ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Data Scraping & Android Application

Web Service για αυτόματη καταχώρηση δεδομένων προσλήψεων αναπληρωτών εκπαιδευτικών και δημιουργία εφαρμογής για κινητές συσκευές

Φοιτητής:
Αποστολίδης Μιχαήλ
Student ID: 175108

Επιβλέπων:
Ουγιάρογλου Στέφανος
Επ. Καθηγητής

Μάιος 2025

Τίτλος Π.Ε. Data Scraping/ Android Application
Κωδικός Π.Ε. 23292
Ονοματεπώνυμο φοιτητή Αποστολίδης Μιχαήλ
Ονοματεπώνυμο εισηγητή Ουγιάρογλου Στέφανος
Ημερομηνία ανάληψης Π.Ε. 30-10-2023
Ημερομηνία περάτωσης Π.Ε. 25-05-2025

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.

«Αφιερωμένο στην οικογένειά μου.»

Πρόλογος

Κάθε χρόνο, το ελληνικό κράτος προσλαμβάνει χιλιάδες αναπληρωτές εκπαιδευτικούς σε πολλές φάσεις, ώστε να καλύψει τα κενά στην πρωτοβάθμια και δευτεροβάθμια εκπαίδευση. Οι προσλήψεις γίνονται μέσω συνεχόμενων ανακοινώσεων στη διάρκεια της σχολικής χρονιάς και συνοδεύονται από αρχεία Excel, στα οποία καταγράφεται σε ποιες περιοχές θα τοποθετηθεί κάθε εκπαιδευτικός. Παρόλο που αυτά τα δεδομένα είναι δημόσια διαθέσιμα, δεν είναι εύκολα προσβάσιμα. Ο ενδιαφερόμενος χρήστης, εκτός από την πληροφορία σχετικά με τον δικό του διορισμό, πρέπει να ψάξει χειροκίνητα μέσα σε πολλά διαφορετικά αρχεία, συχνά με διαφορετική δομή και ημερομηνίες έκδοσης, για να βρει επιπλέον πληροφορίες. Αυτό καθιστά την αναζήτηση χρονοβόρα και δύσκολη. Το πρόβλημα γίνεται πιο έντονο λόγω του ότι ο αριθμός των εκπαιδευτικών που προσλαμβάνονται ή συμμετέχουν στους πίνακες αλλάζει κάθε χρόνο. Επιπλέον, το γεγονός ότι οι εκπαιδευτικοί συχνά μετακινούνται από περιοχή σε περιοχή ενισχύει την αβεβαιότητα γύρω από τη σταθερότητα της θέσης τους και την πορεία της καριέρας τους.

Περίληψη

Η πτυχιακή χωρίζεται σε 3 επί μέρους μέρη: η απόκτηση ανοιχτής πληροφορίας με αυτόματο μηχανισμό, η προ-επεξεργασία αυτών των δεδομένων και τέλος η ανάπτυξη μιας Android εφαρμογής. Υπό κανονικές συνθήκες, τα δεδομένα θα χρειαζόταν να συλλεχθούν και να επεξεργαστούν από ανθρώπινο χέρι, το οποίο κάνει την διαδικασία δύσκολη. Για αυτό αναπτύχθηκε ένας μηχανισμός ο οποίος συλλέγει αυτόματα τα δεδομένα από την επίσημη ιστοσελίδα του Υπουργείου Παιδείας. Τα δεδομένα είναι σε μορφή excel, αφού επισκεφτεί την ιστοσελίδα, τα ψάχνει και τα αποθηκεύει στον υπολογιστή. Παρατηρήθηκε πως από excel σε excel αρχείο, η πληροφορία διαφέρει ως προς την δομή της. Για αυτόν τον λόγο, ο αλγόριθμος τρέχει το δεύτερο κομμάτι της πτυχιακής που είναι η προ-επεξεργασία δεδομένων. Αφού τελείωσε η κανονικοποίηση της, δημιουργείται ένα τελικό CSV αρχείο το οποίο είναι έτοιμο για να εισαχθεί στην βάση δεδομένων. Για τους χρήστες Android συσκευών, δημιουργήθηκε μια εφαρμογή ως μια παραπάνω επιλογή. Κρατώντας τον ίδιο UI/UX σχεδιασμό με την ήδη υπάρχων ιστοσελίδα, ο χρήστης δεν θα χρειάζεται παραπάνω χρόνο για να εξοικειωθεί στην συγκεκριμένη πλατφόρμα.

Abstract

The thesis is divided into three main parts: the acquisition of open data through an automated mechanism, the preprocessing of this data, and finally the development of an Android application. Under normal circumstances, the data would need to be collected and processed manually, which makes the process difficult. For this reason, a mechanism was developed that automatically collects data from the official website of the Ministry of Education. The data is in Excel format; after visiting the website, the system searches for the files and saves them to the computer. It was observed that the structure of the information varies from one Excel file to another. For this reason, the algorithm proceeds to the second part of the thesis, which is data preprocessing. After the normalization is complete, a final CSV file is created, ready to be imported into the database. For Android users, an application was developed as an additional option. By keeping the same UI/UX design as the existing website, users will not need extra time to become familiar with this specific platform.

Περιεχόμενα

Κατάλογος σχημάτων

Κεφάλαιο 1

Εισαγωγή

1.1 Σύστημα προσλήψεων αναπληρωτών εκπαιδευτικών στην Ελλάδα

Η πρόσληψη των αναπληρωτών καθηγητών στην Ελλάδα πραγματοποιείται μέσω ενός συστήματος κατάταξης, το οποίο βασίζεται σε διάφορους παράγοντες, όπως τα ακαδημαϊκά προσόντα, η προηγούμενη υπηρεσία και κοινωνικά κριτήρια. Αυτό το σύστημα, αν και ενθαρρύνει τη συνεχιζόμενη εκπαίδευση και τη βελτίωση των προσόντων μέσω προγραμμάτων δια βίου μάθησης, προκαλεί σημαντική ψυχολογική πίεση στους αναπληρωτές, καθώς η συνεχής σύγκριση των προσόντων τους με εκείνα των συναδέλφων τους εντείνει τον ανταγωνισμό. Οι αναπληρωτές συχνά αναφέρουν ότι η συνεχής αναζήτηση νέων προσόντων δημιουργεί άγχη και αβεβαιότητες για το μέλλον τους, αφού δεν γνωρίζουν αν θα καταφέρουν να εξασφαλίσουν τη θέση τους στην κατάταξη για την επόμενη χρονιά. Αυτή η πίεση εντείνεται κάθε φορά που οι πίνακες κατάταξης ανοίγουν, προκαλώντας αυξημένο άγχος για την τοποθέτησή τους [1].

Η απόκτηση ακαδημαϊκών προσόντων, που συχνά θεωρείται απαραίτητη για την επαγγελματική επιβίωση, μετατρέπεται σε μια διαδικασία που δεν αντανακλά την προσωπική εξέλιξη ή την ικανοποίηση από τη μάθηση, αλλά καθίσταται μια υποχρέωση για να διατηρηθεί η θέση στην αγορά εργασίας. Πολλοί αναπληρωτές αισθάνονται αδικημένοι και απογοητευμένοι, καθώς η συμμετοχή τους σε προγράμματα δια βίου μάθησης είναι περισσότερο μια αναγκαία διαδικασία για να παραμείνουν ανταγωνιστικοί, παρά μια ελεύθερη επιλογή για ανάπτυξη γνώσεων [1].

Σε αυτό το πλαίσιο, το νέο σύστημα πρόσληψης του ΥΠΑΙΘΑ, που εφαρμόστηκε το σχολικό έτος 2021-2022, προσφέρει λύσεις για την αναβάθμιση της διαδικασίας πρόσληψης και την εξοικονόμηση χρόνου, με στόχο να μειώσει την πίεση και την αβεβαιότητα στους αναπληρωτές. Το σύστημα αυτό στηρίζεται στη χρήση αλγορίθμων για την τοποθέτηση των καθηγητών σε σχολικές μονάδες, τη διαχείριση των συμβάσεων και την ενημέρωση των πλατφορμών «Διαύγεια» και «Εργάνη». Οι υποψήφιοι αναπληρωτές δηλώνουν τις προθέσεις τους μέσω του ΟΠΣΥΔ και κατατάσσουν τα σχολεία κατά προτεραιότητα, ενώ δεν είναι υποχρεωμένοι να διδάσκουν σε ένα μόνο σχολείο, αλλά μπορούν να διδάξουν σε πολλά κατά τη διάρκεια της ίδιας σχολικής χρονιάς. Τα αποτελέσματα της ανάληψης υπηρεσίας αποστέλλονται μέσω SMS, και

αν ο καθηγητής έχει περισσότερες τοποθετήσεις, επιλέγεται εκείνη με τις περισσότερες ώρες. Παρά τις ετήσιες διαφοροποιήσεις στις διαδικαστικές λεπτομέρειες, το νέο σύστημα εξοικονομεί χρόνο, δημιουργεί λιγότερη σύγχυση για τους ενδιαφερόμενους και προσφέρει μεγαλύτερη διαφάνεια και ευχέρεια στις διαδικασίες, μειώνοντας το άγχος και την αβεβαιότητα που συνοδεύουν την παραδοσιακή μέθοδο πρόσληψης [2].

Βλέποντας πώς ζουν και εργάζονται οι αναπληρωτές καθηγητές, καταλαβαίνουμε ότι παρά τις αλλαγές στο σύστημα, ακόμα αντιμετωπίζουν μεγάλη αβεβαιότητα για το μέλλον τους. Όσα πτυχία κι αν μαζέψουν αυξάνει τις πιθανότητες πρόσληψής, ενώ το να αλλάζουν σχολείο κάθε χρόνο δεν τους αφήνει να δεθούν με τα παιδιά.

1.2 Ανοιχτά δεδομένα

«Ανοιχτά δεδομένα» ονομάζουμε τα δεδομένα που είναι διαθέσιμα σε όλους για ελεύθερη χρήση, επαναχρησιμοποίηση και διανομή, υπό την προϋπόθεση ότι αναφέρεται η πηγή και ότι τυχόν παράγωγα διατίθενται με τους ίδιους όρους. Με απλά λόγια, πρόκειται για πληροφορίες (συνήθως ψηφιακές) στις οποίες οποιοσδήποτε μπορεί να έχει πρόσβαση χωρίς περιορισμούς. Στην πράξη, ο όρος συχνά αναφέρεται σε δεδομένα του δημόσιου τομέα που δημοσιεύονται από κυβερνήσεις ή οργανισμούς σε ειδικές διαδικτυακές πύλες, ώστε να μπορούν όλοι οι πολίτες, ερευνητές ή επιχειρήσεις να τα βρουν και να τα αξιοποιήσουν. Για παράδειγμα, η εθνική πύλη data.gov.gr στην Ελλάδα λειτουργεί ως κεντρικός κατάλογος όπου συγκεντρώνονται ανοικτά δημόσια δεδομένα από διάφορους φορείς, διαθέσιμα σε μηχαναγνώσιμη μορφή για κάθε ενδιαφερόμενο. Τα ανοιχτά δεδομένα καλύπτουν ένα ευρύ φάσμα θεμάτων και χρήσεων. Μπορεί να περιλαμβάνουν απλές πληροφορίες της καθημερινότητας, όπως δρομολόγια αστικών λεωφορείων ή δεδομένα για την κίνηση στους δρόμους, αλλά και κρίσιμα στοιχεία διαφάνειας, όπως τα δημοσιονομικά του κράτους, τα αποτελέσματα εκλογών ή τις συμβάσεις δημοσίων έργω. Το κλειδί είναι ότι όλα αυτά δημοσιεύονται ελεύθερα στο διαδίκτυο σε μορφές φιλικές προς τον χρήστη και επεξεργάσιμες από υπολογιστές. Με αυτόν τον τρόπο, κάθε πολίτης μπορεί εύκολα να ενημερωθεί και να αξιοποιήσει την πληροφορία. Η λογική πίσω από την ανοικτή διάθεση των δεδομένων είναι διττή: αφενός διασφαλίζεται η διαφάνεια στη λειτουργία του κράτους και ο δημόσιος έλεγχος, και αφετέρου δίνεται η βάση για τη δημιουργία νέων υπηρεσιών και εφαρμογών που βελτιώνουν την καθημερινή ζωή. Έτσι, για παράδειγμα, η δημοσιοποίηση στοιχείων για τις δημόσιες δαπάνες επιτρέπει στον πολίτη να παρακολουθεί «το δρόμο του χρήματος», ενώ τα ανοιχτά δεδομένα σε τομείς όπως οι μεταφορές μπορούν να αξιοποιηθούν από προγραμματιστές για την ανάπτυξη εφαρμογών πλοήγησης και ενημέρωσης δρομολογίων, προς όφελος όλων. [3]

