόρους από αυτόν τον διακομιστή. Ωστόσο, για εφαρμογές παραγωγής, συνήθως συνιστάται η διαμόρφωση του CORS ώστε να περιορίζεται η πρόσβαση μόνο σε ορισμένες προελεύσεις για λόγους ασφαλείας.

Το middleware body-parser χρησιμοποιείται για την ανάλυση των request bodies σε μια αλυσίδα middleware πριν από τους handlers. Είναι ιδιαίτερα χρήσιμο για την εξαγωγή δεδομένα JSON από το body των εισερχόμενων HTTP POST requests.

Το body-parser module εισάγεται στο project με τη χρήση της συνάρτησης require(). Η συνάρτηση app.use() χρησιμοποιείται για την εισαγωγή του middleware body-parser στην εφαρμογή Express.

Η bodyParser.json() καθορίζει ότι το middleware θα αναλύει μόνο τα ωφέλιμα φορτία JSON. Αυτό σημαίνει ότι οι εισερχόμενες αιτήσεις με τύπους περιεχομένου εκτός του JSON δεν θα αναλύονται.

Όταν ένας πελάτης στέλνει ένα JSON payload στο body του HTTP request, αυτό το middleware θα το αναλύσει ώστε να έχει πρόσβαση σε αυτό μέσω του req.body στους χειριστές των διαδρομών.

Το πλεονέκτημα της χρήσης του bodyParser.json() είναι ότι απλοποιεί την εξαγωγή δεδομένων JSON από το αίτημα POST, εξαλείφοντας την ανάγκη χειροκίνητης ανάλυσης των εισερχόμενων δεδομένων. Αξίζει να αναφερθεί ότι η χρήση τέτοιων αναλυτών σώματος είναι ζωτικής σημασίας για RESTful APIs, όπου η δημιουργία και η τροποποίηση πόρων στη βάση δεδομένων είναι συχνές

Όπως έχει προαναφερθεί η εφαρμογή χωρίζεται σε δύο κομμάτια, το ένα της ονομαστικής αναζήτησης όπου δηλαδη γίνεται αναζήτηση με ένα συγκεκριμένο ονοματεπωνυμο και εκει παρουσιάζονται πληροφορίες για τον συγκεκριμένο αυτόν εκπαιδευτικό. Το άλλο κομμάτι είναι αυτό της γενικής αναζήτησης όπου χρησιμοποιούνται διάφορα φίλτρα για την παραγωγή και την παρουσίαση πληροφορίας. Αρχικά θα παρουσιαστούν τα endpoint που αφορούν το σενάριο της προσωπικής αναζήτησης.

```
26  app.post("/anaplirotes", (req, res) => {
27    const { name } = req.body;
28    const parts = name.split(" ");
29    const whereClauses = parts.map(
30      (part) => `CONCAT(Eponymo, ' ', Onoma) LIKE '%${part}%'`
31    );
32    const whereClause = whereClauses.join(" AND ");
33    const query = `SELECT DISTINCT CONCAT(Eponymo, ' ', Onoma) AS FULLNAME FROM anaplirotes WHERE $
      {whereClause} LIMIT 10`;
34    connection.query(query, (error, result) => {
35      if (error) throw error;
36      const fullNames = result.map((anaplirotes) => anaplirotes.FULLNAME);
37      res.send(fullNames);
38    });
39  });
```

Εικόνα 4.5: Endpoint αναπληρωτών

Ξεκινώντας ο χρήστης εισάγει το όνομα που θέλει να αναζητήσει σε ένα select. Αυτό το select υποστηρίζει autocomplete ώστε να γίνει πιο εύκολα η αναζήτηση και να αποφευχθεί κάποιο λάθος κατα την πληκτρολόγηση. Το endpoint που χρησιμοποιείται για το σερβίρισμα των ονομάτων είναι το /anaplirotes και είναι τύπου POST. Αυτή η διαδρομή έχει ως στόχο την αναζήτηση προφίλ εκπαιδευτικών με βάση τα ονόματά τους και την επιστροφή μιας λίστας ονομάτων που ταιριάζουν με τα κριτήρια αναζήτησης. Πιο αναλυτικά:

Μια διαδρομή POST ορίζεται με τη χρήση της app.post(). Η διαδρομή λαμβάνει μια συνάρτηση ανάκλησης με δύο ορίσματα, το req (το αντικείμενο αίτησης) και το res (το αντικείμενο απάντησης). Το πεδίο name εξάγεται από το σώμα της αίτησης (req.body), το οποίο έχει αναλυθεί ως JSON χάρη στην προηγούμενη ενσωμάτωση του body-parser. Το όνομα μετά χωρίζεται σε επιμέρους τμήματα (υποθέτοντας ότι θα μπορούσε να είναι όνομα, επώνυμο κ.λπ.) και δημιουργείται μια σειρά από WHERE clauses με βάση αυτά τα τμήματα. Στη συνέχεια, αυτά συνδυάζονται σε ένα ενιαίο WHERE clause που χρησιμοποιείται για το φιλτράρισμα του ερωτήματος της βάσης δεδομένων.

Κατασκευάζεται ένα ερώτημα SQL SELECT για την αναζήτηση ονομάτων στον πίνακα anaplirotes που ταιριάζουν με το WHERE clause. Το ερώτημα περιορίζει τα αποτελέσματα σε 10 διαφορετικά ονόματα. Ο λόγος που υλοποιήθηκε με αυτόν τον τρόπο είναι γιατί αν ζητούνταν όλα τα ονόματα για να γεμίσουν τις επιλογές του select, λόγω του μεγάλου πλήθους αυτών θα δημιουργούσε μεγάλο

πρόβλημα καθυστέρησης στην εφαρμογή. Έτσι για να αποφευχθεί αυτό κάθε φορά που ο χρήστης πληκτρολογεί ένα γράμμα μέσα στο input θα καλείται το endpoint αυτό και θα γυρνάει τα πρώτα δέκα matching αποτελέσματα. Εν συνεχεία, το ερώτημα SQL εκτελείται και αν προκύψει σφάλμα κατά την εκτέλεση του ερωτήματος, αυτό απορρίπτεται και η εφαρμογή σταματά. Διαφορετικά, το αποτέλεσμα αντιστοιχίζεται σε έναν πίνακα που ονομάζεται fullNames και αποστέλλεται πίσω ως απάντηση.

Όταν ο χρήστης επιλέξει το όνομα που θέλει να ψάξει, τότε καλείται ένα δεύτερο endpoint σε περίπτωση που υπάρχει συνωνυμία. Λόγω των πολλών ονομάτων που υπάρχουν στην βάση πολλές φορές υπάρχουν συνωνυμίες, για να επιλυθεί αυτό, αφού εντοπιστούν οι συνωνυμίες αυτές ο χρήστης επιλέγει το πατρώνυμο του ατόμου αυτού ώστε να κάνει πιο συγκεκριμένη την αναζήτηση του. Ο τρόπος που υλοποιείται αυτό είναι ο εξής:

```
67  app.get("/onoma/:id", (req, res) => {
68    const { id } = req.params;
69    console.log(id);
70    const newId = id.toUpperCase().split("_");
71    console.log("newId", newId);
72    const query = `SELECT * FROM anaplirotes WHERE anaplirotes.Eponymo='${String(
73      newId[0]
74    )}' AND anaplirotes.Onoma='${String(newId[1])}'`;
75    console.log(query);
76    connection.query(query, (error, result) => {
77      if (error) throw error;
78      res.send(result);
79    });
80  });
```

Εικόνα 4.6: Endpoint για τον έλεγχο περίπτωσης συνώνυμων

Αυτό το μπλοκ κώδικα περιγράφει Express GET route στη διεύθυνση /onoma/:id. Σκοπός του είναι να κάνει ερώτημα στη βάση δεδομένων για να βρει όλες τις εγγραφές που ταιριάζουν με το επιλεγμένο όνομα, για τη διαφοροποίηση μεταξύ ατόμων με το ίδιο όνομα.

Μια διαδρομή GET ορίζεται με τη χρήση της app.get(). Η διαδρομή περιλαμβάνει μια παράμετρο URL :id η οποία θα καταγράφει το επιλεγμένο όνομα από την πλευρά του πελάτη. Η παράμετρος αυτή εξάγεται από τη διεύθυνση URL χρησιμοποιώντας την req.params. Το id μετατρέπεται πρώτα σε κεφαλαία και στη συνέχεια χωρίζεται σε έναν πίνακα newId με βάση τον χαρακτήρα υπογράμμισης (_). Ουσιαστικά χωρίζεται το ονοματεπώνυμο σε όνομα και επίθετο για να γίνει μετα το SQL ερώτημα. Κατασκευάζεται λοιπόν ένα ερώτημα SQL για την αναζήτηση στον πίνακα anaplirotes για εγγραφές όπου το επώνυμο (Eponymo) και το όνομα (Onoma) ταιριάζουν με το αναλυμένο newId. Εκτελείται το SQL ερώτημα , τυχόν σφάλματα που παρουσιάζονται κατά τη διάρκεια του ερωτήματος απορρίπτονται και τα επιτυχημένα αποτελέσματα αποστέλλονται πίσω στον πελάτη στην απόκριση HTTP. Αυτό που επιστρέφεται τελικά είναι τα πατρώνυμα για το επιλεγμένο ονοματεπώνυμο. Αφού επιλέξει το σωστό πατρώνυμο ή αν εξαρχής το όνομα είναι μοναδικό και δεν υπάρχουν συνωνυμίες, τότε κατευθύνεται σε ένα διαφορετικό route με παραμέτρους το ονομα, το επώνυμο και το πατρώνυμο, όπου καλείται τρίτο endpoint για την άντληση δεδομένων. Με την είσοδο στο route αυτό καλείται το εξής endpoint:

```
app.get("/user/:Eponymo/:Onoma/:Patronymo", (req, res) => {
  const { Eponymo, Onoma, Patronymo } = req.params;
  const query = `SELECT * FROM anaplirotes WHERE anaplirotes.eponymo='${Eponymo}' AND anaplirotes.onoma='${Onoma}' AND anaplirotes.Patronymo='${Patronymo}'`;
  connection.query(query, (error, result) => {
    if (error) throw error;
    res.send(result);
  });
});
```

Εικόνα 4.7: Endpoint για την ονομαστική αναζήτηση

Αυτή το μπλοκ κώδικα ορίζει ένα Express GET route στο /user/:Eponymo/:Onoma/:Patronymo. Σκοπό έχει να παρέχει λεπτομερέστερες πληροφορίες για έναν συγκεκριμένο εκπαιδευτικό με βάση τρία αναγνωριστικά, το επώνυμο (Eponymo), το όνομα (Onoma) και το πατρώνυμο (Patronymo). Αρχικά, ορίζεται μια διαδρομή GET με παραμέτρους URL αυτά τα αναγνωριστικά τα οποία μετά εξάγονται από τη διεύθυνση URL χρησιμοποιώντας την req.params. Κατασκευάζεται ένα ερώτημα SQL για την αναζήτηση στον πίνακα anaplirotes για εγγραφές όπου το επώνυμο (Eponymo), το όνομα (Onoma) και το όνομα του πατέρα (Patronymo) ταιριάζουν με τις παραμέτρους από τη διεύθυνση URL. Το ερώτημα εκτελείται μέσω της μεθόδου MySQL query. Εκτελείται το SQL ερώτημα , τυχόν σφάλματα που παρουσιάζονται κατά τη διάρκεια του ερωτήματος απορρίπτονται και τα επιτυχημένα αποτελέσματα αποστέλλονται πίσω στον πελάτη στην απόκριση HTTP.