Τα ανοιχτά δεδομένα καλύπτουν ένα ευρύ φάσμα θεμάτων και χρήσεων. Μπορεί να περιλαμβάνουν απλές πληροφορίες της καθημερινότητας, όπως δρομολόγια αστικών λεωφορείων ή δεδομένα για την κίνηση στους δρόμους, αλλά και κρίσιμα στοιχεία διαφάνειας, όπως τα δημοσιονομικά του κράτους, τα αποτελέσματα εκλογών ή τις συμβάσεις δημοσίων έργων. Το κλειδί είναι ότι όλα αυτά δημοσιεύονται ελεύθερα στο διαδίκτυο σε μορφές φιλικές προς τον χρήστη και επεξεργάσιμες από υπολογιστές. Με αυτόν τον τρόπο, κάθε πολίτης μπορεί εύκολα να ενημερωθεί και να αξιοποιήσει την πληροφορία. Η λογική πίσω από την ανοικτή διάθεση των δεδομένων είναι διττή: αφενός διασφαλίζεται η διαφάνεια στη λειτουργία του κράτους και

ο δημόσιος έλεγχος, και αφετέρου δίνεται η βάση για τη δημιουργία νέων υπηρεσιών και εφαρμογών που βελτιώνουν την καθημερινή ζωή. Έτσι, για παράδειγμα, η δημοσιοποίηση στοιχείων για τις δημόσιες δαπάνες επιτρέπει στον πολίτη να παρακολουθεί «το δρόμο του χρήματος», ενώ τα ανοιχτά δεδομένα σε τομείς όπως οι μεταφορές μπορούν να αξιοποιηθούν από προγραμματιστές για την ανάπτυξη εφαρμογών πλοήγησης και ενημέρωσης δρομολογίων, προς όφελος όλων. [4]

Τα οφέλη από τη χρήση και διάθεση ανοικτών δεδομένων είναι πολλαπλά, όπως έχουν επισημάνει τόσο η επιστημονική κοινότητα όσο και κρατικοί φορείς. Σύμφωνα με πληθώρα ακαδημαϊκών μελετών, η ενίσχυση της διαφάνειας και της λογοδοσίας είναι από τα σημαντικότερα πλεονεκτήματα: όταν τα κυβερνητικά δεδομένα είναι προσβάσιμα και ελέγξιμα από όλους, ενδυναμώνεται η εμπιστοσύνη των πολιτών προς τη διοίκηση και δίνεται η δυνατότητα στους πολίτες να ζητούν υπευθυνότητα από τους αξιωματούχους. Παράλληλα, η οικονομική ανάπτυξη και η καινοτομία συγκαταλέγονται στα θετικά αποτελέσματα. Έρευνες δείχνουν ότι τα ανοιχτά δεδομένα μπορούν να λειτουργήσουν ως μοχλός ανάπτυξης, καθώς η ελεύθερη διάθεση πληροφοριών ενθαρρύνει την αγορά να δημιουργήσει νέα προϊόντα και υπηρεσίες, γεγονός που αυξάνει την παραγωγικότητα, δημιουργεί νέες θέσεις εργασίας και αποφέρει πρόσθετα έσοδα στο κράτος (π.χ. από φόρους). Επιπλέον, η ενεργός συμμετοχή των πολιτών ενισχύεται: όταν οι πολίτες έχουν πρόσβαση σε δεδομένα, μπορούν να αλληλεπιδρούν με το κράτος πιο ενημερωμένα και ουσιαστικά, συμμετέχοντας πιο αποτελεσματικά στη λήψη αποφάσεων και στα κοινά. Αξίζει να σημειωθεί ότι η ανοικτότητα δεν ωφελεί μόνο το κοινό, αλλά και τους ίδιους τους φορείς που δημοσιεύουν δεδομένα, για παράδειγμα, έχει παρατηρηθεί ότι όταν τα δεδομένα είναι δημόσια, οι πολίτες μπορούν να επισημάνουν λάθη ή ελλείψεις, συμβάλλοντας έτσι στη βελτίωση της ποιότητας των πληροφοριών που διαθέτει η διοίκηση. [5]

Οι κυβερνήσεις και οι δημόσιοι οργανισμοί ανά τον κόσμο αναγνωρίζουν τα παραπάνω οφέλη και τα προβάλλουν στις πολιτικές τους. Σε ευρωπαϊκό επίπεδο, τα ανοικτά δεδομένα θεωρούνται καταλύτης για οικονομική ανάπτυξη, εργαλείο διαφάνειας και λογοδοσίας, αλλά και πηγή καινοτομίας και γνώσης για την κοινωνία[6]. Στην Ελλάδα, η εθνική ψηφιακή στρατηγική υπογραμμίζει ότι η ανοικτή διάθεση των δεδομένων του Δημοσίου ενισχύει τη συμμετοχή των πολιτών και τη διαφάνεια, ενώ παράλληλα προωθεί την επιχειρηματικότητα παρέχοντας στις νεοφυείς επιχειρήσεις πολύτιμη πρώτη ύλη για την ανάπτυξη υπηρεσιών προστιθέμενης αξίας [7]. Ένα χαρακτηριστικό παράδειγμα είναι το πρόγραμμα «Δι@ύγεια», μέσω του οποίου δημοσιεύονται διαδικτυακά όλες οι κυβερνητικές αποφάσεις και δαπάνες, επιτρέποντας τον άμεσο έλεγχο της διοίκησης από τους πολίτες. Αντίστοιχα, σε παγκόσμιο επίπεδο, πολλές πρωτοβουλίες ανοικτών δεδομένων έχουν οδηγήσει σε καινοτόμες εφαρμογές που ωφελούν το κοινωνικό σύνολο. Για παράδειγμα, η ανοικτή διάθεση δεδομένων μεταφορών στο Λονδίνο αξιοποιείται από περισσότερους από 5.000 προγραμματιστές για τη δημιουργία δημοφιλών εφαρμογών αστικών μετακινήσεων, διευκολύνοντας τις μεταφορές των πολιτών[8]. Με αυτόν τον τρόπο, τα ανοιχτά δεδομένα καθιερώνονται πλέον ως θεμελιώδες εργαλείο για πιο διαφανή διακυβέρνηση και ως κινητήριος δύναμη καινοτομίας, προς όφελος τόσο της οικονομίας όσο και της κοινωνίας.

Εξετάζοντας τα οφέλη των ανοιχτών δεδομένων, διαπιστώνουμε ότι θα μπορούσαν να βελτιώσουν σημαντικά το σύστημα πρόσληψης αναπληρωτών. Η διαφάνεια θα επέτρεπε στους εκπαιδευτικούς να γνωρίζουν τα κενά, τα κριτήρια και τις διαδικασίες, μειώνοντας την αβεβαιότητα. Τα στατιστικά στοιχεία για τις ανάγκες των σχολείων θα βοηθούσαν τους αναπληρωτές

να προγραμματίζουν καλύτερα την πορεία τους και να αποκτούν τα προσόντα που πραγματικά χρειάζονται. Επιπλέον, θα μπορούσαν να αναπτυχθούν εφαρμογές για καλύτερη αντιστοίχιση εκπαιδευτικών με σχολεία, λαμβάνοντας υπόψη περισσότερους παράγοντες πέρα από τα τυπικά προσόντα. Έτσι, τα ανοιχτά δεδομένα θα συνέβαλαν σε ένα πιο δίκαιο και λιγότερο αγχωτικό σύστημα.

1.3 Κίνητρο

Η ανάγκη για συλλογή και κανονικοποίηση δεδομένων προέκυψε ως συνέχεια μιας προηγούμενης πτυχιακής εργασίας, στην οποία είχε δημιουργηθεί ένας ιστότοπος για την προβολή πληροφοριών σχετικά με τις προσλήψεις αναπληρωτών εκπαιδευτικών. Παρόλο που τα δεδομένα είναι διαθέσιμα δημόσια, κάθε χρόνο το Υπουργείο Παιδείας δημοσιεύει ανακοινώσεις και πίνακες σε μορφή Excel που παρουσιάζουν σημαντικές διαφοροποιήσεις στη δομή, την ονοματολογία και το περιεχόμενο. Αυτό καθιστά δύσκολη τη συγκέντρωση και επεξεργασία των αρχείων, καθώς απαιτείται χειροκίνητος έλεγχος και προσαρμογή για να μπορούν να χρησιμοποιηθούν οργανωμένα. Η ασυμβατότητα των δεδομένων και η έλλειψη ενιαίας μορφής αποτελούν εμπόδιο τόσο για την ομαλή λειτουργία του ιστότοπου όσο και για τους ίδιους τους χρήστες που αναζητούν πληροφορίες. Έτσι, διαπιστώθηκε η ανάγκη για μια λύση που να επιτρέπει την αυτόματη συγκέντρωση και ομογενοποίηση των δεδομένων, προκειμένου να διατίθενται με εύχρηστο και κατανοητό τρόπο.

1.4 Συνεισφορά

Στο πλαίσιο της παρούσας πτυχιακής εργασίας αναπτύχθηκε ένας μηχανισμός αυτοματοποιημένης συλλογής και κανονικοποίησης δεδομένων. Ο μηχανισμός αυτός αντλεί τα αρχεία που δημοσιεύονται από τους αρμόδιους φορείς, επεξεργάζεται το περιεχόμενό τους και το μετατρέπει σε ενιαία μορφή, κατάλληλη για την απευθείας εισαγωγή και χρήση από τον ιστότοπο που είχε δημιουργηθεί σε προηγούμενη εργασία. Με αυτόν τον τρόπο, επιλύεται το πρόβλημα της διαφορετικής δομής και ασυνέπειας των αρχείων, ενώ παράλληλα μειώνεται σημαντικά η ανάγκη για χειροκίνητη επεξεργασία. Επιπλέον, στο πλαίσιο της εργασίας υλοποιήθηκε και μια Android εφαρμογή, η οποία προσφέρει στους χρήστες τη δυνατότητα να έχουν άμεση και εύκολη πρόσβαση στα επεξεργασμένα δεδομένα μέσω κινητής συσκευής, διευκολύνοντας έτσι ακόμη περισσότερο την ενημέρωσή τους.

1.5 Οργάνωση εργασίας

Η παρούσα πτυχιακή εργασία είναι οργανωμένη σε έξι κεφάλαια. Στο πρώτο κεφάλαιο παρουσιάζεται η εισαγωγή, όπου εξηγείται συνοπτικά ποιοι είναι οι αναπληρωτές καθηγητές, πώς λειτουργεί το σύστημα πρόσληψής τους και ποια προβλήματα αντιμετωπίζουν. Στη συνέχεια, γίνεται αναφορά στη σωστή αξιοποίηση των ανοιχτών δεδομένων που παρέχει το Υπουργείο Παιδείας και περιγράφεται η δημιουργία μιας διαδικτυακής εφαρμογής, η οποία έρχεται να δώσει λύση στα υπάρχοντα προβλήματα.

Στο δεύτερο κεφάλαιο γίνεται αναλυτική επεξήγηση της ήδη υπάρχουσας εφαρμογής eAnaplirotes, με έμφαση στη δομή της βάσης δεδομένων και στη λειτουργία του Web API που τη στηρίζει.

Το τρίτο κεφάλαιο επικεντρώνεται στις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση της εργασίας, καθώς καλύπτει δύο αρκετά διαφορετικά σημεία. Τα κύρια εργαλεία που αξιοποιήθηκαν είναι: το React Native για την ανάπτυξη εφαρμογής σε κινητές συσκευές, το Python Selenium για την αυτόματη συλλογή δεδομένων και το Python Pandas για την επεξεργασία και κανονικοποίηση των δεδομένων.

Στο τέταρτο κεφάλαιο αναλύεται η διαδικασία σχεδίασης και ανάπτυξης ενός web service για την αυτόματη ανάκτηση και προεπεξεργασία αρχείων προσλήψεων αναπληρωτών. Παρουσιάζεται ο τρόπος σκέψης πίσω από την υλοποίηση, τα προβλήματα που προέκυψαν και οι τρόποι αντιμετώπισής τους, καθώς και γίνεται αξιολόγηση των επιλογών που πραγματοποιήθηκαν σε επίπεδο τεχνολογίας και σχεδίασης.