Αυτά είναι τα τρία endpoint που χρησιμοποιούνται στο σενάριο της προσωπικής αναζήτησης. Το /anaplirotes για την λειτουργικότητα του autocomplete, /onoma/:id για να ελεγχθεί αν υπάρχει κάποια συνωνυμία και το /user/:Eponymo/:Onoma/:Patronymo το οποίο παρέχει όλη την πληροφορία που είναι γνωστή για το συγκεκριμένο όνομα για κάθε χρονιά.

Στο σενάριο της συνολικής αναζήτησης τα πράγματα είναι διαφορετικά, εκεί ο χρήστης χρησιμοποιεί τέσσερα φίλτρα τα οποία φιλτράρουν όλη την πληροφορία και επιστρέφουν συγκεκριμένα αποτελέσματα. Τα φίλτρα αυτά είναι ο τύπος εκπαίδευσης(γενικής, ειδικής και μουσική), τον κλάδο, την περιοχή τοποθέτησης και την διεύθυνση εκπαίδευσης. Αρχικα το πρώτο που πρέπει να γίνει είναι να γεμίσουν αυτα τα select ώστε να μπορεί μετά ο χρήστης να επιλέξει αυτό που θέλει. Αυτό γίνεται χρησιμοποιώντας τέσσερα endpoint, ένα για το κάθε φίλτρο.

```
app.get("/typos", (res) => {
  const query = "SELECT DISTINCT Typos FROM anaplirotes";
  connection.query(query, (error, result) => {
    if (error) throw error;
    const typos = result.map((item) => item.Typos);
    res.send(typos);
  });
});
```

Εικόνα 4.8: Endpoint για την εύρεση όλων των τύπων εκπαίδευσης

Ξεκινώντας με τον τύπο, δημιουργείται μια διαδρομή GET που ακούει στη διεύθυνση /typos. Κατά την πρόσβαση, ο διακομιστής πραγματοποιεί μια αναζήτηση στη βάση δεδομένων για μοναδικές τιμές Typos. Το ερώτημα SQL αποσκοπεί στην ανάκτηση διακριτών τιμών Typos από τον πίνακα anaplirotes. Αυτό εξασφαλίζει ότι οι επιλογές δεν θα έχουν επαναλαμβανόμενες τιμές, καθιστώντας το

περιβάλλον εργασίας του χρήστη καθαρότερο και πιο αποτελεσματικό. Το ερώτημα SQL εκτελείται και μετά την επιτυχή ολοκλήρωσή του, μια map λειτουργία μετατρέπει τον πίνακα αποτελεσμάτων σε έναν νέο πίνακα που περιέχει μόνο τις τιμές Typos. Ο φιλτραρισμένος κατάλογος μοναδικών τιμών Typos αποστέλλεται στη συνέχεια πίσω στον πελάτη ως HTTP response. Εάν προκύψει σφάλμα κατά την εκτέλεση του ερωτήματος, αυτό απορρίπτεται, διακόπτοντας την εκτέλεση του ακόλουθου κώδικα.

Κάτι παρόμοιο γίνεται και για τα άλλα 3 φίλτρα και δεν αξίζει να αναλυθεί παραπάνω, Ενδεικτικά σε εικόνες:

```
91  app.get("/klados", (res) => {
92    const query = "SELECT DISTINCT Klados FROM anaplirotes";
93    connection.query(query, (error, result) => {
94      const klados = result.map((item) => item.Klados);
95      res.send(klados);
96    });
97  });
```

Εικόνα 4.9: Endpoint για την εύρεση όλων των κλάδων

```
99  app.get("/perioxh", (res) => {
100   const query = "SELECT DISTINCT Perioxh_Topothethshs FROM anaplirotes";
101   connection.query(query, (error, result) => {
102     if (error) throw error;
103     const perioxh = result.map((item) => item.Perioxh_Topothethshs);
104     res.send(perioxh);
105   });
106 });
```

Εικόνα 4.10: Endpoint για την εύρεση όλων των περιοχών τοποθέτησης

```
108 app.get("/dieythynsh", (res) => {
109   const query = "SELECT DISTINCT Dieytynsh_Topothethshs FROM anaplirotes";
110   connection.query(query, (error, result) => {
111     if (error) throw error;
112     const dieythynsh = result.map(item => item.Dieytynsh_Topothethshs).filter(value => value !== "-");
113     res.send(dieythynsh);
114   });
115 });
```

Εικόνα 4.11: Endpoint για την εύρεση όλων των διευθύνσεων τοποθέτησης

Στην συνέχεια αφού επιλεκτούν κάποια από τα φίλτρα και γίνει η αναζήτηση ο χρήστης κατευθύνεται σε ένα άλλο route με τις τιμές των φίλτρων σαν παραμέτρους και καλείται το endpoint /search, το οποίοι είναι και η ραχοκοκαλιά του δεύτερου σεναρίου.

Αρχικά αποδιαρθρώνετε το req.query για να ληφθούν τα query parameters typos, klados, perioxh και dieythynsh. Τα φίλτρα αυτά είναι προαιρετικά, που σημαίνει ότι οποιαδήποτε από αυτά μπορεί είτε να υπάρχει είτε να απουσιάζει σε ένα ερώτημα αναζήτησης.

Στη συνέχεια δημιουργείται ένας πίνακας που ονομάζεται conditions, ο οποίος θα περιέχει τα SQL query conditions με βάση τις παραμέτρους του ερωτήματος που παρέχονται από τον χρήστη. Για κάθε

φίλτρο (π.χ. typos, klados, perioxh, dieythynsh), χρησιμοποιείτε μια έκφραση JavaScript που ελέγχει αν το φίλτρο έχει τιμή (δηλαδή, δεν είναι null ή undefined). Εάν έχει, σχηματίζεται η αντίστοιχη συνθήκη SQL χρησιμοποιώντας τη λέξη-κλειδί IN.

Ο χρήστης θα μπορεί να εισάγει μόνο μία τιμή για τα φίλτρα typos, klados, dieythynsh άλλα παραπάνω από ένα για την περιοχή, έτσι ενώ όλα τα άλλα φίλτρα θα είναι ένα string, η περιοχή θα είναι ένας πινακας. Έτσι στην περίπτωση της περιοχής γίνεται μια χαρτογράφηση (map) για κάθε στοιχείο και αν είναι πάνω από ένα ενώνονται με κόμμα. Αυτό έχει ως αποτέλεσμα μια συμβολοσειρά όπως 'value1', 'value2', 'value3' κατάλληλη για τη ρήτρα IN της SQL.

Χρησιμοποιείται το .filter(Boolean) για να εξαλειφθούν οποιεσδήποτε συνθήκες που δεν έχουν αντίστοιχες τιμές στις παραμέτρους του ερωτήματος (και επομένως θα ήταν ψευδείς). Ελέγχεται αν το condition array έχει conditions (δηλαδή, το μήκος του είναι μεγαλύτερο από μηδέν). Αν έχει, ενώνονται με τη λέξη-κλειδί SQL AND και προτάσσονται με τη λέξη-κλειδί SQL WHERE για να σχηματιστεί το πλήρης WHERE clause. Στη συνέχεια ενσωματώνεται η συμβολοσειρά whereClause στο ερώτημα SQL, ολοκληρώνοντας την εντολή SQL για την άντληση εγγραφών με βάση τα φίλτρα που παρέχει ο χρήστης, ταξινομημένων με βάση το πεδίο Etos σε αύξουσα σειρά.

Ο πυρήνας του κώδικα βρίσκεται μέσα στο callback function που παρέχεται στη connection.query(). Σε περίπτωση που η εκτέλεση του ερωτήματος αποτύχει το αντικείμενο σφάλματος θα συμπληρωθεί και μια απάντηση HTTP με status 500 (Internal Server Error) θα σταλεί πίσω στον client.

Στη συνέχεια, αρχικοποιούνται δύο δομές δεδομένων (anaplirotesCount και moriaPinakaStructure). Αυτές οι δομές είναι σχεδιασμένες για να κρατούν τα δεδομένα που απαιτούνται για τη συμπλήρωση γραφημάτων του Chart.js. Στο interface της εφαρμογής για την καλύτερη οπτικοποίηση των δεδομένων χρησιμοποιούνται εκτενώς γραφήματα με την βοήθεια της javascript βιβλιοθήκης Chart.js, οπότε τα αποτελέσματα αποστέλλονται με τέτοιο τρόπο ώστε οι δομή τους είναι κατάλληλη για να εισαχθούν μετά στο component του διαγράμματος. Έτσι και οι δύο δομές έχουν πίνακες labels και datasets, οι οποίοι θα συμπληρωθούν αργότερα με βάση τα αποτελέσματα του ερωτήματος και τα εφαρμοζόμενα φίλτρα. Αρχικοποιείται ένας κενός πίνακας που ονομάζεται labelParts. Στη συνέχεια, ελέγχει αν κάθε φίλτρο (typos, klados και dieythynsh) είναι αληθές (δηλαδή δεν είναι null, undefined ή κενό) και προστίθενται στον πίνακα labelParts. Αυτός ο πίνακας θα χρησιμοποιηθεί αργότερα για την κατασκευή των ετικετών για τα σύνολα δεδομένων του διαγράμματος. Για κάθε περιοχή που υπάρχει δημιουργείται ένα αντικείμενο με δύο ιδιότητες το label το οποίο είναι ένα string που περιέχει την περιοχή και άλλες επιλεγμένες τιμές φίλτρων (όπως typos, klados, dieythynsh), χωρισμένες με κόμμα. Και το data property το οποίο αρχικά είναι ένας άδειος πίνακας που θα συμπληρωθεί στη συνέχεια. Καλείται η συνάρτηση groupByYear η οποία λαμβάνει έναν πίνακα αντικειμένων και επιστρέφει ένα αντικείμενο που ομαδοποιεί αυτά τα δεδομένα ανά έτος, συγκεκριμένα με βάση την ιδιότητα Etos κάθε αντικειμένου, το αποτέλεσμα της αποθηκεύεται στη μεταλητή groupedData. Αν η groupedData έχει τιμές γεμίζει η μεταβλητή anaplirotesCount η οποία τελικά είναι το response του endpoint.

Μία άλλη λειτουργικότητα της εφαρμογής είναι ότι όταν ο χρήστης εισάγει κάποια τιμή και για τον κλάδο και για την περιοχή θα στέλνεται πίσω και ένα δεύτερο αντικείμενο αποτελεσμάτων που αφορά τον ελάχιστο αριθμό μορίων που χρειάζονταν ανα χρονιά για την πρόσληψη του συγκεκριμένου κλάδου στις συγκεκριμένη περιοχή ή περιοχές που έχει εισάγει ο χρήστης. Έτσι στην συνέχεια του κώδικα, εάν υπάρχει κλάδος δημιουργείται ένα ακόμη σύνολο δεδομένων για τα ελάχιστα μόρια πρόσληψης (minMoriaDataset). Τα δεδομένα (minMoriaData) αντλούνται με τη χρήση του getMinMoriaByYear(areaData). Το function getMinMoriaByYear λαμβάνει ουσιαστικά έναν πίνακα

αντικειμένων, καθένα από τα οποία έχει τουλάχιστον δύο ιδιότητες: Etos και Moria_Pinaka. Στη συνέχεια επιστρέφει ένα αντικείμενο όπου κάθε κλειδί είναι ένα μοναδικό έτος (Etos) και η αντίστοιχη τιμή είναι η ελάχιστη Moria_Pinaka για το συγκεκριμένο έτος στον πίνακα δεδομένων. Συμπερασματικά Το /search endpoint επιτρέπει ουσιαστικά μια ευέλικτη απόκριση, προσαρμοσμένη στις παραμέτρους του ερωτήματος του πελάτη και παρέχοντας ως αντάλλαγμα ένα συγκεντρωτικό σύνολο δεδομένων. Επιτρέπει στον πελάτη να κατανοήσει πώς ο αριθμός προσλήψεων αναπληρωτών, καθώς και οι ελάχιστες τιμές μορίων πρόσληψης, ποικίλλουν με βάση πολλαπλούς παράγοντες όπως ο τύπος (typos), ο κλάδος (klados), η περιοχή τοποθέτησης (perioxh) και η διεύθυνση εκπαίδευσης (dieythynsh).

Στο τέλος καλείται η app.listen(PORT, callback) η οποία ουσιαστικά εκκινεί την εφαρμογή Express.js και την προετοιμάζει για να λαμβάνει αιτήσεις HTTP στη συγκεκριμένη θύρα. Αυτή η γραμμή είναι αναγκαία ώστες το backend να ακούει τις εισερχόμενες αιτήσεις του πελάτη.

4.3 Υλοποίηση του Front-end

Το βασικό τεχνολογικό stack που χρησιμοποιήθηκε για την υλοποίηση του front-end κομματιού της εφαρμογής είναι το javascript framework Vue.js, η Typescript η οποία επεκτείνει την Javascript και προσφέρει κάποιες έξτρα δυνατότητες, η Tailwind CSS η οποία είναι ένα CSS framework, η vuetify η οποία είναι ένα component library και η chart.js που χρησιμοποιήθηκε για την οπτικοποίηση των δεδομενων με την μορφή γραφημάτων. Στο παρόν κεφάλαιο θα γίνει ανάλυση του front-end κομματιού της εφαρμογής.

Αρχικά η εφαρμογή από πλευρά interface και δομή κώδικα σπάει σε τρία κομμάτια ή σε τρία views καλύτερα. Στην αρχική σελίδα (Home page) και μία σελίδα για κάθε ένα από τα δύο σενάρια που προαναφέρθηκαν, αυτό της ονομαστικής αναζήτησης με την χρήση ονοματεπώνυμου και αυτό της γενικής αναζήτησης με την χρήση φίλτρων.

Ξεκινώντας με την αρχική σελίδα και το πρώτο σενάριο , γίνεται η χρήση του Hero component, το οποίο περιέχει το structure με τα inputs των δύο σεναρίων.

```
148     <div v-if="!toggleState" class="flex flex-col justify-center items-center">
149       <v-autocomplete
150         v-model="name"
151         hide-no-data
152         :custom-filter="customFilter"
153         :loading="loading"
154         class="bg-white w-96"
155         label="Αναζητήστε το ονοματεπώνυμό σας"
156         :items="nameArray"
157         @update:search="getNames"
158       />
159       <v-btn class="w-96" :disabled="!userInput" @click="getInfo">
160         Αναζητηση
161       </v-btn>
162     </div>
```

Εικόνα 4.12: Υλοποίηση του autocomplete

Ενδεικτικά αυτό είναι το input που χρησιμοποιήθηκε για το σενάριο της ονομαστικής αναζήτησης. Γίνεται χρήση ενός autocomplete component της Vuetify, ο λόγος που χρησιμοποιήθηκε το

autocomplete και όχι ένα απλό input είναι για την ευκολότερη και γρηγορότερη εύρεση ενός ονόματος από τον χρήστη και την αποφυγή τυπογραφικών λαθών. Στο component εισάγονται διάφορα options, απο πιο απλά όπως το placeholder, μέχρι πιο περίπλοκα όπως λογική για το filtering των αποτελεσμάτων.

Ο τρόπος που δουλεύει το autocomplete και εξηγήθηκε παραπάνω στην ανάλυση του backend είναι με την χρήση του /getInfo endpoint κάθε φορά που εισάγει ο χρήστης κάποιον χαρακτήρα, αυτό υλοποιείται χρησιμοποιώντας το update@search της vuetify, το οποίο κάθε φορά που αλλάζει το input του autocomplete εκτελεί μια ενέργεια, στην προκειμένη περίπτωση, καλεί την συνάρτηση getNames.

```
51  function getNames(names: string) {
52    if (names && names.length > 0)
53      name.value = names.normalize('NFD').replace(/[\u0300-\u036F]/g, '')
54    loading.value = true
55    axios
56      .post(
57        'http://localhost:5173/anaplirotes',
58        {
59          name: name.value,
60        },
61        {
62          headers: {
63            'Content-Type': 'application/json',
64          },
65        },
66      )
67      .then((response) => {
68        nameArray.value = response.data
69        loading.value = false
70      })
71      .catch((error) => {
72        // eslint-disable-next-line no-console
73        console.log(error)
74      })
75  }
```

Εικόνα 4.13: Κώδικας για την ανάκτηση των ονομάτων

Η getNames είναι υπεύθυνη για να επιστρέφει τα κατάλληλα ονόματα ανάλογα το input του χρήστη. Πιο συγκεκριμένα δέχεται σαν παράμετρο ένα string. Πρώτο βήμα είναι η αφαίρεση ειδικών χαρακτήρων με την χρήση του normalize αφαιρούνται οι ειδικοί χαρακτήρες, όπως παραδείγματος χάρη οι τόνοι και αντικαθίστανται με μια κενή συμβολοσειρά. Αυτό γίνεται χρησιμοποιώντας μια κανονική έκφραση που ταιριάζει με χαρακτήρες Unicode που χρησιμοποιούνται για τόνους. Στη συνέχεια, αφού έρθει το όνομα στην κατάλληλη μορφή γίνεται μια αίτηση POST χρησιμοποιώντας τη βιβλιοθήκη Axios η οποία απευθύνεται στο endpoint /anaplirotes, τα δεδομένα στέλνονται σε μορφή JSON. Σε περίπτωση που το POST αίτημα πραγματοποιηθεί επιτυχώς τα δεδομένα που επιστρέφονται γεμίζουν το nameArray, το οποίο με την σειρά του τροφοδοτεί το autocomplete component. Αυτή είναι η και γενική λειτουργικότητα του autocomplete, ο χρήστης τροφοδοτεί ένα input και σε κάθε εισαγωγή χαρακτήρα επιστρέφονται τα ταιριαστά αποτελέσματα που γεμίζουν τα options του select.

Αφού επιλεγεί το όνομα, πατώντας το κουμπί αναζήτηση καλείται η getInfo()

```
77  function getInfo() {
78    const infoName = name.value.replace(/ /g, '_')
79    axios
80      .get(`http://localhost:5173/onoma/${infoName}`)
81      .then((response) => {
82        filteredArray.value = groupData(response.data as Anapliroths)
83
84        if (filteredArray.value.length > 1) {
85          synonymyModal.value = true
86        }
87        else {
88          const { Eponymo, Onoma, Patronymo } = filteredArray.value[0][0]
89          router.push({
90            name: 'User',
91            params: {
92              Eponymo,
93              Onoma,
94              Patronymo,
95            },
96          })
97        }
98      })
99      .catch((error) => {
100       // eslint-disable-next-line no-console
101       console.log(error)
102     })
103 }
```

Εικόνα 4.14: Κώδικας για τον έλεγχο τυχόν συνωνυμίας

Ο λόγος που χρησιμοποιείται η getInfo() είναι για να αποσαφηνιστεί αν το όνομα που επιλέχθηκε είναι μοναδικό, καθώς λόγω του μεγάλου πλήθους των ονομάτων που υπάρχουν στην βάση υπάρχουν και πολλές συνωνυμίες. Αρχικά αντικαθίστανται απο την συμβολοσειρά του ονόματος τυχόν κενά με την κάτω παύλα ώστε να έρθουν στην σωστή μορφή για να γίνει αναζήτηση στην βάση. Η αναζήτηση γίνεται χρησιμοποιώντας το endpoint /onoma που αναλύθηκε προηγουμένως με την χρήση του ονόματος. Το HTTP GET αίτημα πραγματοποιείται χρησιμοποιώντας το Axios. Το Axios είναι ένας δημοφιλής HTTP client για JavaScript που βασίζεται σε promises. Αν το αίτημα είναι επιτυχημένο τότε με την χρήση της συνάρτησης groupData() αποθηκεύονται τα αποτελέσματα σε μια μεταβλητή.

```
33  function groupData(data: Anapliroths) {
34    const result: Anapliroths[] = []
35    const groups: { [key: string]: Anapliroths } = {}
36
37    data.forEach((item) => {
38      const patronymo = item.Patronymo
39
40      if (!groups[patronymo]) {
41        groups[patronymo] = []
42        result.push(groups[patronymo])
43      }
44
45      groups[patronymo].push(item)
46    })
47
48    return result
49  }
```

Εικόνα 4.15: Ομαδοποίηση των δεδομένων με βάση το πατρώνυμο

Η groupData παίρνει τα αποτελέσματα που επιστράφηκαν από το αίτημα προηγουμένως και δημιουργεί μια νέα δομή με βάση το πατρώνυμο. Πιο συγκεκριμένα, για να αποσαφηνίσουμε τελικά αν το όνομα είναι μοναδικό κοιτάμε το πατρώνυμο. Αν υπάρχει το ίδιο ονοματεπώνυμο στην βάση άλλα με άλλο πατρώνυμο σημαίνει ότι υπάρχουν διαφορετικά άτομα με το ίδιο ονοματεπώνυμο. Η δομή λοιπόν που δημιουργεί η groupData() είναι ένας πίνακας που αποτελείται από πίνακες για κάθε πατρώνυμο, δηλαδή για το όνομα που έχει επιλεχθεί προηγουμένως θα δημιουργηθεί τελικά ένας πίνακας που περιέχει με την σειρά του έναν πίνακα για το κάθε πατρώνυμο, ο οποιός αυτός εσωτερικός πίνακας περιέχει όλη την πληροφορία που υπάρχει για το συγκεκριμένο όνομα που επιλέχθηκε για ένα συγκεκριμένο πατρώνυμο. Ετσι συνεχίζοντας με την getInfo(), σε περίπτωση που ο πίνακας έχει πάνω από μία καταχώρηση, δηλαδή πάνω από ένα πατρώνυμο υπάρχει συνωνυμία. Αν υπάρχει μόνο ένα πατρώνυμο, δηλαδή δεν υπάρχει συνωνυμία και το όνομα είναι μοναδικό τότε η συνάρτηση ανακτά τα πεδία Eponymo, Onoma και Patronymo από την πρώτη καταχώρηση στο filteredArray και στη συνέχεια γίνεται δρομολόγηση στη σελίδα 'User' χρησιμοποιώντας το Vue Router, περνώντας αυτά τα πεδία ως παραμέτρους. Επιστρέφοντας στο σενάριο της συνονιμίας, αν αυτή υπάρχει τότε ανοίγει ένα dialog component (modal). Στο component αυτό περνάει παραμετρικά το filterredArray και δίνει την επιλογή στο χρήστη να επιλέξει κάποιο πατρώνυμο, αφού το επιλέξει τότε η διαδικασία είναι παρόμοια, ανακτώνται τα πεδία Eponymo, Onoma και Patronymo και στη συνέχεια γίνεται δρομολόγηση στη σελίδα 'User' χρησιμοποιώντας το Vue Router, περνώντας αυτά τα πεδία ως παραμέτρους. Συμπερασματικά, η συνάρτηση getInfo() αντλεί δεδομένα που σχετίζονται με ένα όνομα από έναν backend διακομιστή,επεξεργάζεται και φιλτράρει αυτά τα δεδομένα. Ανάλογα με το μήκος του πίνακα των φιλτραρισμένων δεδομένων, είτε ανοίγει ένα modal για να επιλέξει ο χρήστης συγκεκριμένο πατρώνυμο και μετά να δρομολογηθεί σε μια διαφορετική σελίδα ή εφόσος έχει επιλέξει κάποιο μοναδικό ονοματεπώνυμο δρομολογείται κατευθείαν.