Στο πέμπτο κεφάλαιο περιγράφεται η ανάπτυξη της εφαρμογής για κινητές συσκευές, όπου αναπτύχθηκε μια εφαρμογή front-end κάνοντας χρήση του ήδη υπάρχοντος API. Αναλύονται οι δυνατότητες που παρέχονται στον χρήστη, τα πλεονεκτήματα της επιλεγμένης τεχνολογίας ανάπτυξης εφαρμογών, καθώς και ορισμένοι περιορισμοί που προέκυψαν κατά την υλοποίηση και ενσωμάτωση των APIs.

Στο έκτο και τελευταίο κεφάλαιο παρατίθενται τα συμπεράσματα και οι παρατηρήσεις που προέκυψαν τόσο κατά τη διάρκεια της ανάπτυξης όσο και μετά την ολοκλήρωση της εργασίας. Τέλος, προτείνονται μελλοντικές βελτιώσεις και επεκτάσεις που θα μπορούσαν να αναβαθμίσουν περαιτέρω το έργο και τον ιστότοπο eAnaplirotes.

Κεφάλαιο 2

Η εφαρμογή eAnaplirotes

2.1 Περιγραφή

Η παρούσα εργασία προτείνει τη δημιουργία ενός συστήματος με την ονομασία *eAnaplirotes*, το οποίο υλοποιείται ως διαδικτυακή εφαρμογή. Το σύστημα αποτελείται από τρία βασικά μέρη: την ιστοσελίδα (front-end), τη βάση δεδομένων και ένα Web API (back-end) που διασφαλίζει την επικοινωνία ανάμεσά τους.

Η εφαρμογή παρέχει δύο βασικές λειτουργίες αναζήτησης: την αναζήτηση βάσει ονοματεπωνύμου και τη γενική αναζήτηση με φίλτρα.

Στην πρώτη περίπτωση, ο χρήστης εισάγει το ονοματεπώνυμο ενός εκπαιδευτικού. Η εφαρμογή προτείνει δυναμικά αποτελέσματα μέσω autocomplete, και εφόσον χρειαστεί, ζητείται και το πατρώνυμο για ακριβή ταυτοποίηση. Στη συνέχεια, ο χρήστης μεταφέρεται σε μια σελίδα όπου προβάλλεται ένας γράφος με τα μόρια ή τη σειρά πίνακα του εκπαιδευτικού ανά έτος. Κάτω από τον γράφο εμφανίζεται αναλυτικός πίνακας με τα υπόλοιπα στοιχεία του.

Η δεύτερη λειτουργία, αυτή της γενικής αναζήτησης, δίνει τη δυνατότητα στον χρήστη να επιλέξει έναν ή περισσότερους συνδυασμούς φίλτρων (τύπος, κλάδος, περιοχή τοποθέτησης, διεύθυνση εκπαίδευσης). Τα αποτελέσματα παρουσιάζονται με γραφική απεικόνιση των προσλήψεων ανά έτος και σχετικό πίνακα. Επιπλέον, όταν έχει οριστεί και ο κλάδος και η περιοχή, εμφανίζεται δεύτερος γράφος με τα ελάχιστα μόρια πρόσληψης ανά χρονιά για τον συγκεκριμένο συνδυασμό.

2.2 Βάση Δεδομένων

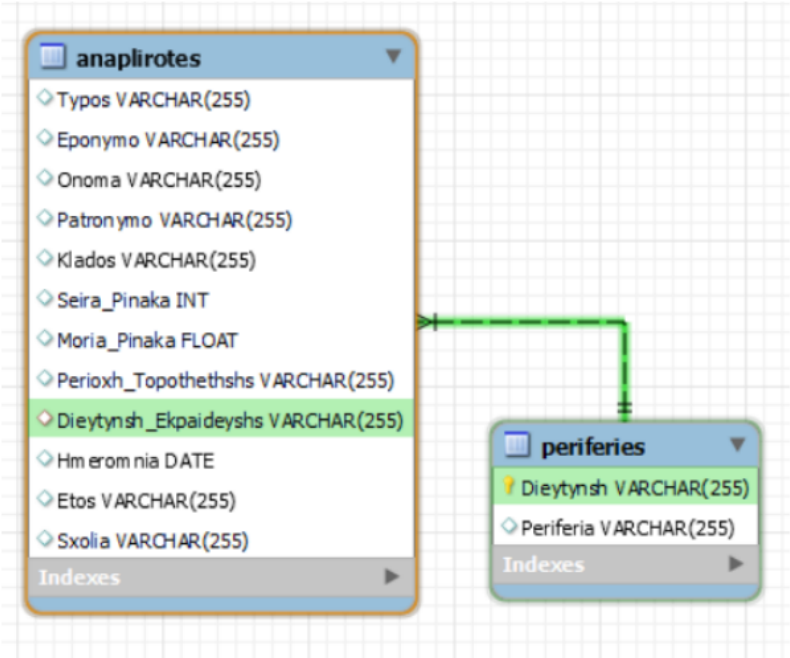
Όλα ξεκινούν από τα πρωτογενή δεδομένα που δημοσιεύονται από το Υπουργείο Παιδείας. Τα αρχεία αυτά παρουσιάζουν σημαντικές διαφορές τόσο στη δομή όσο και στην ονοματολογία τους, γεγονός που καθιστά τη συλλογή και διαχείρισή τους αρκετά απαιτητική. Δεν ακολουθούν κάποιο σταθερό πρότυπο και συχνά περιέχουν πεδία με διαφορετικά ονόματα αλλά παρόμοιο περιεχόμενο.

Γι' αυτόν τον λόγο, πριν εισαχθούν τα δεδομένα στη βάση, ακολουθεί μια διαδικασία καθαρι-

σμού και προσαρμογής. Γίνεται επιλογή συγκεκριμένων πεδίων που κρίνονται χρήσιμα, ενοποιούνται έννοιες, διορθώνονται ασυνέπειες και προστίθενται νέα πεδία όπως το σχολικό έτος και η ημερομηνία. Ο στόχος είναι τα δεδομένα να έρθουν σε μια όσο το δυνατόν πιο σταθερή και αναγνώσιμη μορφή.

Η βάση δεδομένων αποτελείται από δύο πίνακες: έναν κύριο πίνακα με όλους τους αναπληρωτές και έναν δεύτερο, βοηθητικό πίνακα για τις διευθύνσεις εκπαίδευσης και τις αντίστοιχες περιφέρειες. Οι δύο πίνακες συνδέονται μεταξύ τους μέσω ξένου κλειδιού. Η δομή αυτή απεικονίζεται στο διάγραμμα οντοτήτων-συσχετίσεων (ERD) του σχήματος 2.1, όπου φαίνονται καθαρά οι πίνακες, τα πεδία τους και η σχέση που τους συνδέει. Το ERD αποτελεί τη βάση για την υλοποίηση του σχήματος της βάσης δεδομένων και διευκολύνει την κατανόηση της οργάνωσης των πληροφοριών.

Πριν τα δεδομένα αποθηκευτούν στη βάση, ακολουθείται μια τυποποιημένη διαδικασία μετατροπής για κάθε αρχείο. Η διαδικασία αυτή περιλαμβάνει τον καθαρισμό κενών ή περιττών πεδίων, την κανονικοποίηση της ονοματολογίας και την εναρμόνιση των δεδομένων σε ένα ενιαίο σχήμα. Όπου χρειάζεται, γίνονται απλοποιήσεις ή συγχωνεύσεις στηλών και τιμών, ώστε οι εγγραφές να είναι κατάλληλα δομημένες και εύκολα αξιοποιήσιμες από την εφαρμογή. Με αυτόν τον τρόπο διασφαλίζεται ότι η πληροφορία που τελικά αποθηκεύεται είναι καθαρή, πλήρης και συνεπής με τη λογική της βάσης.



Σχήμα 2.1: Διάγραμμα Οντοτήτων - Συσχετίσεων της βάσης δεδομένων eAnaplirotes

2.3 Web API

Το Web API της εφαρμογής λειτουργεί ως γέφυρα μεταξύ της βάσης δεδομένων και ότι βλέπει ο χρήστης, παρέχοντας όλα τα απαραίτητα δεδομένα μέσα από οργανωμένα endpoints. Η υλοποίησή του έγινε με χρήση της Node.js και του framework Express(που ανοίκει στην Node.js), ενώ για την αποθήκευση των δεδομένων χρησιμοποιείται η MySQL.

Το API διαχειρίζεται δύο βασικά σενάρια αναζήτησης: την προσωπική (με βάση τα στοιχεία ενός συγκεκριμένου εκπαιδευτικού) και τη γενική (με χρήση φίλτρων όπως τύπος, κλάδος, περιοχή και διεύθυνση εκπαίδευσης).

Στο πρώτο σενάριο, ο χρήστης πληκτρολογεί το όνομα που τον ενδιαφέρει και, με τη βοήθεια του μηχανισμού autocomplete, προτείνονται τα πρώτα διαθέσιμα αποτελέσματα. Αν υπάρχουν συνωνυμίες, ζητείται και το πατρώνυμο ώστε να γίνει ακριβής ταυτοποίηση. Αφού εντοπιστεί το πρόσωπο, καλείται ένα ειδικό endpoint που επιστρέφει το σύνολο των πληροφοριών του εκπαιδευτικού για όλα τα διαθέσιμα έτη.

Στο δεύτερο σενάριο, ο χρήστης εφαρμόζει φίλτρα για να δει συγκεντρωτικά αποτελέσματα. Υπάρχουν endpoints που επιστρέφουν τις διαθέσιμες τιμές για κάθε φίλτρο (τύπος, κλάδος, περιοχή και διεύθυνση), ώστε να γεμίζουν τα αντίστοιχα select. Μόλις οριστούν τα κριτήρια, ενεργοποιείται το endpoint αναζήτησης (/search), το οποίο φέρνει όλα τα σχετικά δεδομένα. Ο Πίνακας 2.1 παρουσιάζει αναλυτικά τα διαθέσιμα endpoints του API, τη λειτουργία τους και τις παραμέτρους που δέχονται, παρέχοντας μια συνολική εικόνα της διεπαφής προγραμματισμού της εφαρμογής.

Το API είναι σχεδιασμένο να χειρίζεται τα ερωτήματα χτίζοντας δυναμικά τα WHERE clauses και επιστρέφοντας μόνο τα απαραίτητα αποτελέσματα. Επιστρέφει επίσης τα δεδομένα με τέτοιο τρόπο ώστε να μπορούν να χρησιμοποιηθούν άμεσα σε γραφήματα και πίνακες της διεπαφής. Ανάλογα με το αν έχει επιλεγεί και κλάδος, στέλνει και ένα δεύτερο σύνολο δεδομένων για τα ελάχιστα μόρια πρόσληψης ανά περιοχή και έτος.

Μέθοδος	**API Endpoint**
POST	`/api/anaplirotes/`
GET	`/api/onoma/:id`
GET	`/api/user/:Eponymo/:Onoma/:Patronymo`
GET	`/api/typos`
GET	`/api/typos`
GET	`/api/klados`
GET	`/api/perioxh`
GET	`/api/dieythynsh`
GET	`/api/search?typos={typ}&klados={klad}&perioxh={per}` `&dieythynsh={diey}`

Πίνακας 2.1: API Endpoints

POST /api/anaplirotes/

Επιστρέφει λίστα ονομάτων αναπληρωτών που ταιριάζουν με την αναζήτηση. Το αίτημα περιλαμβάνει JSON body με πεδίο name, π.χ. { ”name”: ”Μαρ” }.