Η σελίδα στην οποία δρομολογείται ονομάζεται User και είναι στην ουσία το view που βλέπει ο χρήστης μετά την ονομαστική αναζήτηση, όπου και γίνεται η οπτικοποίηση των δεδομένων.

```
48  const { Eponymo, Onoma, Patronymo } = route.params
49
50  onMounted(async () => {
51    await axios
52      .get<Anapliroths>(`http://localhost:5173/user/${Eponymo}/${Onoma}/${Patronymo}`)
53      .then((response) => {
54        userData.value = response.data
55        for (let i = 0; i < response.data.length; i++) {
56          chartLabels.value.push(response.data[i].Etos)
57          moriaPinaka.value.push(response.data[i].Moria_Pinaka)
58          seiraPinaka.value.push(response.data[i].Seira_Pinaka)
59        }
60      })
61      .catch((error) => {
62      // eslint-disable-next-line no-console
63        console.log(error)
64      })
65    data.value = moriaPinaka.value
66  })
```

Εικόνα 4.16: Αντληση δεδομένων ενός εκπαιδευτικού

Πρώτο βήμα με την είσοδο στην σελίδα είναι να αντληθούν μέσω της UseRoute() συνάρτησης της Vue οι παράμετροι Eponymo, Onoma και Patronymo από το url και μέσα στην onMounted να γίνει η κλήση του API. Η onMounted είναι ένα lifecycle hook της Vue 3 που εκτελείται μετά την τοποθέτηση του στοιχείου στο DOM. Η πραγματοποίηση κλήσεων API σε αυτό το hook είναι γενικά μια κοινή πρακτική. Γίνεται λοιπόν μια αίτηση GET με τη χρήση του Axios στη /user που περιλαμβάνει το Eponymo, το Onoma και το Patronymo ως τμήματα. Υπενθυμίζεται ότι στην εφαρμογή η οπτικοποίηση των δεδομένων γίνεται και με την χρήση γραφημάτων μέσω της της Javascript βιβλιοθήκης chart.js. Για αυτόν το λόγο έχει γίνει προεργασία ήδη στην υλοποίηση του backend να έρχεται μια απάντηση με κατάλληλη δομή ώστε να "κουμπώνει" μετά στο γράφημα. Έτσι σε περίπτωση που το αίτημα πραγματοποιηθεί επιτυχώς γεμίζουν οι μεταβλητές chartLabels με τα έτη για τα οποία υπάρχουν δεδομένα, τα οποία θα είναι και τα labels του γραφήματος και μια μεταβλητή για τα μόρια πίνακα και τη σειρά πίνακα. Όπως προαναφέρθηκε τα δεδομένα που μετέπειτα θα "τροφοδοτήσουν" το γράφημα πρέπει να έχουν μια συγκεκριμένη δομή.

```
import {
  CategoryScale,
  Chart as ChartJS,
  Legend,
  LineElement,
  LinearScale,
  PointElement,
  Title,
  Tooltip,
} from 'chart.js'
import { Line } from 'vue-chartjs'

ChartJS.register(
  CategoryScale,
  LinearScale,
  PointElement,
  LineElement,
  Title,
  Tooltip,
  Legend,
)
```

Εικόνα 4.17: Κτίσιμο του γραφήματος

```
const chartData = computed(() => ({
  labels: chartLabels.value,
  datasets: [{
    label: select.value,
    data: data.value,
    borderColor: 'rgb(8 145 178)',
    backgroundColor: 'rgb(8 145 178)',
  }],
}))

const chartOptions = ref({
  responsive: true,
})
```

Εικόνα 4.18: Τα options του γραφήματος

Αρχικα, στις παραπάνω εικόνες παρουσιάζεται το πως εισάγεται και χτίζεται το γράφημα της chart.js. Εισάγονται διάφορα modules της βιβλιοθήκη όπως το CategoryScale και το LinearScale που είναι τύποι κλιμάκων που χρησιμοποιούνται στους άξονες. Τα PointElement και LineElement χρησιμοποιούνται για τον καθορισμό των ιδιοτήτων των σημείων και των γραμμών στο διάγραμμα. Τα Title, Tooltip και Legend είναι στοιχεία που προσθέτουν επιπλέον λειτουργικότητα και διακόσμηση στο γράφημα, όπως προσθήκη τίτλων, tooltips και legends. Επίσης εισάγεται το Line component από το vue-chartjs, το οποίο είναι ένας wrapper γύρω από τον τύπο διαγράμματος Line που παρέχεται από το Chart.js. Στην συνέχεια με τη χρήση του ChartJS.register καταχωρούνται τα

εισαγόμενα components στο γράφημα. Η computed μεταβλητή chartData είναι αυτή που τροφοδοτεί το γράφημα. Αυτή συντάσσει ένα αντικείμενο που περιέχει όλες τις πληροφορίες που απαιτούνται για τη σχεδίαση ενός γραμμικού διαγράμματος. Περιλαμβάνει τις ετικέτες για τον άξονα (chartLabels.value), ένα ενιαίο σύνολο δεδομένων που περιλαμβάνει μια ετικέτα (select.value), τα πραγματικά δεδομένα (data.value) και πληροφορίες μορφοποίησης του γραφήματος (borderColor και backgroundColor). Η chartData μεταβλητή ορίζεται ως computed χρησιμοποιώντας τη συνάρτηση computed της Vue. Τα computed properties στη Vue είναι αντιδραστικές παραγώγους κάποιων άλλων αντιδραστικών ιδιοτήτων. Όταν κάποια από τα dependencies της αλλάξει, τότε αυτή θα επαναυπολογιστει.. Αυτό την καθιστά ιδανική για περιπτώσεις όπου μια τιμή προέρχεται από μία ή περισσότερες αντιδραστικές πηγές. Η χρήση computed properties διασφαλίζει ότι κάθε φορά που αλλάζουν οι ετικέτες chartLabels, η επιλογή ή τα δεδομένα, το chartData θα ενημερώνεται αυτόματα, προκαλώντας την εκ νέου εμφάνιση του γραφήματος με τα νέα δεδομένα. Αυτό αξιοποιεί το σύστημα αντιδραστικότητας της Vue για να διατηρεί το UI συγχρονισμένο με το μοντέλο δεδομένων. Εκτός από τα δεδομένα στο component του γραφήματος περνάνε και διάφορα options. Στην προκειμένη περίπτωση περνάει μόνο το boolean option responsive με τιμή true που κάνει responsive το γράφημα.

```
83        <div class="w-full md:w-3/4">
84          <Line
85            id="my-chart-id"
86            :options="chartOptions"
87            :data="chartData"
88          />
89        </div>
90        <div class="flex flex-col mb-10 md:mb-0 items-center md:w-1/4 mt-7 pr-5">
91          <div>
92            <v-select
93              v-model="select"
94              label="Select"
95              :items="['Σειρά Πίνακα', 'Μόρια Πίνακα']"
96              class="w-[20rem] h-20"
97              @update:model-value="changeChartValues"
98            />
99          </div>
100         <div class="flex items-center">
101           <v-card title="Στοιχεία Αναπληρωτή" width="320">
102             <v-card-text>
103               <p>
104                 Επώνυμο: {{ route.params.Eponymo }}
105               </p>
106               <p>Όνομα: {{ route.params.Onoma }}</p>
107               <p>Πατρώνυμο: {{ route.params.Patronymo }}</p>
108             </v-card-text>
```

Εικόνα 4.19: Κώδικας του interface της ονομαστικής αναζήτησης

Στην παραπάνω εικόνα φαίνεται και πως περνάνε εν τέλει τα options αυτά στο γράφημα. Το Line είναι ένα vue-chartjs component στο οποίο περνιούνται τα options και τα δεδομενα μέσω props. Φαίνεται επίσης και ένα select component της vuetify το οποίο αλλάξει τα δεδομένα που παρουσιάζονται από το γράφημα. Όπως αναφέρθηκε και πρότερα, όταν γυρνάνε τα δεδομένα από το endpoint /user τότε αποθηκεύονται τα μόρια πίνακα και η σειρά πίνακα σε δύο διαφορετικές μεταβλητές. Με τη χρήση του select γίνεται εναλλαγή των δύο αυτών μεταβλητών με αποτέλεσμα να ανανεώνεται δυναμικά και

το γράφημα. Επίσης στον κώδικα φαίνεται και η χρήση ενός ακόμη component της vuetify, μιας κάρτας που έχει τις βασικές πληροφορίες του χρήστη, όπως το όνομα, το επώνυμο και το πατρώνυμο.

```
144        <v-select
145          v-model="typos"
146          class="w-96 px-5"
147          clearable
148          label="Τύπος"
149          :items="typoi"
150        />
151        <v-select
152          v-model="klados"
153          class="w-96 px-5"
154          clearable
155          label="Κλάδος"
156          :items="kladoi"
157        />
158        <v-select
159          v-model="perioxh"
160          class="w-96 px-5"
161          clearable
162          chips
163          label="Περιοχή Τοποθέτησης"
164          :items="perioxes"
165          multiple
166        />
167        <v-select
168          v-model="dieythynsh"
169          class="w-96 px-5"
170          clearable
171          label="Διεύθυνση Τοποθέτησης"
172          :items="dieythynseis"
```

Εικόνα 4.20: Κώδικας του interface της αναζήτησης με τη χρήση κριτηρίων

Επιστρέφοντας στην αρχική σελίδα και στο δεύτερο σενάριο της γενικης αναζήτησης με την χρήση φίλτρων, όπως υποδεικνύεται παραπάνω υπάρχουν τέσσερα select για τέσσερα φίλτρα, τον τύπο, τον κλάδο, την περιοχή τοποθέτησης και την διεύθυνση εκπαίδευσης. Χρησιμοποιήθηκε και πάλι το select component της vuetify. Κάθε ένα είναι δεσμευμένο και με μια μεταβλητή μέσω του v-model. Στην

περίπτωση της περιοχής υπάρχει και ένα παραπάνω option, το multiple, όπου επιτρέπεται να είναι επιλεγμένες πάνω από μία τιμές.

```
112  axios.get('http://localhost:5173/typos').then((response) => {
113    typoi.value = response.data
114  })
115  axios.get('http://localhost:5173/klados').then((response) => {
116    kladoi.value = response.data
117  })
118  axios.get('http://localhost:5173/perioxh').then((response) => {
119    perioxes.value = response.data
120  })
121  axios.get('http://localhost:5173/dieythynsh').then((response) => {
122    dieythynseis.value = response.data
123  })
```

Εικόνα 4.21: Endpoint που γεμίζουν τα φίλτρα

Τα options του κάθε select γεμίζουν χρησιμοποιώντας ένα endpoint για το καθένα, τα αποτελέσματα αυτών αποθηκεύονται σε τέσσερις διαφορετικές μεταβλητές οι οποίες περνάνε ως prop στα select components. Για να γίνει η αναζήτηση πρέπει να έχει έστω και ένα από αυτά τα select τιμή, σε αυτή την περίπτωση το κουμπί της αναζήτησης ενεργοποιείται και το κλικ του ο χρήστης δρομολογείται σε μια άλλη σελίδα η οποία έχει ως παραμέτρους τα οποιαδήποτε φίλτρα εισήχθησαν πριν απο τον χρήστη.

```
async function performSearch() {
  const params = route.query

  await axios.get('http://localhost:5173/search', { params })
    .then((response) => {
      chartLabels.value = response.data.anaplirotesCount.labels
      data.value = response.data.anaplirotesCount.datasets.map(
        (item: chartDataInterface) => {
          return {
            label: item.label,
            data: item.data,
          }
        },
      )
      chartLabelsMin.value = response.data?.minMoria?.labels
      dataMin.value = response.data?.minMoria?.datasets.map(
        (item: chartDataInterface) => {
          return {
            label: item.label,
            data: item.data,
          }
        },
      )
      dataArrived.value = true
    })
    .catch((error) => {
      console.error(error)
    })
}
```

Εικόνα 4.22: Κώδικας για την αναζήτηση με φίλτρα

Το πρώτο πράγμα που γίνεται και πάλι στην OnMounted είναι η εκτέλεση της performSearch() συνάρτησης. Αρχικά μέσω της UseRoute() συνάρτησης της Vue αντλούνται οι παράμετροι typos, klados, perioxh, dieythynsh, όταν αυτές υπάρχουν καθώς είναι όλες προαιρετικές. Στη συνέχεια γίνεται μια αίτηση τύπου GET με τη χρήση του Axios στη /search που περιλαμβάνει τις επιλεγμένες παραμέτρους ως τμήματα. Υπενθυμίζεται και πάλι ότι στην εφαρμογή η οπτικοποίηση των δεδομένων γίνεται και με την χρήση γραφημάτων μέσω της της Javascript βιβλιοθήκης chart.js. Για αυτόν το λόγο έχει γίνει προεργασία ήδη στην υλοποίηση του backend να έρχεται μια απάντηση με κατάλληλη δομή ώστε να "κουμπώνει" μετά στο γράφημα. Έτσι αρχικά σε περίπτωση που το αίτημα πραγματοποιηθεί επιτυχώς γεμίζουν οι μεταβλητές chartLabels με τα έτη. Η υλοποίηση σε αυτό το σενάριο είναι λίγο διαφορετική. Επειδή ο χρήστης μπορεί να βάλει πάνω από μία περιοχή δεν μπορεί να καθοριστεί από την αρχή πόσες μεταβλητές πρέπει να δημιουργηθούν για να αποθηκεύονται οι πληροφορίες της κάθε περιοχής, ούτε είναι καλή επιλογή να υπάρχει ένα select που μπορεί να έχει μέχρι και είκοσι περιοχές ώστε να επιλέγονται μία-μία οι μεταβλητές/περιοχές ώστε να εμφανίζεται η κατάλληλη. Έτσι στο συγκεκριμένο σενάριο υπάρχει ένα multiline γράφημα, δηλαδή ένα γράφημα με πολλές γραμμές, μία για κάθε περιοχή. Αφού λοιπον γεμίσουν τα labels του γραφήματος γεμίζουν και με κατάλληλο τρόπο και τα δεδομένα που θα το τροφοδοτήσουν δημιουργώντας ένα νέο κατάλληλο αντικείμενο. Υπάρχει επίσης και μία ακόμη περίπτωση, όπως περιγράφηκε και στην ανάλυση του backend σε περίπτωση που ο χρήστης εισάγει κάποιον κλάδο ταυτόχρονα με κάποια περιοχή τότε εκτός από το γράφημα με τον αριθμό των προσλήψεων υπάρχει και ένα δεύτερο γράφημα με τα

ελάχιστα μόρια που προσλήφθηκε ένα αναπληρωτής αυτού του κλαδου για την συγκεκριμένη περιοχή. Σε αυτή λοιπόν την περίπτωση η απάντηση του /search έχει αυτή την έξτρα πληροφορία η οποία αντλείται και διαμορφώνεται ακριβώς με τον ίδιο τρόπο όπως και με την πληροφορία του αριθμού των προσλήψεων και προβάλλεται σε δεύτερο γράφημα.

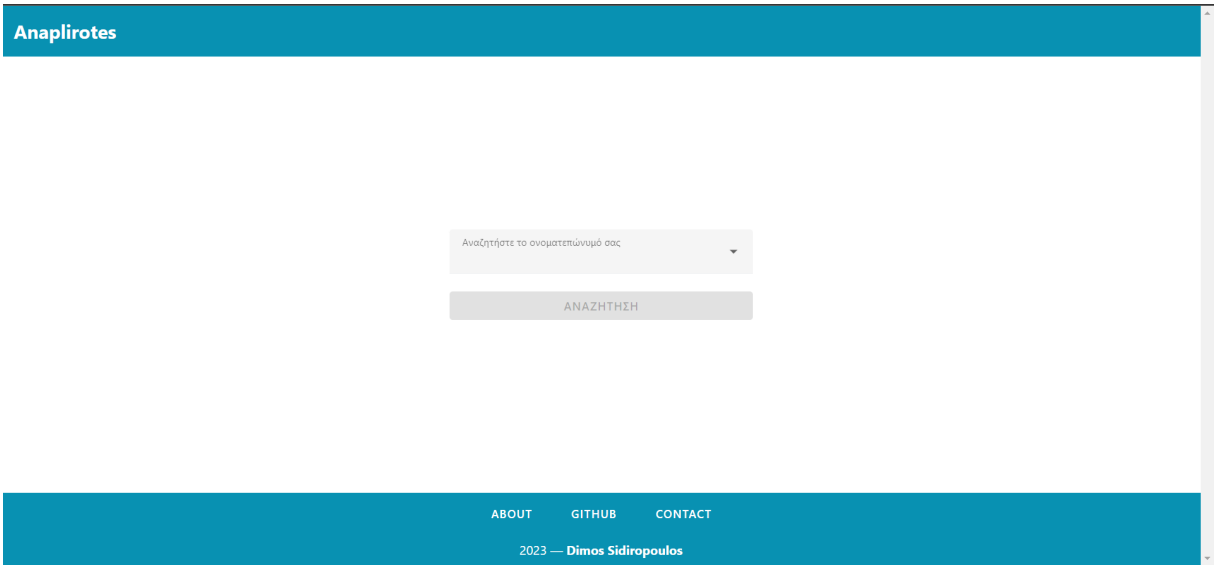
4.4 Github Repository

Ο κώδικας της web εφαρμογής και του web API είναι ανεβασμένος σε ένα github repository το οποίο βρίσκεται στο https://github.com/DimosSidiropoulos/eAnaplirotes. Ο τρόπος που δομείται το repository είναι με δύο φακέλους, έναν για το frontend και ένας για το backend.

Κεφάλαιο 5ο: Παρουσίαση της εφαρμογής eAnaplirotes

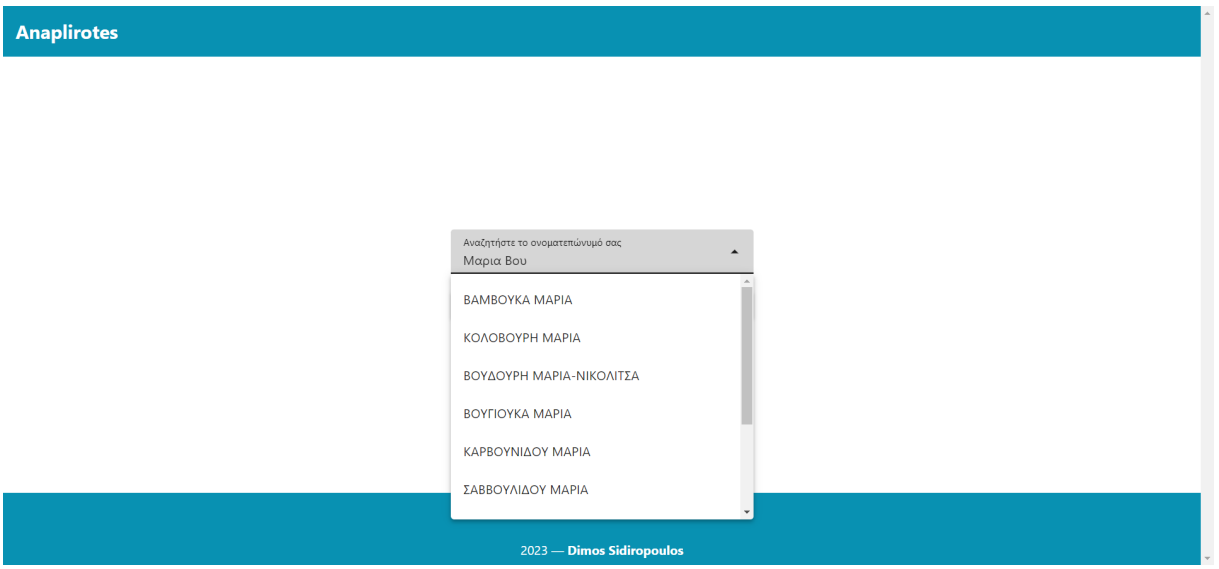
Σε αυτό το κεφάλαιο θα αναλυθεί το τελικό προϊόν, αυτό της web εφαρμογής. Όπως έχει προαναφερθεί η εφαρμογή έχει δύο κύρια σενάρια. Αυτό της ονομαστικής αναζήτησης και αυτό της γενικής.

5.1 Αναζήτηση βάσει ονόματος



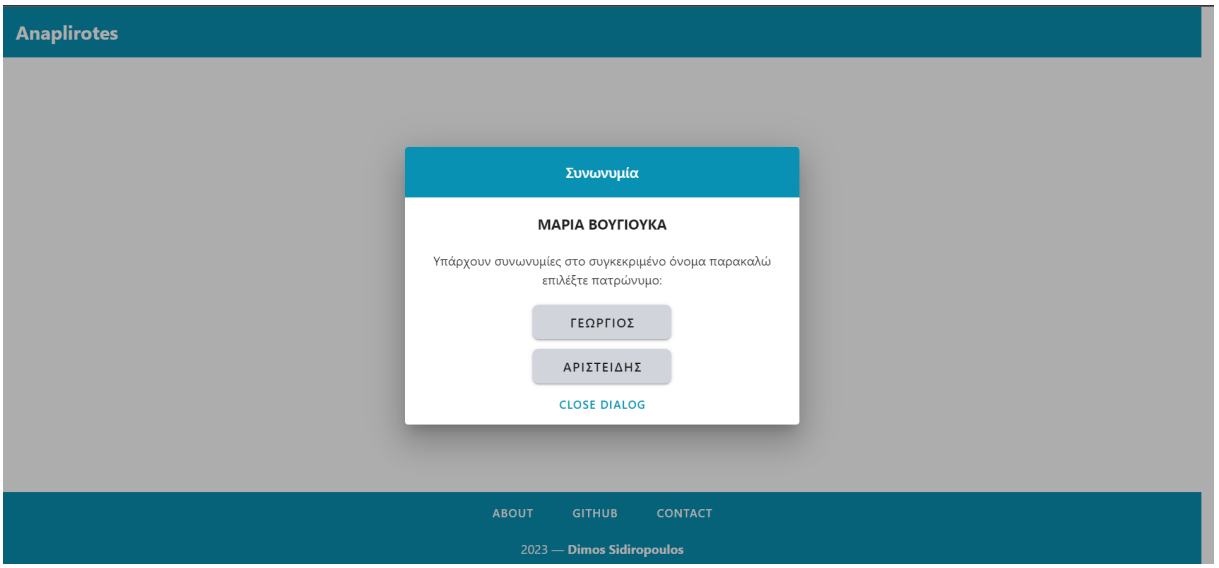
Εικόνα 5.1: Interface της ονομαστικής αναζήτησης

Στο σενάριο της ονομαστικής αναζήτησης, το interface με το οποίο αλληλεπιδρά ο χρήστης είναι ένα autocomplete component και ένα κουμπί για να πραγματοποιήσει ο χρήστης την αναζήτηση. Το κουμπί της αναζήτησης ενεργοποιείται μόνο εάν ο χρήστης έχει εισάγει κάποιον χαρακτήρα στο input. Αφού λοιπόν ο χρήστης ξεκινά να εισάγει χαρακτήρες, ξεκινά και η λειτουργικότητα του autocomplete.