Παράδειγμα αιτήματος

```
POST /api/anaplirotes/
Content-Type: application/json

{
    "name": "Μαρ"
}
```

Παράδειγμα απάντησης

```
[
    "ΧΑΣΑΠΗ ΜΑΡΙΑ",
    "ΚΑΡΑΔΗΜΑ ΜΑΡΙΓΙΑΝΝΑ",
    "ΜΑΡΚΑΚΗ ΔΗΜΗΤΡΑ",
    "ΚΟΣΜΑΡΙΚΟΥ ΔΗΜΗΤΡΑ",
```

```
    "ΚΙΟΠΡΟΥΛΗ ΜΑΡΙΑ",
    "ΑΥΓΕΡΙΝΟΥΔΗ ΔΗΜΗΤΡΑ ΜΑΡΙΑ",
    "ΤΡΙΑΝΤΑΦΥΛΛΙΔΟΥ ΜΑΡΙΑ ΑΝΝΑ",
    "ΚΕΜΠΕΣΟΥ ΜΑΡΙΑΝΘΗ",
    "ΚΟΥΤΣΟΥΜΑΡΗ ΝΙΚΟΛΕΤΤΑ",
    "ΦΙΛΙΠΠΟΠΟΥΛΟΥ ΜΑΡΙΑ"

]
```

GET /api/onoma/:id

Επιστρέφει όλη την πληροφορία για έναν συγκεκριμένο εκπαιδευτικό με βάση Επώνυμο, Όνομα και Πατρώνυμο. Η απάντηση είναι ένας πίνακας από αντικείμενα, ένα για κάθε σχολικό έτος.

Παράδειγμα αιτήματος

```
GET /onoma/ΒΟΥΓΙΟΥΚΑ_ΜΑΡΙΑ
```

Παράδειγμα απάντησης (μερική λόγο μεγάλου πλήθους γραμμών)


```
[
  {
        "Typos": "ΓΕΝΙΚΗΣ",
        "Eponymo": "ΒΟΥΓΙΟΥΚΑ",
        "Onoma": "ΜΑΡΙΑ",
        "Patronymo": "ΓΕΩΡΓΙΟΣ",
        "Klados": "ΠΕ79.01",
        "Seira_Pinaka": 640,
        "Moria_Pinaka": 30.039,
        "Perioxh_Topothethshs": "Α ΣΑΜΟΥ",
        "Dieytynsh_Ekpaideyshs": "ΠΕ ΣΑΜΟΥ",
        "Hmeromnia": "2018-09-04T21:00:00.000Z",
        "Etos": "2018-2019",
        "Sxolia": "ΑΠΩ"
    },
    {
        "Typos": "ΓΕΝΙΚΗΣ",
        "Eponymo": "ΒΟΥΓΙΟΥΚΑ",
        "Onoma": "ΜΑΡΙΑ",
        "Patronymo": "ΓΕΩΡΓΙΟΣ",
        "Klados": "ΠΕ79.01/ΤΕ16",
        "Seira_Pinaka": 654,
        "Moria_Pinaka": 52,
        "Perioxh_Topothethshs": "Α ΛΕΣΒΟΥ",
        "Dieytynsh_Ekpaideyshs": "ΠΕ ΛΕΣΒΟΥ",
        "Hmeromnia": "2019-08-24T21:00:00.000Z",
        "Etos": "2019-2020",
```

```
        "Sxolia": "ΑΠΩ"
    },
    ...
]
```

GET /user/:Eponymo/:Onoma/:Patronymo

Επιστρέφει όλη την πληροφορία για έναν συγκεκριμένο εκπαιδευτικό με βάση Επώνυμο, Όνομα και Πατρώνυμο. Η απάντηση είναι ένας πίνακας από αντικείμενα, ένα για κάθε σχολικό έτος.

Παράδειγμα αιτήματος

```
GET /user/ΒΟΥΓΙΟΥΚΑ/ΜΑΡΙΑ/ΓΕΩΡΓΙΟΣ
```

Παράδειγμα απάντησης (μερική λόγο μεγάλου πλήθους γραμμών)

```
[
    {
        "Typos": "ΓΕΝΙΚΗΣ",
        "Eponymo": "ΒΟΥΓΙΟΥΚΑ",
        "Onoma": "ΜΑΡΙΑ",
        "Patronymo": "ΓΕΩΡΓΙΟΣ",
        "Klados": "ΠΕ79.01",
        "Seira_Pinaka": 640,
        "Moria_Pinaka": 30.039,
        "Perioxh_Topothethshs": "Α ΣΑΜΟΥ",
        "Dieytynsh_Ekpaideyshs": "ΠΕ ΣΑΜΟΥ",
        "Hmeromnia": "2018-09-04T21:00:00.000Z",
        "Etos": "2018-2019",
        "Sxolia": "ΑΠΩ"
    },
    {
        "Typos": "ΓΕΝΙΚΗΣ",
        "Eponymo": "ΒΟΥΓΙΟΥΚΑ",
        "Onoma": "ΜΑΡΙΑ",
        "Patronymo": "ΓΕΩΡΓΙΟΣ",
        "Klados": "ΠΕ79.01/ΤΕ16",
        "Seira_Pinaka": 654,
        "Moria_Pinaka": 52,
        "Perioxh_Topothethshs": "Α ΛΕΣΒΟΥ",
        "Dieytynsh_Ekpaideyshs": "ΠΕ ΛΕΣΒΟΥ",
        "Hmeromnia": "2019-08-24T21:00:00.000Z",
        "Etos": "2019-2020",
        "Sxolia": "ΑΠΩ"
```

```
    },
    ...
]
```

GET /typos

Επιστρέφονται όλοι οι τύποι εκπαίδευσης που υπάρχουν στη βάση δεδομένων.

Παράδειγμα αιτήματος

```
GET /typos
```

Παράδειγμα απάντησης

```
[
    "ΜΟΥΣΙΚΑ",
    "ΓΕΝΙΚΗΣ",
    "ΕΙΔΙΚΗΣ"
]
```

GET /klados

Επιστρέφονται όλοι οι κλάδοι εκπαίδευσης που υπάρχουν στη βάση δεδομένων.

Παράδειγμα αιτήματος

```
GET /klados
```

Παράδειγμα απάντησης (μερική λόγο μεγάλου πλήθους γραμμών)

```
[
    "ΠΕ79.01/ΤΕ16",
    "ΠΕ01",
    "ΠΕ02",
    "ΠΕ03",
    "ΠΕ04.01",
    "ΠΕ04.02",
    "ΠΕ04.04",
    "ΠΕ04.05",
    "ΠΕ05",
    "ΠΕ06",
    "ΠΕ07",
    "ΠΕ08",
    "ΠΕ11",
    "ΠΕ78",
    "ΠΕ80",
    "ΠΕ81",
    "ΠΕ82",
    ...
]
```

GET /perioxh

Επιστρέφονται όλες οι περιοχές τοποθέτησης που υπάρχουν στη βάση δεδομένων.

Παράδειγμα αιτήματος

```
GET /perioxh
```

Παράδειγμα απάντησης (μερική λόγο μεγάλου πλήθους γραμμών)

```
[
    "Μουσικό Σχολείο Γιαννιτσών",
    "Μουσικό Σχολείο Δυτικής Λέσβου",
    "Μουσικό Σχολείο Πρέβεζας",
    "Μουσικό Σχολείο Χίου",
    "Μουσικό Σχολείο Πειραιά",
    "Μουσικό Σχολείο Αγρινίου",
    "Μουσικό Σχολείο Αλίμου",
    "Μουσικό Σχολείο Ηρακλείου",
    "Μουσικό Σχολείο Ζακύνθου",
    "Μουσικό Σχολείο Λασιθίου",
    ...
]
```

GET /dieythynsh

Επιστρέφονται όλες οι διευθύνσεις εκπαίδευσης που υπάρχουν στη βάση δεδομένων.

Παράδειγμα αιτήματος

```
GET /dieythynsh
```

Παράδειγμα απάντησης (μερική λόγο μεγάλου πλήθους γραμμών)

```
[
    "ΔΕ Α ΑΘΗΝΑΣ",
    "ΔΕ Α ΘΕΣΣΑΛΟΝΙΚΗΣ",
    "ΔΕ ΑΙΤΩΛΟΑΚΑΡΝΑΝΙΑΣ",
    "ΔΕ ΑΝΑΤΟΛΙΚΗΣ ΑΤΤΙΚΗΣ",
    "ΔΕ ΑΡΓΟΛΙΔΑΣ",
    "ΔΕ ΑΡΚΑΔΙΑΣ",
    "ΔΕ ΑΡΤΑΣ",
    "ΔΕ ΑΧΑΪΑΣ",
    "ΔΕ Β ΑΘΗΝΑΣ",
    "ΔΕ Β ΘΕΣΣΑΛΟΝΙΚΗΣ",
    ...
]
```

GET /search?typos={typ}klados={klad}perioxh={per}dieythynsh={diey}

Επιστρέφει το πλήθος προσλήψεων αναπληρωτών για κάθε σχολική χρονιά, με βάση τους παραμέτρους αναζήτησης `typos`, `klados`, `perioxh` και `dieythynsh`. Εάν δοθούν ταυτόχρονα `klados` και `perioxh`, επιστρέφεται επιπλέον dataset που περιέχει τα ελάχιστα μόρια πρόσληψης ανά χρονιά για κάθε συνδυασμό. Η μορφή των δεδομένων είναι συμβατή με το Chart.js.

Παράδειγμα αιτήματος

```
GET /search?typos=ΓΕΝΙΚΗΣ&klados=ΠΕ70&perioxh=Β+ΑΘΗΝΑΣ
&perioxh=Β+ΘΕΣΣΑΛΟΝΙΚΗΣ&perioxh=ΛΑΡΙΣΑΣ&perioxh=Α+ΠΕΙΡΑΙΑ
```

Παράδειγμα απάντησης (απόσπασμα)

```
{
  "anaplirotesCount": {
    "labels": [
      "2018-2019",
      "2020-2021",
      "2021-2022",
      "2022-2023"
    ],
    "datasets": [
      {
        "label": "Β ΑΘΗΝΑΣ, ΓΕΝΙΚΗΣ, ΠΕ70",
        "data": [106, 224, 183, 196]
      },
      ...
    ]
  },
  "minMoria": {
    "labels": [
      "2018-2019",
      "2020-2021",
      "2021-2022",
      "2022-2023"
    ],
    "datasets": [
      {
        "label": "Β ΑΘΗΝΑΣ, ΓΕΝΙΚΗΣ, ΠΕ70",
        "data": [1.767, 92.93, 49.28, 21.98]
      },
      ...
    ]
  }
}
```

Κεφάλαιο 3

Τεχνολογίες

3.1 Εισαγωγή

Το τρίτο κεφάλαιο παρουσιάζει τις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής, καθώς η συγκεκριμένη πτυχιακή καλύπτει δύο τελείως διαφορετικά πεδία. Αρχικά, για την ανάπτυξη της διεπαφής χρήστη σε κινητές συσκευές αξιοποιήθηκε η γλώσσα JavaScript σε συνδυασμό με το framework React Native, δίνοντας τη δυνατότητα στους χρήστες να έχουν πρόσβαση στην εφαρμογή και από Android συσκευές.

Για την αυτόματη ανάκτηση των δεδομένων επιλέχθηκε η Python, αξιοποιώντας το framework Selenium για την πλοήγηση και συλλογή των αρχείων. Στη συνέχεια, η επεξεργασία και η οργάνωση αυτών των δεδομένων έγινε με την βοήθεια της βιβλιοθήκης Pandas, η οποία αποδείχθηκε ιδανική για την καθαριότητα και τον μετασχηματισμό των αρχείων.

Τέλος, για να μπορέσει η εφαρμογή να επικοινωνήσει με την υπάρχουσα βάση δεδομένων, υλοποιήθηκαν SQL ερωτήματα που εξασφαλίζουν τη σωστή αποθήκευση και ανάκτηση των απαιτούμενων πληροφοριών.

3.2 Python

Η γλώσσα προγραμματισμού Python σχεδιάστηκε από τον Guido Van Rossum το 1991 και αναπτύχθηκε από την Python Software Foundation, βασισμένη στη προστακτική γλώσσα προγραμματισμού ABC. Η Python κατατάσσεται στις γλώσσες υψηλού επιπέδου, καθώς λειτουργεί σε σε πολλαπλά λειτουργικά συστήματα όπως Windows, Linux και Mac OS, χωρίς να χρειάζεται τροποποιήσεις. Παρόλο που είναι ανοιχτού κώδικα με μια μεγάλη κοινότητα που την συντηρεί, τα πνευματικά δικαιώματα παραμένουν στην Python Institute. Η φύση του ανοιχτού κώδικα επιτρέπει την μη περιορισμένη χρήση, τροποποίηση και διανομή, ακόμα και για εμπορικούς σκοπούς διασφαλίζοντας την συνεχή ανάπτυξη του λογισμικού[9].