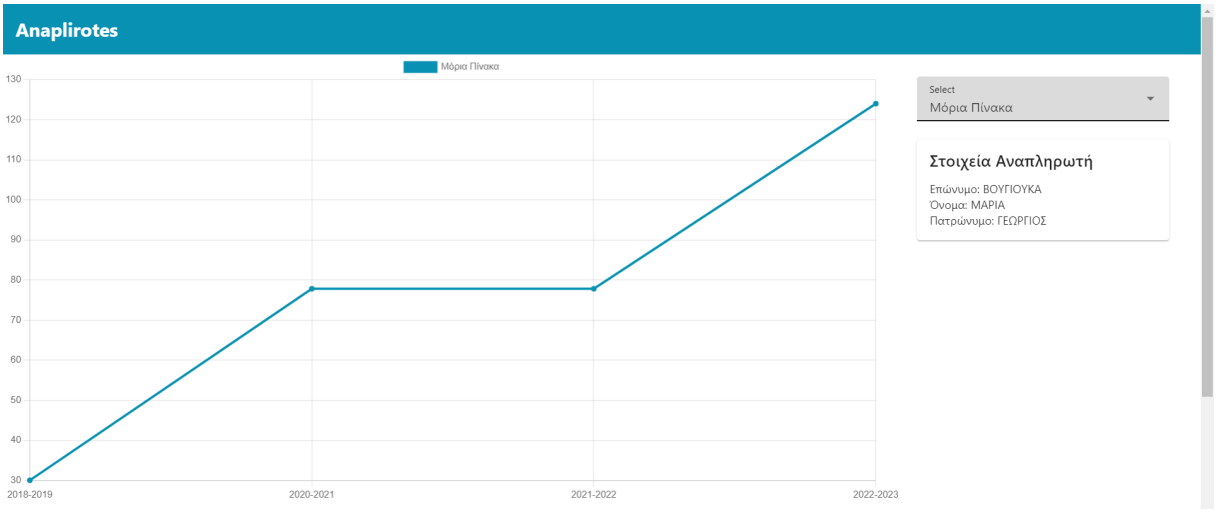
Εικόνα 5.2: Autocomplete

Όπως προαναφέρθηκε και στην λειτουργικοτητα του API παραπάνω κάθε φορά που ο χρήστης εισάγει έναν χαρακτήρα χτυπάει ένα endpoint το οποίο γυρνάει τα πρώτα δέκα αποτελέσματα που υπάρχουν στην βάση και συμφωνούν με αυτά που έχει εισάγει ο χρήστης.



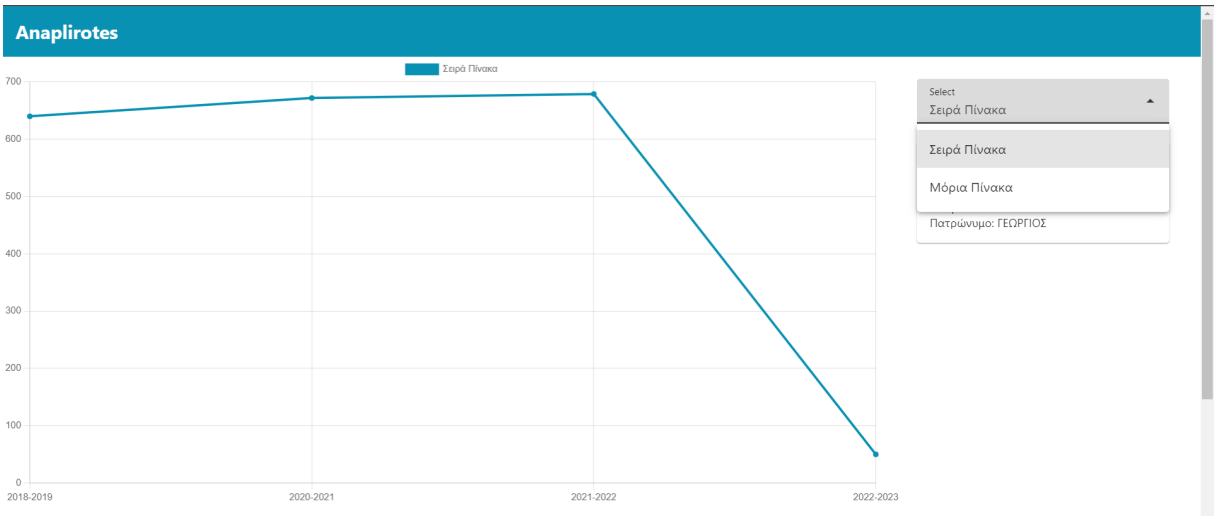
Εικόνα 5.3: Modal συνωνυμίας

Αφού ο χρήστης επιλέξει ένα όνοματεπώνυμο τότε γίνεται ο έλεγχος για τυχόν συνωνυμία. Αν αυτό το ονοματεπώνυμο είναι μοναδικό τότε δρομολογείται στην σελίδα που παρουσιάζεται αναλυτικά ο εκπαιδευτικός αυτός. Αν όχι τότε παρουσιάζεται στον χρήστη ένα modal το οποίο έχει σαν επιλογές τα πατρώνυμα των εκπαιδευτικών με το συγκεκριμένο ονοματεπώνυμο. Αφού επιλεχθεί το πατρώνυμο και αποφανθεί για ποιον εκπαιδευτικό γίνεται η αναζήτηση τότε συνεχίζεται κανονικά η ροή της εφαρμογής και καθοδηγείται στην σελίδα με όλη του την πληροφορία.



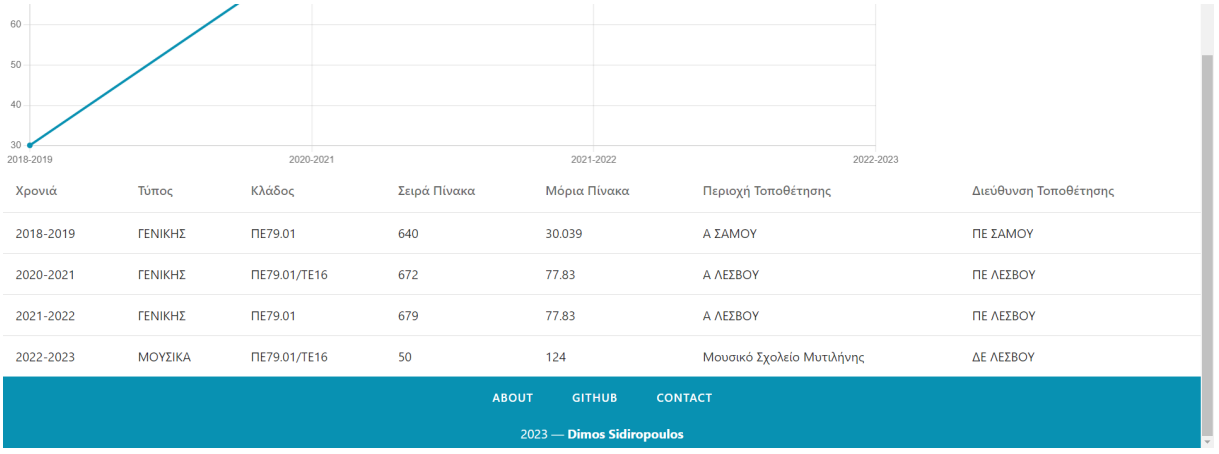
Εικόνα 5.4: Αποτέλεσμα ονομαστικής αναζήτησης

Στην οθόνη αυτή, με το αποτέλεσμα της ονομαστικής αναζήτησης, πρωταγωνιστής είναι το γράφημα. Το γράφημα αναπαριστά δύο είδη πληροφορίας, τα μόρια του εκπαιδευτικού ανα χρονιά και τη σειρά που είχε στον πίνακα ανα χρονιά.



Εικόνα 5.5: Εναλλαγή αποτελεσμάτων

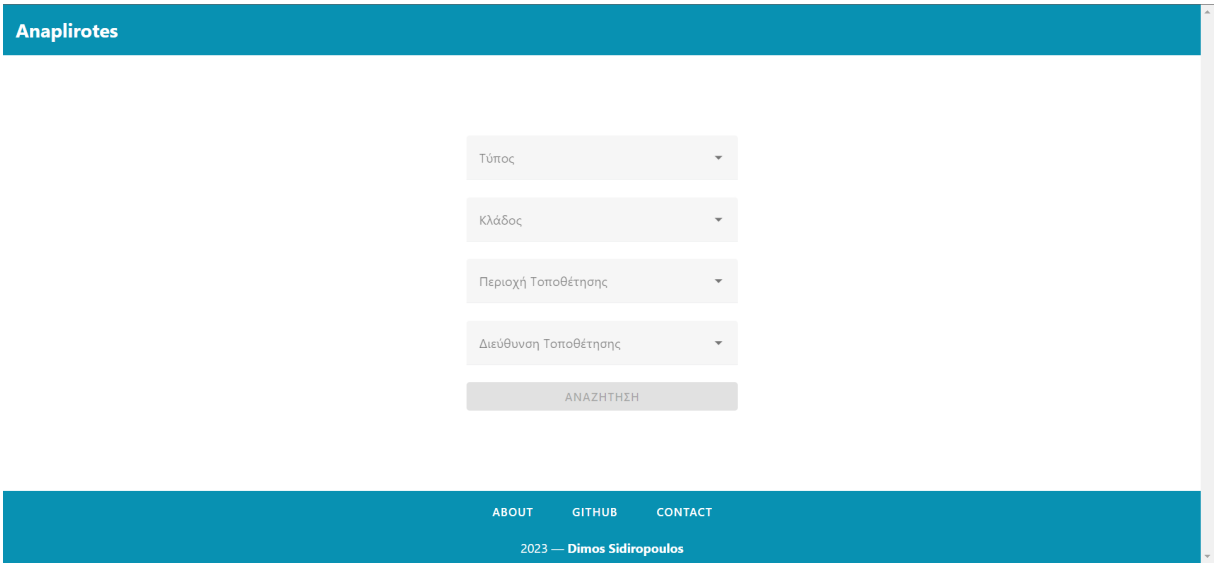
Ο τρόπος που αλλάζει η πληροφορία που προβάλλεται είναι το select component που βρίσκεται πάνω δεξιά από το οποίο όταν επιλέγεται μία τιμή αλλάζει δυναμικά και η προβολή του γραφήματος. Επίσης εμφανίζεται και μία κάρτα με τις βασικές πληροφορίες του εκπαιδευτικού για τον οποίο έγινε η αναζήτηση, το όνομα, το επίθετο και το πατρώνυμο. Η δρομολόγηση σε αυτή σελίδα γίνεται περνώντας στο url σαν παραμέτρους το όνομα, το επίθετο και το πατρώνυμο, ο λόγος που έγινε αυτό είναι για να μπορεί ο σύνδεσμος να μεταφέρεται από ένα άτομο σε ένα άλλο και αφού αυτός έχει την πληροφορία να εμφανίζεται σε όλους το ίδιο αποτέλεσμα. Αυτός είναι και ο λόγος που χρησιμοποιήθηκε και η κάρτα, για να υπάρχει εκεί η πληροφορία για ποιον εκπαιδευτικό γίνεται η αναζήτηση σε περίπτωση που δεν την έχει κάνει ο ίδιος ο χρήστης και απλά είχε πρόσβαση στον σύνδεσμο.