Η Python χρησιμοποιεί τον διερμηνέα για την εκτέλεση της κάθε γραμμής κώδικα, αλλά και εκμεταλλεύεται την διαδικασία της αφαίρεσης κάνοντας χρήση της αντικειμενοστρέφειας. Προσφέρει την επιλογή στους προγραμματιστές να διαμοιράσουν μεγάλα και πολύπλοκα προβλήματα, σε επιμέρους μικρότερα με την μορφή αντικειμένων. Υποστηρίζει, επίσης, καταμέτρηση

αναφοράς αντικειμένων και μηχανισμούς ανίχνευσης κύκλου ζωής, σε συνδυασμό με συλλογή απορριμμάτων (garbage collector) για την βέλτιστη διαχείριση μνήμης. Το κύριο χαρακτηριστικό της είναι το DNS (Dynamic name resolution) το οποίο επιλύει ονόματα μεθόδων και μεταβλητών κατά την εκτέλεση[9].

Η Python ανήκει στις ευκολότερες και πιο προσβάσιμες γλώσσες προγραμματισμού, καθώς η σύνταξη του κώδικα θεωρείται πιο ευανάγνωστη σε σχέση με άλλες γλώσσες (π.χ. Java, C, JavaScript), όπως για παράδειγμα με τη λειτουργία της αφαίρεσης των ερωτηματικών και της αντικατάστασης των αγκυλών με τις εσοχές. Λόγω αυτών των χαρακτηριστικών και της γενικής δομής της, η Python είναι πιο προσιτή για αρχάριους προγραμματιστές, αλλά και πιο προτιμητέα από επαγγελματίες, με αποτέλεσμα να ξεπερνά άλλες δημοφιλείς και εύχρηστες γλώσσες [10] [9].

Η Python, όπως και πολλές άλλες γλώσσες (Java, C++, C) έχουν τους πρωταρχικούς τύπους μεταβλητών (primitive data types), όπως ο int, long και char, στους οποίους είναι απαραίτητη η αναφορά του τύπου τους κατά τη δήλωση τους. Μία διαφορά που υπάρχει στη διαχείριση των μεταβλητών στην Python, είναι ότι ο τύπος της μεταβλητής που αρμόζει σε κάθε περίπτωση, δηλώνεται κατά τη στιγμή που ανατίθεται τιμή σε αυτή και όχι νωρίτερα[9].

Στην Python όπως αναφέρθηκε πάνω, είναι δυνατή και η χρήση αντικειμενοστρέφειας. Προσφέρει την επιλογή στους προγραμματιστές να διαμοιράσουν μεγαλύτερά, πιο πολύπλοκα προβλήματα σε επιμέρους μικρότερα με την μορφή αντικειμένων. Υποστηρίζοντας καταμέτρηση αναφοράς αντικειμένων και μηχανισμούς ανίχνευσης κύκλου ζωής σε συνδυασμό με συλλογή απορριμμάτων(garbage collector) για την βέλτιστη διαχείριση μνήμης. Το κύριο χαρακτηριστικό είναι το DNS (Dynamic name resolution) το οποίο επιλύει ονόματα μεθόδων και μεταβλητών κατά την εκτέλεση[9].

Η Pandas είναι μια βιβλιοθήκη που έχει αλλάξει τον τρόπο που δουλεύουμε με δεδομένα στην Python. Φτιαγμένη από τον Wes McKinney το 2008, έχει γίνει το αγαπημένο εργαλείο πολλών αναλυτών. Δουλεύει κυρίως με δύο τύπους δεδομένων: τη Series, που μοιάζει με μια στήλη σε πίνακα, και το DataFrame, που είναι σαν ένας ολόκληρος πίνακας με γραμμές και στήλες. Αυτό που κάνει την Pandas τόσο δημοφιλή είναι πόσο εύκολα μπορεί να διαβάσει και να γράψει διάφορα είδη αρχείων - από απλά CSV μέχρι Excel, HTML και βάσεις δεδομένων SQL. Είναι πολύ καλή στο να καθαρίζει ”βρώμικα” δεδομένα, να συμπληρώνει κενά και να μετατρέπει τα δεδομένα στη μορφή που χρειαζόμαστε. Επίσης, μας επιτρέπει να ομαδοποιούμε και να συνοψίζουμε μεγάλα σύνολα δεδομένων με απλές εντολές. Το καλύτερο είναι ότι συνεργάζεται τέλεια με άλλες βιβλιοθήκες όπως η NumPy για υπολογισμούς και η Matplotlib για γραφήματα, οπότε μπορούμε να κάνουμε όλη τη δουλειά μας χωρίς να αλλάζουμε εργαλεία [**?**].

Το Web Scraping αποτελεί μια ιδιαίτερα χρήσιμη τεχνική για τη συλλογή μεγάλου όγκου δεδομένων από το διαδίκτυο. Η διαδικασία βασίζεται στην αυτόματη πλοήγηση σε ιστοσελίδες, από τις οποίες εξάγονται και αποθηκεύονται χρήσιμες πληροφορίες σε οργανωμένες μορφές. Τα δεδομένα αυτά μπορούν να αξιοποιηθούν τόσο από ιδιώτες όσο και από οργανισμούς, ανάλογα με τον σκοπό και τις ανάγκες τους.

Η Python διευκολύνει σημαντικά τη διαδικασία του web scraping χάρη σε μια σειρά βιβλιοθηκών που έχουν αναπτυχθεί γι’ αυτό τον σκοπό. Ανάμεσά τους ξεχωρίζουν τα PythonRequest, BeautifulSoup, MechanicalSoup και Selenium. Αυτά τα εργαλεία χρησιμοποιούνται ευρέως σε

εφαρμογές όπως η παρακολούθηση τιμών προϊόντων, η ανάλυση δεδομένων, η μελέτη κοινωνικών τάσεων και η υποστήριξη σε έργα μηχανικής μάθησης.

Στην ουσία, οι τεχνικές scraping ενισχύουν σημαντικά την ικανότητα πρόσβασης και αξιοποίησης πληροφορίας, επιτρέποντας την εξαγωγή δεδομένων με συστηματικό και αποδοτικό τρόπο.

3.3 Javascript

Η JavaScript είναι μια δημοφιλής γλώσσα προγραμματισμού [11] που αρχικά σχεδιάστηκε για τον εμπλουτισμό των ιστοσελίδων με διαδραστικά στοιχεία. Σύμφωνα τον Flanagan, η JavaScript ορίζεται ως μια δια-πλατφορμική, αντικειμενοστρεφής γλώσσα scripting, η οποία χρησιμοποιείται για να κάνει τις ιστοσελίδες δυναμικές και διαδραστικές [12]. Σε μια τυπική εφαρμογή ιστού, το HTML ορίζει τη δομή και το περιεχόμενο, το CSS τη μορφοποίηση και εμφάνιση, ενώ η JavaScript προσθέτει την αλληλεπίδραση και επιτρέπει τη δημιουργία πλούσιων web εφαρμογών. Η σημασία της JavaScript στον σύγχρονο προγραμματισμό είναι τεράστια, καθώς αποτελεί μία από τις βασικές τεχνολογίες του παγκόσμιου ιστού και εκτελείται σε όλους τους σύγχρονους φυλλομετρητές (Chrome, Firefox, Safari κλπ.)[13].

Ένα χαρακτηριστικό της JavaScript είναι ότι μπορεί να τρέχει και πέρα από τον την πλευρά του χρήστη. Υπάρχουν υλοποιήσεις της στην πλευρά του διακομιστή, με πιο γνωστό το περιβάλλον Node.js, που επιτρέπει τη χρήση της JavaScript για server-side προγραμματισμό (π.χ. υλοποίηση web servers, αλληλεπίδραση με βάσεις δεδομένων κ.ά.)Η ευελιξία και η δύναμη της JavaScript οδήγησαν στην ανάπτυξη πολλών frameworks και βιβλιοθηκών που βασίζονται σε αυτήν, με σκοπό να διευκολύνουν τους προγραμματιστές στην κατασκευή σύνθετων εφαρμογών. Τα frameworks αυτά παρέχουν προκαθορισμένη δομή και έτοιμες λειτουργίες, ώστε να απλοποιείται η ανάπτυξη εφαρμογών μεγάλης κλίμακας. Ένα από τα δημοφιλέστερα frontend frameworks/βιβλιοθήκες JavaScript είναι η React: Μια βιβλιοθήκη ανοιχτού κώδικα που δημιουργήθηκε από τη Meta (Facebook) για την κατασκευή διεπαφών χρήστη. Η React εισήγαγε την αρχιτεκτονική βασισμένη σε components (επαναχρησιμοποιούμενα συστατικά UI) και υιοθετεί δηλωτικό προγραμματισμό για τον ορισμό του UI, καθιστώντας ευκολότερη τη διαχείριση της κατάστασης και τον εκσυγχρονισμό δυναμικών σελίδων Έτσι, η ίδια γλώσσα χρησιμοποιείται τόσο στο frontend όσο και στο backend, συμβάλλοντας στην ενοποίηση του οικοσυστήματος ανάπτυξης εφαρμογών. Η JavaScript είναι δυναμική και πολυπαραδειγματική γλώσσα(multi-paradigm language), υποστηρίζει διαφορετικά παραδείγματα προγραμματισμού, όπως αντικειμενοστρεφή και συναρτησιακό προγραμματισμό – γεγονός που την καθιστά ευέλικτη για ποικίλες χρήσεις. Επιπλέον, δεν απαιτεί μεταγλώττιση (compile) πριν την εκτέλεση, καθώς είναι συνήθως ερμηνευόμενη (interpreted) ή JIT-μεταγλωττισμένη γλώσσα, κάτι που διευκολύνει την άμεση εκτέλεση κώδικα στον ιστό. Συνολικά, η JavaScript, λόγω της ευρείας υιοθέτησής της και του πλούσιου οικοσυστήματος εργαλείων και βιβλιοθηκών, θεωρείται απαραίτητο εργαλείο στον σύγχρονο προγραμματισμό εφαρμογών[13].

Η ευελιξία και η δύναμη της JavaScript οδήγησαν στην ανάπτυξη πολλών frameworks και βιβλιοθηκών που βασίζονται σε αυτήν, με σκοπό να διευκολύνουν τους προγραμματιστές στην κατασκευή σύνθετων εφαρμογών. Τα frameworks αυτά παρέχουν προκαθορισμένη δομή και

έτοιμες λειτουργίες, ώστε να απλοποιείται η ανάπτυξη εφαρμογών μεγάλης κλίμακας.

3.3.1 React

Η React περιγράφεται επίσημα ως μια βιβλιοθήκη JavaScript για την κατασκευή διεπαφών χρήστη (User Interfaces). Σε αντίθεση με αλλα frameworks, όπου η ενημέρωση του περιεχομένου μιας σελίδας γινόταν με άμεσες μεταβολές στη DOM, Η React εισάγει το Virtual DOM: μια εικονική αναπαράσταση του DOM στη μνήμη. Κάθε φορά που αλλάζει η κατάσταση (state) της εφαρμογής, Η React υπολογίζει αποδοτικά ποιες αλλαγές χρειάζονται στο Virtual DOM και ενημερώνει στοχευμένα το πραγματικό DOM, αντί να το αναδομεί εξ ολοκλήρου. Αυτή η προσέγγιση βελτιώνει την απόδοση και κάνει την ανάπτυξη δηλωτική – ο προγραμματιστής δηλώνει πώς πρέπει να φαίνεται η διεπαφή για μια δεδομένη κατάσταση, και Η React αναλαμβάνει να ενημερώσει το UI όταν τα δεδομένα μεταβληθούν[14].

Ένα από τα βασικά πλεονεκτήματα της React είναι η component-based αρχιτεκτονική του. Η διεπαφή «σπάει» σε μικρά, επαναχρησιμοποιούμενα components (π.χ. κουμπιά, φόρμες, λίστες), το καθένα με τη δική του λογική και κατάσταση. Τα components μπορούν να συνδυαστούν ιεραρχικά για να δημιουργήσουν πιο σύνθετες δομές UI. Αυτή η απομόνωση της λογικής σε μικρές μονάδες κάνει τον κώδικα πιο οργανωμένο και ευκολότερο στη συντήρηση. Επιπλέον, Η React είναι επεκτάσιμο και μπορεί να συνεργαστεί με άλλες βιβλιοθήκες ή frameworks. Για παράδειγμα, η διαχείριση κατάστασης μπορεί να γίνει με πρόσθετες βιβλιοθήκες όπως το Redux, ενώ η δρομολόγηση (routing) σε μια μονοσέλιδη εφαρμογή υλοποιείται συνήθως με Η React Router[14].