Εικόνα 5.6: Αναλυτικός Πίνακας στη προσωποποιημένη αναζήτηση

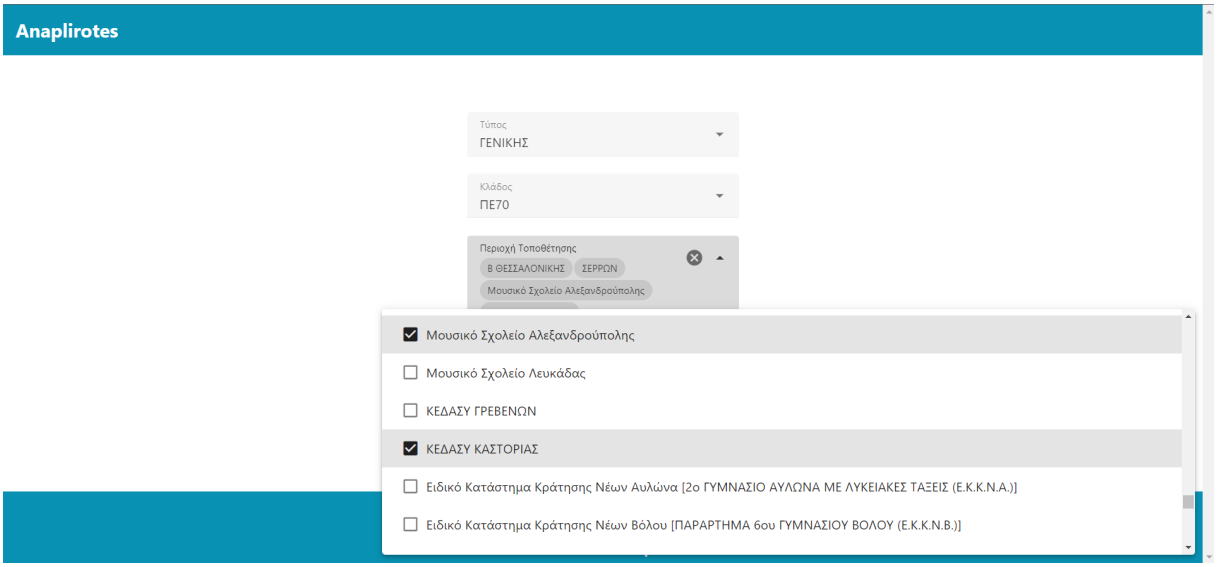
Τέλος, με την χρήση ενός πίνακα εμφανίζεται πιο λεπτομερή πληροφορία για τον συγκεκριμένο εκπαιδευτικό που περιέχει τον τύπο εκπαίδευσης, τον κλάδο, την σειρά και τα μόρια πίνακα, την περιοχή τοποθέτησης και την διεύθυνση εκπαίδευσης που ανήκει.

5.2 Αναζήτηση βάσει κριτηρίων



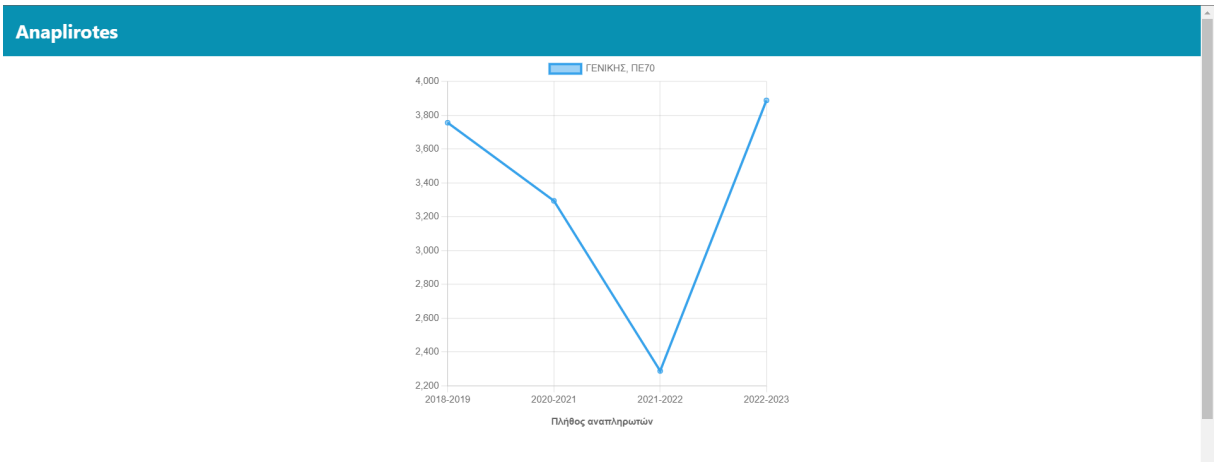
Εικόνα 5.7: Interface αναζήτησης βάσει κριτηρίων

Στην περίπτωση της αναζήτησης με βάση κριτηρίων το interface του χρήστη αποτελείται από τέσσερα inputs, για τον τύπο, τον κλαδο την περιοχή τοποθέτησης και την διεύθυνση τοποθέτησης όπως και από ένα κουμπί για την αναζήτηση. Όπως και στο πρώτο σενάριο το κουμπί ενεργοποιείται μόνο αν ο χρήστης έχει εισάγει τιμή σε κάποια από τα πεδία. Η αναζήτηση γίνεται και με μόλις ένα πεδίο και μπορεί να γίνει με κάθε πιθανό συνδυασμό των πεδίων αυτών



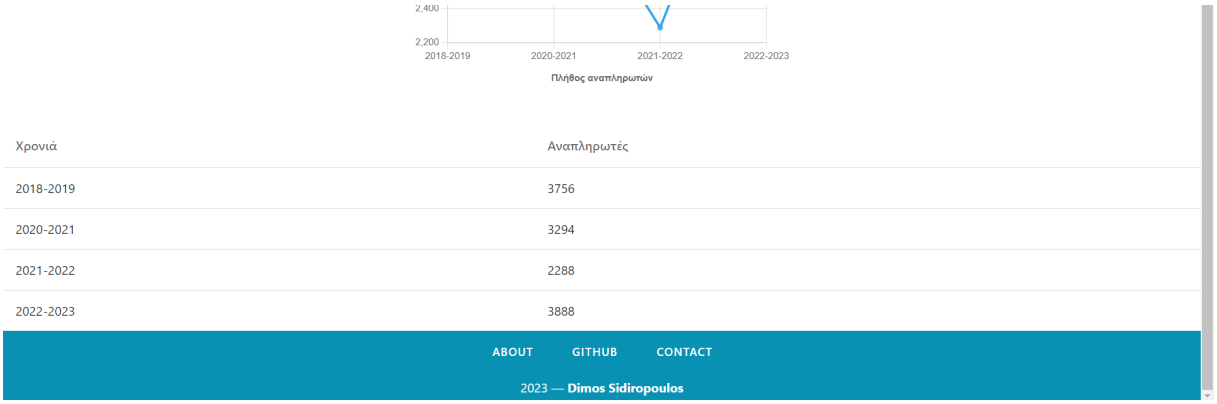
Εικόνα 5.8: Multiselect

Στην περίπτωση της περιοχής τοποθέτησης μπορεί να επιλεχθεί πάνω από μία τιμή και δεν υπάρχει κάποιο όριο, ενώ σε όλα τα άλλα πεδία μπορεί να επιλεχθεί μόνο μία τιμή. Αφού ο χρήστης γεμίσει τα πεδία που επιθυμει τότε πατώντας το κουμπί της αναζήτησης δρομολογείται σε κάποια άλλη σελίδα όπου προβάλλονται τα αποτελέσματα της αναζήτησης. Εδώ μπορούν να προκύψουν τρία υποσενάρια.



Εικόνα 5.9: Αποτέλεσμα με την χρήση κριτηρίων

Στη περίπτωση που πραγματοποιηθεί μία αναζήτηση με την επιλογή παραδείγματος χάρη του τύπου και του κλάδου τότε εμφανίζεται ένα απλό γράφημα το οποίο επιδεικνύει τον αριθμό των προσλήψεων αναπληρωτών που έγινε ανα χρονιά. Όπως φαίνεται πάνω στο γράφημα φαίνονται και τα φιλτρα που έχουν επιλεχθεί.



Εικόνα 5.10: Πίνακας αποτελεσμάτων

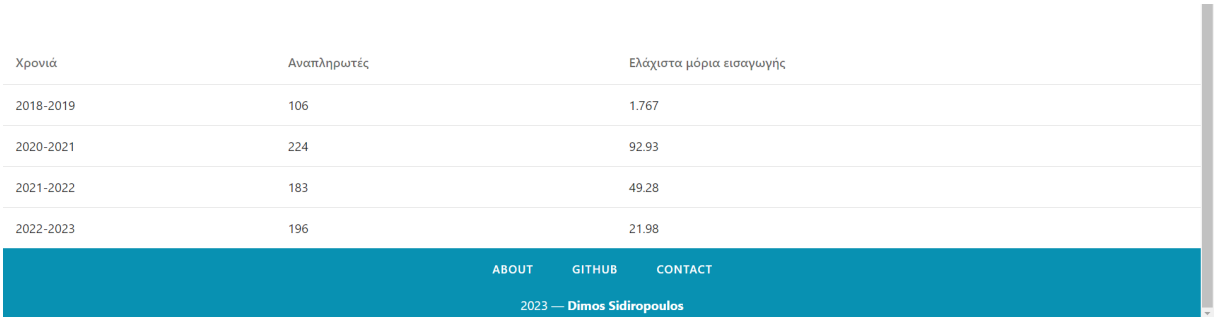
Επιπλέον υπάρχει και μία εναλλακτική οπτικοποίηση των δεδομένων με την χρήση πίνακα.

Εικόνα 5.11: Γράφημα πολλαπλών περιοχών

Στο υποσενάριο της αναζήτησης με παραπάνω από μία περιοχές τοποθέτησης τότε σαν αποτέλεσμα προβάλλεται ένα multiline γράφημα, με μία γραμμή για κάθε περιοχή ώστε να γίνεται πιο εύκολα η σύγκριση μεταξύ περιοχών. Υπενθυμίζεται ότι δεν υπάρχει όριο στο πόσες περιοχές μπορεί να επιλεχθούν. Και σε αυτή την περίπτωση εμφανίζεται ένας πίνακας με τις συνολικές προσλήψεις αναπληρωτών ανα χρονιά για τα φίλτρα που επιλέχθηκαν.