Ένα σημαντικό σύνθημα της φιλοσοφίας της React είναι «Learn Once, Write Anywhere». Αυτό σημαίνει ότι οι γνώσεις που αποκτά ένας προγραμματιστής ση React μπορούν να εφαρμοστούν σε πολλαπλά περιβάλλοντα. Παρότι η React ξεκίνησε ως βιβλιοθήκη για τον ιστό (browser), η ίδια λογική των components και της δηλωτικής απόδοσης UI μπορεί να χρησιμοποιηθεί και εκτός του φυλλομετρητή. Όπως αναφέρει η επίσημη τεκμηρίωση, η React μπορεί να κάνει render στον διακομιστή (Server-Side Rendering με Node.js) αλλά και να τροφοδοτήσει εφαρμογές για κινητά μέσω της React Native. Αυτό ανοίγει τον δρόμο για τη χρήση της JavaScript και της React στην ανάπτυξη native mobile εφαρμογών, γεγονός που θα εξετάσουμε στη συνέχεια[14].

3.3.2 React Native & Expo

Η React Native είναι ένα framework που επιτρέπει την ανάπτυξη native εφαρμογών για κινητά (Android, iOS, κ.ά.) χρησιμοποιώντας JavaScript και React. Με η React Native, οι προγραμματιστές μπορούν να γράψουν κώδικα σε JavaScript/JSX, ο οποίος όμως τελικά μεταφράζεται σε εγγενή στοιχεία διεπαφής (native UI components) στις αντίστοιχες πλατφόρμες. Ουσιαστικά, η React Native φέρνει τα καλύτερα στοιχεία της React στην ανάπτυξη για κινητά, προσφέροντας μια εμπειρία “learn once, write anywhere” και στο πεδίο των εφαρμογών για κινητές συσκευές[15].

Σύμφωνα με την επίσημη τεκμηρίωση, ο κώδικας που γράφεται σε JavaScript εκτελείται μέσω μιας μηχανής JavaScript στο κινητό, και τα React primitives (τα βασικά δομικά στοιχεία της

React, όπως τα components View, Text, Image κ.ά.) αποδίδονται σε native στοιχεία της πλατφόρμας. Με άλλα λόγια, μια <Text> komponent στη React Native θα εμφανιστεί ως native στοιχείο κειμένου στο Android (TextView) ή στο iOS (UILabel). Έτσι, η εφαρμογή που προκύπτει είναι πραγματικά native και χρησιμοποιεί τα ίδια API του λειτουργικού συστήματος που θα χρησιμοποιούσε μια εφαρμογή γραμμένη σε Java ή Kotlin για Android, ή σε Swift/Objective-C για iOS. Αυτή η προσέγγιση προσφέρει κατά κανόνα υψηλή απόδοση και εμπειρία χρήστη αντίστοιχη με των εφαρμογών που υλοποιούνται απευθείας με τις native γλώσσες[16].

Ένα από τα μεγάλα οφέλη της React Native είναι η πολλαπλή πλατφόρμα (cross-platform) ανάπτυξης. Μπορεί κανείς να μοιράζεται το μεγαλύτερο μέρος του κώδικα μεταξύ iOS και Android εφαρμογών. Η React Native παρέχει ένα βασικό σύνολο από platform-agnostic native components (δηλαδή components που λειτουργούν ανεξάρτητα από την πλατφόρμα) όπως τα View, Text, Image, τα οποία χαρτογραφούνται στα αντίστοιχα native building blocks κάθε συστήματος. Η επιχειρηματική λογική και η διαμόρφωση του UI γράφονται μία φορά και την React Native αναλαμβάνει να τις εμφανίσει σωστά σε κάθε λειτουργικό. Φυσικά, υπάρχει δυνατότητα ανίχνευσης της πλατφόρμας και χρήσης πλατφορμο-ειδικού κώδικα όταν χρειάζεται, αλλά στις περισσότερες περιπτώσεις το ίδιο component δουλεύει και στα δύο[16].

Επειδή η React Native χρησιμοποιεί React στον πυρήνα της, η ανάπτυξη εφαρμογών μοιάζει αρκετά με την ανάπτυξη React web εφαρμογών. Ο προγραμματιστής γράφει JSX, διαχειρίζεται state και props των components, και μπορεί να χρησιμοποιήσει hooks ή άλλες React τεχνικές. Η μεγάλη διαφορά είναι ότι αντί για HTML elements χρησιμοποιούνται components της React Native, και η εφαρμογή τρέχει σε κινητή συσκευή αντί σε φυλλομετρητή. Το οικοσύστημα της React Native είναι επίσης πλούσιο, με πολλές βιβλιοθήκες διαθέσιμες για πρόσθετη λειτουργικότητα (π.χ. για πλοήγηση μεταξύ οθονών, διαχείριση κατάστασης, πρόσβαση σε APIs συσκευής όπως κάμερα, GPS, αποθήκευση κ.λπ.). Σε περιπτώσεις που απαιτείται λειτουργικότητα εκτός των παρεχόμενων δυνατοτήτων της JavaScript, η React Native επιτρέπει την επέκταση με native modules, δηλαδή, μπορεί κάποιος να γράψει κομμάτια κώδικα σε Java/Swift και να τα γεφυρώσει ώστε να καλούνται από τον JavaScript κώδικα.

Η React Native από μόνη της εστιάζει στον πυρήνα της απόδοσης του UI και δεν καθορίζει αυστηρά τον τρόπο υλοποίησης ορισμένων λειτουργιών υψηλού επιπέδου. Για παράδειγμα, δεν επιβάλλει κάποιο συγκεκριμένο σύστημα πλοήγησης (routing) ή διαχείρισης κατάστασης, οι προγραμματιστές επιλέγουν τις βιβλιοθήκες που ταιριάζουν καλύτερα στις ανάγκες τους (π.χ. React Navigation για πλοήγηση). Για να διευκολυνθεί όμως η συνολική διαδικασία ανάπτυξης mobile εφαρμογών, έχουν δημιουργηθεί εργαλεία που συμπληρώνουν την React Native. Ένα από τα πιο σημαντικά είναι η Expo, η οποίο μάλιστα προτείνεται επίσημα από την ομάδα της React Native ως η καλύτερη αφετηρία για να ξεκινήσει κανείς ένα νέο έργο mobile εφαρμογής[15].

Η Expo είναι ένα framework και ένα πακέτο εργαλείων που έχει σκοπό να κάνει ακόμη πιο εύκολη την ανάπτυξη εφαρμογών για κινητές συσκευές με χρήση της React Native. Η Expo κάθεται πάνω από την React Native και προσφέρει ένα ολοκληρωμένο οικοσύστημα για τη δημιουργία, δοκιμή και διανομή εφαρμογών, χωρίς να απαιτείται από τον προγραμματιστή να διαχειριστεί εξαρχής τις περιπλοκές των native build tools. Η Expo είναι ένα framework ανοιχτού κώδικα που κάνει την ανάπτυξη Android και iOS εφαρμογών πιο εύκολη, παρέχοντας χαρακτηριστικά όπως δρομολόγηση βασισμένη στο σύστημα αρχείων (file-based routing), μια

στάνταρ βιβλιοθήκη από native modules, και πολλά άλλα, όλα έτοιμα προς χρήση. Η φιλοσοφία του Expo είναι να αυτοματοποιήσει τα δύσκολα μέρη της διαδικασίας, επιτρέποντας στους δημιουργούς εφαρμογών να επικεντρωθούν περισσότερο στη γραφή του κώδικα της εφαρμογής τους και λιγότερο στη διαμόρφωση εργαλείων ή στην αντιμετώπιση προβλημάτων συμβατότητας[17].

Η Expo επιτρέπει γρήγορη εκκίνηση και ανάπτυξη μέσω του CLI (expo init), δημιουργώντας ένα νέο React Native project χωρίς την ανάγκη ρυθμίσεων σε Xcode ή Android Studio. Με την εφαρμογή Expo Go, ο προγραμματιστής μπορεί να δει ζωντανά τις αλλαγές στον κώδικα με τη λειτουργία Fast Refresh. Παράλληλα, προσφέρει πλούσιο SDK με ενσωματωμένα modules για κάμερα, βίντεο, αισθητήρες, push notifications και άλλες λειτουργίες, διευκολύνοντας την ανάπτυξη χωρίς την ανάγκη επιπλέον native plugins. Η Expo επίσης επιτρέπει την επέκταση με custom native κώδικα μέσω εξαγωγής (ejecting) ή development builds, και προσφέρει υπηρεσίες για εύκολη διανομή και ενημερώσεις μέσω του EAS. Με υποστήριξη για over-the-air updates και εργαλεία όπως το sentry, παρέχει γρήγορη διόρθωση σφαλμάτων και υποστήριξη μέσω μιας ισχυρής κοινότητας προγραμματιστών[17] .

3.4 MySQL

Η SQL (Structured Query Language) αναπτύχθηκε τη δεκαετία του 1970 από την IBM, αρχικά με το όνομα SEQUEL. Καθιερώθηκε ως διεθνές πρότυπο (ANSI/ISO) το 1986-1987 και έγινε μια απο τις κυρίαρχε γλώσσες[18] για επικοινωνία με σχεσιακές βάσεις δεδομένων, επιτρέποντας την αποθήκευση, ανάκτηση και διαχείριση δομημένων δεδομένων.

Η MySQL γεννήθηκε όταν προγραμματιστές αναζητούσαν μια πιο αποδοτική λύση από την υπάρχουσα mSQL. Αρχικά σχεδίαζαν να χρησιμοποιήσουν το mSQL για σύνδεση στους πίνακές τους μέσω δικών τους ρουτινών χαμηλού επιπέδου (ISAM), αλλά διαπίστωσαν ότι δεν ήταν αρκετά γρήγορη ή ευέλικτη. Έτσι, δημιούργησαν μια νέα διεπαφή SQL με API παρόμοιο με το mSQL για εύκολη μεταφορά κώδικα [19].

Το όνομα ”MySQL” προέρχεται από την κόρη του συνιδρυτή Monty Widenius, τη My, ενώ το λογότυπο της, ένα δελφίνι ονόματι ”Sakila”, επιλέχθηκε μέσω διαγωνισμού και προτάθηκε από τον Ambrose Twebaze από το Εσουατίνι [19].

Η MySQL έγινε μία από τις πολλές ”διαλέκτους” SQL που, παρότι διατηρούν τον βασικό πυρήνα εντολών του προτύπου, προσθέτουν επεκτάσεις για συγκεκριμένες ανάγκες, όπως συμβαίνει και με την T-SQL (Microsoft) και την PL/SQL (Oracle) [19].

Η αρχιτεκτονική πολλαπλών νημάτων (multithreaded) της MySQL επιτρέπει την αξιοποίηση όλων των διαθέσιμων πυρήνων του επεξεργαστή, κάτι που βελτιώνει σημαντικά την ταχύτητα επεξεργασίας δεδομένων. Παράλληλα, χρησιμοποιεί αποδοτικά συστήματα αποθήκευσης όπως το MyISAM, που οργανώνει τα δεδομένα σε δομές B-tree και συμπιέζει τα ευρετήρια (indexes) για εξοικονόμηση χώρου και ταχύτερη πρόσβαση. Όταν χρειάζεται να συνδυάσει δεδομένα από διαφορετικούς πίνακες, η MySQL χρησιμοποιεί έξυπνους αλγόριθμους συνένωσης (joins) που βρίσκουν γρήγορα τις σχετικές πληροφορίες. Επιπλέον, δημιουργεί προσωρινούς hash tables στη μνήμη RAM αντί στο δίσκο, επιταχύνοντας έτσι σημαντικά την εκτέλεση πολύπλοκων ερωτημάτων [20].

Ένα από τα σημαντικότερα πλεονεκτήματα της MySQL είναι η δυνατότητα κλιμάκωσης - (scalability) που προσφέρει. Μπορεί να διαχειριστεί τεράστιες βάσεις δεδομένων με δισεκατομμύρια εγγραφές (records), έως και 5 δισεκατομμύρια σύμφωνα με την επίσημη τεκμηρίωση. Κάθε πίνακας μπορεί να έχει μέχρι 64 διαφορετικά ευρετήρια, επιτρέποντας γρήγορη αναζήτηση με πολλούς τρόπους. Χάρη σε αυτά τα χαρακτηριστικά, η MySQL προσαρμόζεται εύκολα σε διαφορετικές ανάγκες και μεγέθη εφαρμογών, από ένα απλό blog μέχρι πολύπλοκα enterprise systems με εκατομμύρια χρήστες [20].

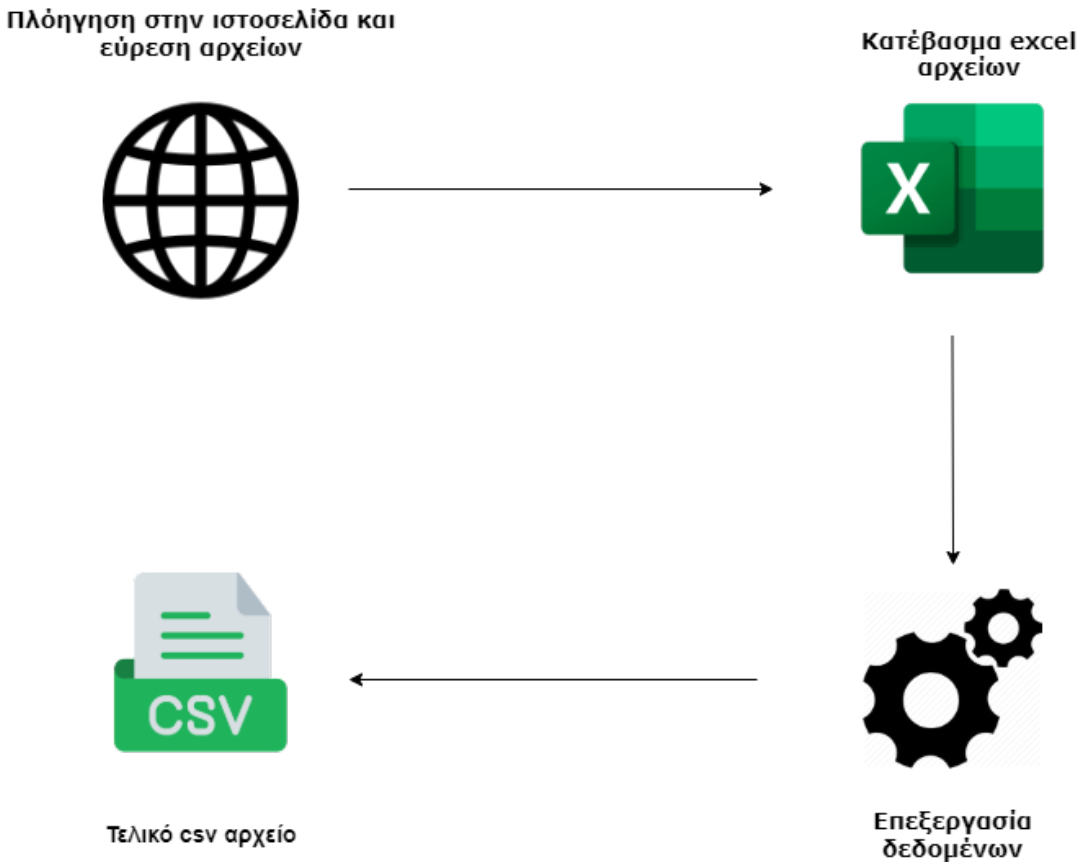
Η συνδεσιμότητα αποτελεί άλλο ένα ισχυρό χαρακτηριστικό της MySQL. Προσφέρει ευελιξία στον τρόπο επικοινωνίας με τη βάση δεδομένων, υποστηρίζοντας διάφορα πρωτόκολλα σύνδεσης, από το καθιερωμένο TCP/IP για δικτυακές συνδέσεις μέχρι πιο εξειδικευμένες μεθόδους όπως τα named pipes και shared memory στα Windows ή τα Unix sockets στα συστήματα τύπου Linux. Οι προγραμματιστές μπορούν να επικοινωνήσουν με τη MySQL χρησιμοποιώντας σχεδόν οποιαδήποτε γλώσσα προγραμματισμού προτιμούν, καθώς διαθέτει επίσημα APIs για C, C++, Java, Perl, PHP, Python, Ruby και πολλές άλλες. Επιπλέον, η υποστήριξη για τα πρότυπα ODBC και JDBC διευκολύνει τη σύνδεση με επιχειρηματικές εφαρμογές και εργαλεία αναφορών, κάνοντας τη MySQL προσιτή σε προγραμματιστές με διαφορετικές δεξιότητες και προτιμήσεις [20].

Τέλος, ενα από τα πιο ξεχωριστά χαρακτηριστικά της είναι η υποστήριξη διαφορετικών μηχανών αποθήκευσης (storage engines), που επιτρέπει στους χρήστες να επιλέξουν τον τρόπο που θα αποθηκεύονται και θα προσπελαύνονται τα δεδομένα τους. Για παράδειγμα, μπορούν να επιλέξουν μεταξύ συναλλακτικής (transactional) λειτουργίας με την InnoDB για εφαρμογές που απαιτούν αξιοπιστία και ακεραιότητα δεδομένων, ή μη-συναλλακτικής (non-transactional) με τη MyISAM για μεγαλύτερη ταχύτητα σε περιπτώσεις που προτεραιότητα έχει η απόδοση. Επιπλέον, η αρχιτεκτονική της επιτρέπει την εύκολη προσθήκη νέων λειτουργιών μέσω επεκτάσεων (extensions) και πρόσθετων (plugins), δίνοντας τη δυνατότητα στους διαχειριστές βάσεων δεδομένων να προσαρμόσουν το σύστημα στις συγκεκριμένες ανάγκες του έργου τους [20].

Κεφάλαιο 4

Web service για την αυτόματη ανάκτηση και προ-επεξεργασία αρχείων προσλήψεων

4.1 Αρχιτεκτονική



Σχήμα 4.1: Κύκλος ζωής/αρχιτεκτονική προγράμματος

Στο πρώτο στάδιο της ροής, το σύστημα πραγματοποιεί την εξαγωγή (Extract) των δεδομένων μέσω της βιβλιοθήκης Selenium WebDriver. Η εφαρμογή πλοηγείται αυτόματα στην ιστοσελίδα του Υπουργείου Παιδείας και ανακτά τα διαθέσιμα αρχεία τύπου Excel και RAR που αφορούν προσλήψεις αναπληρωτών. Παράλληλα, η εφαρμογή συνδέεται με τη βάση δεδομένων MySQL και ανακτά τη μέγιστη ημερομηνία καταγραφής προσλήψεων, προκειμένου να συγκρίνει τα νέα δεδομένα με τα ήδη αποθηκευμένα και να αποτρέψει την επεξεργασία διπλότυπων αρχείων. Η λειτουργικότητα αυτή υλοποιείται στο module scraper_module.py, όπου περιλαμβάνονται επίσης επιμέρους διεργασίες όπως η αναγνώριση ημερομηνιών, η δυναμική μετονομασία των αρχείων και η αποσυμπίεση περιεχομένων.

Στη συνέχεια, τα δεδομένα που έχουν ληφθεί υφίστανται επεξεργασία μέσω του αρχείου preprocess_module.py. Η φάση αυτή περιλαμβάνει καθαρισμό και κανονικοποίηση των στηλών, μετατροπή τύπων αρχείων, αφαίρεση αχρήστων εγγραφών και εμπλουτισμό των δεδομένων με νέες πληροφορίες όπως ημερομηνίες, ωράρια ή κατηγορίες εκπαιδευτικών. Η χρήση βοηθητικών συναρτήσεων επιτρέπει την ευέλικτη και στοχευμένη επεξεργασία των δεδομένων με βάση τις ανάγκες του συστήματος.

4.2 Ανάκτηση δεδομένων

Η διαδικασία ανάκτησης δεδομένων έχει αναπτυχθεί με Python, χρησιμοποιώντας μια ποικιλία βιβλιοθηκών και εργαλείων που επιτρέπουν το scraping, το χειρισμό αρχείων και τον αυτοματισμό. Ο κώδικας ανακτά δομημένα δεδομένα από τον ιστότοπο του ελληνικού Υπουργείου Παιδείας, φιλτράρει σχετικά έγγραφα με βάση συγκεκριμένες λέξεις-κλειδιά και ημερομηνίες, κατεβάζει τα σχετικά αρχεία (σε μορφή .xlsx ή .rar), επεξεργάζεται τα ονόματα αρχείων προσθέτοντας σημασιολογικές πληροφορίες και εξάγει αρχεία όταν χρειάζεται. Αυτή η διαδικασία διασφαλίζει ότι κάθε ανακτηθέν σύνολο δεδομένων είναι εύκολα ανιχνεύσιμο και επαναχρησιμοποιήσιμο.

Σύνδεση στην βάση δεδομένων

Το αρχείο ”database.py” μαζί με το ”config.py” είναι υπεύθυνα για ότι διεργασίες σχετίζονται με την βάση δεδομένων ”eanaplirotes”.

Από την ”main.py”, καλείται η ”connect_to_db()” που ανήκει στο αρχείο ”database.py”. Η μέθοδος ”connect_to_db()” θέτει τις παραμέτρους ”host”, ”user”, ”password”, ”database” και ”port” στην ”mysql.connector.connect()” (έτοιμη μέθοδος που παρέχεται από την mysql) για την πετυχημένη σύνδεση στην βάση δεδομένων. Μέσα από την ”config.py” χρησιμοποιείται το λεξικό(Το dictionary στην Python παρέχει το ζεύγος κλειδί-τιμή) ”DATABASE_CONFIG” το οποίο για κάθε παράμετρο κλειδί θέτει και μια τιμή η οποία έρχεται ως ”environment variable”(αυτές οι μεταβλητές δίνουν παραπάνω ασφάλεια άμα ο κώδικας μοιράζεται από κάποια versioning πλατφόρμα μιας και τα ευαίσθητα στοιχεία δεν φαίνονται κάπου), όπως φαίνεται στο Απόσπασμα Κώδικα 4.1.

```
1 def connect_to_db():
2     return mysql.connector.connect(
3         host=DATABASE_CONFIG['host'],
4         user=DATABASE_CONFIG['user'],
5         password=DATABASE_CONFIG['password'],
6         database=DATABASE_CONFIG['database'],
7         port=DATABASE_CONFIG['port']
8     )
```

Απόσπασμα Κώδικα 4.1: Σύνδεση στην βάση δεδομένων

Εύρεση τελευταίας ημερομηνίας

Αφού πραγματοποιηθεί επιτυχής σύνδεση με την βάση δεδομένων, το επόμενο βήμα είναι να εντοπιστεί η πιο πρόσφατη ημερομηνία από την οποία ξεκινάει η διαδικασία της συλλογής δεδομένων. Αυτό επιτυγχάνεται με την χρήση της ”get_max_date()”, η οποία επίσης ανήκει στο αρχείο ”database.py”. Η συνάρτηση αυτή δημιουργεί σύνδεση μέσω του connection pool και εκτελεί ένα απλό SQL ερώτημα που επιστρέφει την μέγιστη τιμή της στήλης ”Hmeromnia” από τον πίνακα ”anaplirotes”, δηλαδή την πιο πρόσφατη ημερομηνία που έχει ήδη αποθηκευτεί στην βάση όπως φαίνεται στο απόσπασμα 4.2.
Η επιστροφή της τιμής γίνεται με ασφάλεια, ελέγχοντας αν όντως υπάρχει αποτέλεσμα μέσω μιας σύντομης λογικής συνθήκης. Αν δεν υπάρχει καταχωρημένη ημερομηνία, επιστρέφεται η τιμή ”None”. Η συνάρτηση κλείνει με ασφάλεια τον cursor και την βάση, επιστρέφοντας το αποτέλεσμα.
Στην ”main.py”, η ”get_max_date()” καλείται μέσα σε ένα try-finally block ώστε να διασφαλιστεί πως η σύνδεση με την βάση θα κλείσει ανεξαρτήτως σφαλμάτων. Η επιστρεφόμενη ημερομηνία (”max_date”) χρησιμοποιείται ως φίλτρο για την λειτουργία της συνάρτησης ”scrap()”, η οποία αναλαμβάνει να συλλέξει μόνο δεδομένα που είναι νεότερα από αυτήν, θα αναλυθεί εις βάθος παρακάτω.

```
1 def get_max_date():
2     connection = connection_pool.get_connection()
```

```
3     cursor = connection.cursor()
4
5     query = "SELECT MAX(Hmeromnia) FROM anaplirotes ;"
6     cursor.execute(query)
7     result = cursor.fetchall()
8
9     max_date = result[0][0] if result and result[0] and
    result[0][0] else None
10
11    cursor.close()
12    connection.close()
13
14    return max_date
```

Απόσπασμα Κώδικα 4.2: Συνάρτηση εύρεσης τελευταίας ημερομηνίας

Αρχικοποίηση του driver: driver_init()

Ένα θεμελιώδης βήμα είναι η αρχικοποίηση του web driver ώστε να ανοίξει το πρόγραμμα περιήγησης Chrome και έπειτα να κατέβουν τα αρχεία. Στο Απόσπασμα Κώδικα 4.3 παρουσιάζεται ο κώδικας.

```
1 def driver_init():
2     print("Creating Chrome Driver")
3
4     chrome_options = webdriver.ChromeOptions()
5     prefs = {"download.default_directory":
    r"C:\sxoli\preprocess_folder"}
6     chrome_options.add_experimental_option("prefs", prefs)
7     chrome_options.add_argument("--headless=new")  # comment it
    if you want headless=false
8     my_driver = webdriver.Chrome(options=chrome_options)
9     yield my_driver
10    print("Closing Chrome Driver")
11    my_driver.quit()
```

Απόσπασμα Κώδικα 4.3: Συνάρτηση αρχικοποίησης Chrome Driver

Η μέθοδος επιτυγχάνει τους εξής στόχους:

- **Φάκελος αποθήκευσης αρχείων**: Το πρόγραμμα περιήγησης προσαρμόζεται ώστε ο χρήστης να μπορεί να θέσει τον δικό του φάκελο που θα αποθηκεύονται τα αρχεία όταν κατεβαίνουν από την σελίδα.
- **Επιλογή "headless mode"**: Στην μέθοδο υπάρχει η δυνατότητα να εκτελείται χωρίς να ανοίγει η ιστοσελίδα, ιδανικό για περιβάλλον διακομιστή(server).