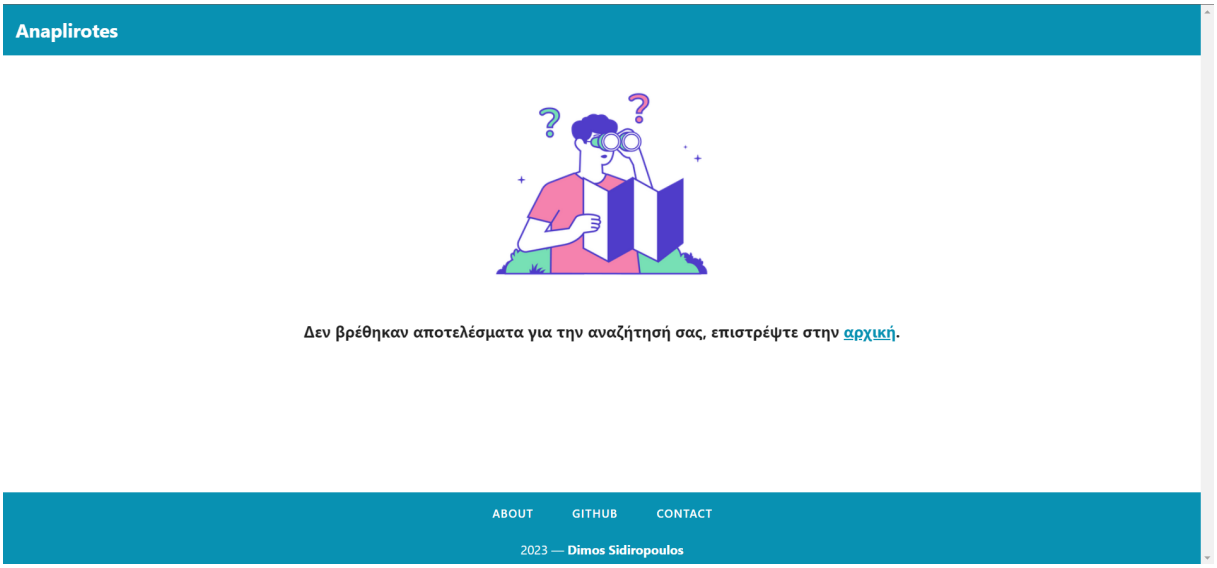
Εικόνα 5.12: Γράφημα με την ταυτόχρονη αναζήτηση κλάδου και περιοχής

Σε περίπτωση τώρα που επιλεχθούν ταυτόχρονα το φίλτρο του κλαδου και το φίλτρο της περιοχής τοποθέτησης σαν αποτέλεσμα εμφανίζεται στον χρήστη και ένα δεύτερο γράφημα. Το δεύτερο γράφημα παρουσιάζει τα ελάχιστα μόρια πρόσληψης για τον συγκεκριμένο κλάδο στις συγκεκριμένες περιοχές που επιλέχθηκαν.



Εικόνα 5.13: Πίνακας με την ταυτόχρονο αναζήτηση κλάδου και περιοχής

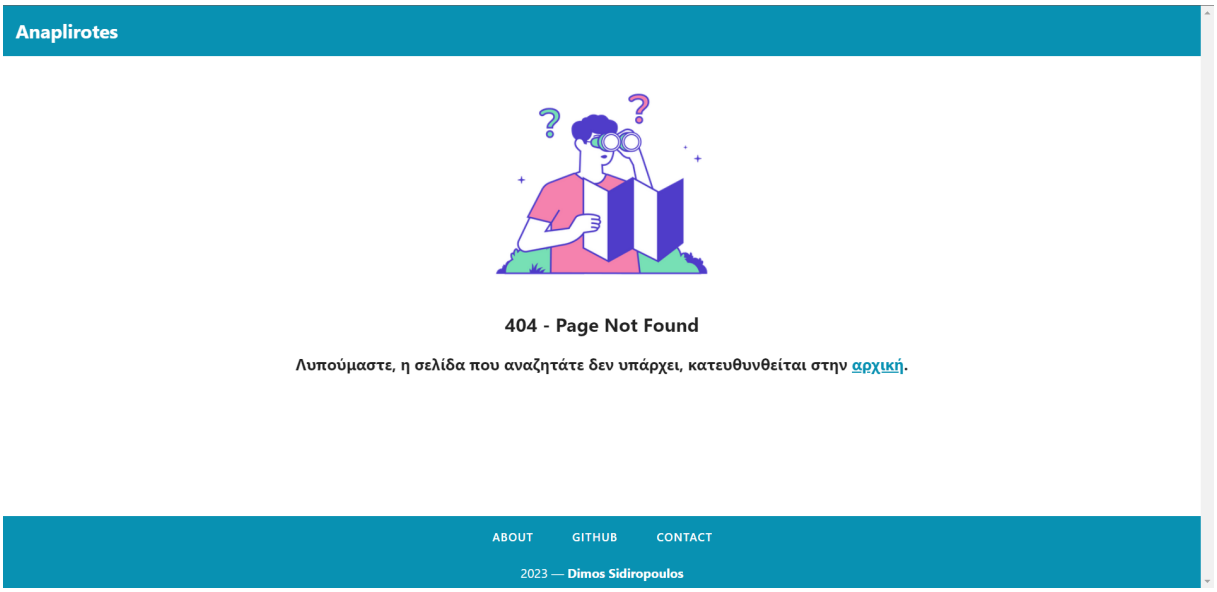
Και εδώ εμφανίζεται ένα πίνακας ο οποίος εκτός από τον συνολικό αριθμό προσλήψεων αναπληρωτών περιέχει και τα ελάχιστα μόρια εισαγωγής για κάθε χρονιά.



Εικόνα 5.14: Αναζήτηση χωρίς αποτελέσματα

Σε περίπτωση που ο χρήστης κάνει κάποιο συνδυασμό φίλτρων που δεν παράγει κάποιο αποτέλεσμα, παραδείγματος χάρη επιλέξει τύπο μουσικής με έναν κωδικό κλάδου για εκπαιδευτικό ειδικής αγωγής τότε θα του παρουσιαστεί κατάλληλο μήνυμα.

Μια παρόμοια οθόνη παρουσιάζεται και όταν ο χρήστης κατευθυνθεί σε οποιαδήποτε route από αυτά που δεν υπάρχουν στην εφαρμογή.



Εικόνα 5.15: 404 Page

Κεφάλαιο 6ο: Συμπεράσματα και Μελλοντικές επεκτάσεις

6.1 Συμπεράσματα

Η παρούσα πτυχιακή λύνει ένα πολύ σημαντικό πρόβλημα με το τωρινό σύστημα ανακοινωσεις των προσλήψεων, κάνοντας τα δεδομένα ουσιαστικά προσβάσιμα και δημιουργώντας πληροφορία. Αν και η πληροφορία δίνεται στους εκπαιδευτικούς λόγω της μορφής της, πέρα του να αναζητήσει κάποιος τον ίδιο του τον εαυτό δεν μπορεί να κάνει κάτι παραπάνω χωρίς να ξοδέψει πάρα πολύ χρόνο με το κίνδυνο πάντα η τελική του πληροφορία να μην είναι ολοκληρωμένη. Με την δημιουργία αυτής της εφαρμογής πλέον ο καθένας θα μπορεί να αναζητήσει τον οποιοδήποτε και το πιο σημαντικό να παράγει πληροφορία. Πληροφορία η οποία αφορά τις τάσεις των προσλήψεων, το ιστορικό αλλά ακόμα και να προβλέψουν που μπορεί να βρίσκονται οι ίδιοι σε έναν χώρο που οι ίδιοι καταβάλλονται από αβεβαιότητα. Κατά την την ανάπτυξη αυτής της εφαρμογής, όλα τα διαθέσιμα δεδομένα συγκεντρώθηκαν και ήρθαν σε κατάλληλη μορφή ώστε να είναι αποθηκευμένα τώρα σε μία βάση δεδομένων. Έχοντας αυτή την βάση δημιουργήθηκε ένα ανοιχτο API όπου ο καθένας μπορεί να χρησιμοποιήσει και να δημιουργήσει καινούργια πληροφορία άλλα και μία web εφαρμογή η οποία οπτικοποιεί τα δεδομένα αυτά κάνοντας την αναζήτηση διαφόρων τύπου πληροφορίας εύκολη για κάθε χρήστη.

6.2 Μελλοντικές επεκτάσεις

Λόγω του μεγάλου εύρους της πληροφορίας, η εφαρμογή μπορεί να επεκταθεί σε μεγάλο βαθμό. Μία σημαντική επέκταση είναι η δημιουργία mobile εφαρμογής. Λόγω του ανοιχτού διαθέσιμου API που δημιουργήθηκε ανοίγει ο δρόμος για την ανάπτυξη μιας android και μιας ios εφαρμογής στο μέλλον. Με την δημιουργία αυτών των εφαρμογών η συνολική πληροφορία γίνεται ακόμα πιο προσβάσιμη και είναι ένα ακόμη εργαλείο το οποίο μπορεί να εγκαταστήσει ο εκπαιδευτικός για την πληροφόρησή του. Με την δημιουργία mobile εφαρμογών μπορεί να δημιουργηθεί και ένα push-notification σύστημα που να ενημερώνει τον χρήστη όταν βγαίνουν καινούργιες ανακοινώσεις και ενημερώνεται η βάση δεδομένων ώστε να υπάρχει χρήση της εφαρμογής κατά την διάρκεια του χρόνου.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] "Ο αναπληρωτής εκπαιδευτικός, πραγματικότητα και προκλήσεις", Αθηνόγραμμα 2023. [Online]. Available: https://www.athinodromio.gr/o-αναπληρωτής-εκπαιδευτικός-πραγματ/

[2] "Python Documentation", Python.org 2023. [Online]. Available: https://docs.python.org/3/

[3] Guido Van Rossum and Fred L. Drake. 2009. Python 3 Reference Manual. CreateSpace, Scotts Valley, CA.

[4] "Django Documentation", Django 2023. [Online]. Available: https://docs.djangoproject.com/en/4.2/

[5] "Node.js Documentation", Node.js 2023. [Online]. Available: https://nodejs.org/en/docs

[6] Mike Cantelon, Marc Harter, TJ Holowaychuk, and Nathan Rajlich. 2013. Node.js in Action (1st. ed.). Manning Publications Co., USA.

[7] "NPM", NPM 2023. [Online]. Available: https://www.npmjs.com/

[8] "Express.js documentation", Express 2023. [Online]. Available: https://expressjs.com/

[9] "MySQL Documentation", MySQL 2023. [Online]. Available: https://dev.mysql.com/doc/

[10] Mike Cantelon, Marc Harter, TJ Holowaychuk, and Nathan Rajlich. 2013. Node.js in Action (1st. ed.). Manning Publications Co., USA.

[11] "JavaScript Guide", Mozilla Developer Network (MDN) 2022. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide

[12] "JavaScript Syntax & Structure", W3Schools 2022. [Online]. Available: https://www.w3schools.com/js/js_syntax.asp

[13] "ECMAScript 2015 (6th Edition, ECMA-262)", ECMA International 2015. [Online]. Available: https://www.ecma-international.org/ecma-262/6.0/

[14] "Introduction to the DOM", Mozilla Developer Network (MDN) 2022. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

[15] "Web Security", Mozilla Developer Network (MDN) 2021. [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Website_security

[16] "Progressive Web Apps", Google Developers 2021. [Online]. Available: https://web.dev/progressive-web-apps/

[17] "Web APIs", Mozilla Developer Network (MDN) 2022. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API

[18] Leonard Richardson, Mike Amundsen, and Sam Ruby. 2013. RESTful Web APIs. O'Reilly Media, Inc.

[19] "Modern JavaScript Explained For Dinosaurs", Medium 2017. [Online]. Available: https://medium.com/the-node-js-collection/modern-javascript-explained-for-dinosaurs-f695e9747b70

[20] "JavaScript Testing: Unit vs Functional vs Integration Tests", SitePoint 2019. [Online]. Available: https://www.sitepoint.com/javascript-testing-unit-functional-integration

[21] “Vue.js Official Documentation”, Vuejs.org 2023. [Online]. Available: https://vuejs.org/

[22] “Vue.js GitHub Repository”, Vuejs.org 2023. [Online]. Available: https://github.com/vuejs/vue

[23] “Single Page Application: advantages and disadvantages”, Medium 2019. [Online]. Available: https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe5

[24] “State of JS Survey”, Stateofjs 2023. [Online]. Available: https://stateofjs.com/en-US

[25] “Chart.js Documentation”, Chart.js 2023. [Online]. Available: https://www.chartjs.org/docs/latest