- **Διαχείριση κύκλου ζωής**: Η χρήση του ”yield” επιτρέπει την προσωρινή παράδοση του αντικείμενου WebDriver. Μόλις τελειώσει η περίοδος λειτουργίας, το πρόγραμμα περιήγησης κλείνει σωστά χρησιμοποιώντας quit(), διασφαλίζοντας ότι δεν υπάρχει καθυστέρηση διαδικασίας.

Γενική δομή και ροή εφαρμογής

Στον πυρήνα της, η λογική ανάκτησης δεδομένων είναι ενσωματωμένη στη μέθοδο scrap(date_to_compare), η οποία καταφέρνει την πλήρη συνεδρία ανάκτησης δεδομένων. Αυτή η μέθοδος διαχειρίζεται την πλοήγηση στον ιστό, την ανάλυση του DOM δέντρου, την αναγνώριση λέξεων-κλειδιών, τον εντοπισμό λήψης αρχείων, τη μετονομασία και την εξαγωγή αρχείων.

Περιήγηση και αναγνώριση περιεχομένου

Η αυτοματοποιημένη πλοήγηση εκτελείται χρησιμοποιώντας τον driver που λαμβάνεται από την ”driver_init()”, σε συνδυασμό με αναμονές(explicit waits) για να διασφαλιστεί η ετοιμότητα του στοιχείου. Ο scraper επισκέπτεται την πύλη ανακοινώσεων του Υπουργείου και περιηγείται σε έναν πίνακα συνδέσμων χρησιμοποιώντας ερωτήματα ”XPath”, όπως φαίνεται στο απόσπασμα 4.4.

```
driver_generator = driver_init()

    driver = next(driver_generator)

    driver.get("https://www.minedu.gov.gr/")

    driver.find_element("xpath",
   '//*[@id="zentools-1085"]/ul/li/ul/li[6]/a').click()
```

Απόσπασμα Κώδικα 4.4: Περιήγηση ιστοσελίδας

Το κείμενο που ανήκει στην Javascript ετικέτα κάθε σειράς κανονικοποιείται μέσω της ”normalize_greek_text()” μεθοδού για να μετατραπούν οι λέξεις σε αγγλικούς χαρακτήρες ώστε να αφαιρεθούν οι τόνοι, επίσης να μετατραπούν όλα τα γράμματα σε πεζά για ισχυρή αντιστοίχιση. Όταν ανιχνεύεται μια λέξη-κλειδί όπως ”προσλήψεις” (με αγγλικούς χαρακτήρες είναι ως ”proslepseis”), η συμβολοσειρά ημερομηνίας εξάγεται και αναλύεται σε μορφή ISO χρησιμοποιώντας ένα βοηθητικό πρόγραμμα που βασίζεται στην regex (κανονική έκφραση) όπως υπάρχει στην απόσπαση 4.5.

```
def normalize_greek_text(text):
    return unidecode(text).lower()

proslipsis_key_word = normalize_greek_text(a_tag_text)

if "proslepseis" in proslipsis_key_word or "proslepse" in
   proslipsis_key_word:
```

```
7     print("clicked: ", a_tag_text)
8     date_for_preprocess =
    locate_date_from_string_and_normalize_it(a_tag_text)
9     print("normalized date:", date_for_preprocess)
10    obj_date = datetime.strptime(date_for_preprocess,
    '%Y-%m-%d').date()
11    print("date object: ", obj_date)
```

Απόσπασμα Κώδικα 4.5: Αναγνώριση περιεχομένου και ανάθεση ημερομηνίας

Φιλτράρισμα και εξαγωγή περιεχομένου

Μόλις βρεθεί μια σχετική ανακοίνωση, ο αλγόριθμος ανοίγει την σελίδα σε νέα καρτέλα και την αναλύει. Εντοπίζονται σημασιολογικά στοιχεία όπως «πλήρους ωραρίου» ή «μειωμένου ωραρίου» για την κατηγοριοποίηση του τύπου απασχόλησης (ΑΠΩ/ΑΜΩ). Αυτά τα μετα-δεδομένα αργότερα ενσωματώνονται στο όνομα αρχείου για εύκολη ταξινόμηση.

Κατέβασμα αρχείων και λογική αναγνώρισης

Κατά την περιήγηση στις σελίδες του Υπουργείου, το σύστημα εντοπίζει αυτόματα τους συνδέσμους που οδηγούν σε αρχεία .xlsx ή .rar χρησιμοποιώντας κατάλληλες XPath εκφράσεις (δείτε απόσπασμα 4.6). Όταν βρεθούν αυτοί οι σύνδεσμοι, το πρόγραμμα προσομοιώνει το ”κλικ” μέσω του Selenium WebDriver, ώστε να ξεκινήσει η λήψη των αρχείων, όπως το απόσπασμα 4.7.

```
1  combined_xpath = "//a[contains(@href, '.rar') or
     contains(@href, '.xlsx')]"
2  links =
     wait.until(ec.presence_of_all_elements_located((By.XPATH,
     combined_xpath)))
```

Απόσπασμα Κώδικα 4.6: Σύνδεσμοι ".xlsx" και ".rar"

```
1  for link in unique_links:
2      if link.is_displayed() and link.is_enabled():
3          link.click()
4          time.sleep(2)
```

Απόσπασμα Κώδικα 4.7: Πάτημα "κλικ"

Για να βεβαιωθούμε ότι η λήψη έχει ολοκληρωθεί, εφαρμόζεται μια διαδικασία ελέγχου (polling), η οποία παρακολουθεί τον φάκελο λήψεων για λίγα δευτερόλεπτα μέχρι να εμφανιστεί το αρχείο. Ο φάκελος αυτός έχει ήδη οριστεί από την αρχικοποίηση του WebDriver, κάτι που απλοποιεί τη διαχείριση.
Αφού ολοκληρωθεί η λήψη, το αρχείο μετονομάζεται αυτόματα ώστε να περιλαμβάνει την ημερομηνία της ανακοίνωσης και, αν εντοπιστεί, την ένδειξη για το είδος της πρόσληψης (π.χ.

πλήρους ή μειωμένου ωραρίου). Αν το αρχείο είναι συμπιεσμένο .rar, αποθηκεύεται προσωρινά σε ξεχωριστό φάκελο, αποσυμπιέζεται, και τα αρχεία .xlsx που περιέχει μετονομάζονται κατάλληλα και αντιγράφονται ξανά στον φάκελο λήψεων για επεξεργασία.

Διαχείρηση ".rar" αρχείων

Τα ληφθέντα αρχεία εξάγονται χρησιμοποιώντας το "patoolib" και τα περιεχόμενά τους μετονομάζονται με βάση το σημασιολογικό πρόθεμα του γονικού αρχείου. Αυτή η στρατηγική επιτρέπει στις μεταγενέστερες διαδικασίες να συσχετίσουν κάθε σύνολο δεδομένων με τα μετάδεδομένα ανακοίνωσής του.

Ανθεκτικότητα και χειρισμός εξαιρέσεων

Καθ' όλη τη διάρκεια της διαδικασίας scraping, οι πιθανές εξαιρέσεις(exceptions), όπως το stale element DOM και τα χρονικά όρια(timeouts) αντιμετωπίζονται χρησιμοποιώντας μπλοκ "try/except". Τα ενημερωτικά μηνύματα και οι έξοδοι ανίχνευσης βοηθούν στον εντοπισμό σφαλμάτων απροσδόκητων αποτυχιών. Επιπλέον, η χρήση της μεθόδου(driver_init) για τη δημιουργία WebDriver βοηθά στον χειρισμό των sessions του προγράμματος περιήγησης σε μια ενιαία τοποθεσία, διασφαλίζοντας ότι θα κλείσει σωστά μετά τη χρήση.

Επαναληπτική πλοήγηση και τερματισμός

Εάν δεν βρεθεί η "ημερομηνία στόχος"(που έχει ληφθεί από την βάση δεδομένων) στην τρέχουσα σελίδα, το σύστημα πλοηγείται στην επόμενη σελίδα χρησιμοποιώντας το κουμπί «Επόμενο». Αυτός ο βρόχος συνεχίζεται μέχρι να βρεθεί ένα ταίριασμα ή να συμπληρωθεί το όριο σελιδοποίησης, οπότε ο scraper κλείνει και ο driver τερματίζεται.

4.3 Προ-επεξεργασία δεδομένων

Μετατροπή excel αρχείων σε CSV

Η διαδικασία προεπεξεργασίας ξεκινάει με τη μετατροπή όλων των αρχείων Excel (.xlsx) που βρίσκονται στον φάκελο που έχει δοθεί ως παράμετρος, σε ισοδύναμα αρχεία CSV. Αυτό επιτυγχάνεται μέσω της μεθόδου convert_excel_to_csv(), η οποία καλείται από την main.py και υλοποιείται στο αρχείο preprocess_module.py, όπως φαίνεται στο Απόσπασμα Κώδικα 4.3.

Πριν γίνει οποιαδήποτε επεξεργασία, αγνοούνται φύλλα του Excel των οποίων το όνομα περιέχει τις λέξεις-κλειδιά "ΣΤΑΤΙΣΤΙΚΑ" ή "ΣΤΑΤ", ανεξαρτήτως κεφαλαίων, καθώς θεωρούνται μη σχετικές ή συνοπτικές στατιστικές πληροφορίες που δεν συμβάλλουν στην ανάλυση.

Αρχικά, εντοπίζονται όλα τα αρχεία Excel μέσα στον συγκεκριμένο φάκελο. Για κάθε τέτοιο αρχείο, διαβάζονται όλα τα φύλλα που περιέχει σε μορφή dictionary (ένα ζεύγος με όνομα φύλλου και τα περιεχόμενά του). Στη συνέχεια, γίνεται φιλτράρισμα και αφαιρούνται τα φύλλα που περιέχουν στο όνομά τους τις παραπάνω λέξεις-κλειδιά.

Για κάθε φύλλο που παραμένει, δημιουργείται ένα νέο CSV αρχείο. Το όνομα του αρχείου διατηρεί το αρχικό όνομα του Excel αρχείου και προστίθεται το όνομα του φύλλου, ώστε να υπάρχει σαφής αντιστοίχιση. Η αποθήκευση πραγματοποιείται μόνο αν το φύλλο περιέχει δεδομένα (δηλαδή δεν είναι κενό), και το encoding που χρησιμοποιείται είναι UTF-8 ώστε να διατηρείται η υποστήριξη ελληνικών χαρακτήρων.

Τέλος, το αρχικό Excel αρχείο διαγράφεται αυτόματα μετά τη μετατροπή, ώστε να μην παραμένουν αχρείαστα αντίγραφα στο σύστημα. Η διαγραφή αυτή συμβάλλει επίσης στη μείωση του αποθηκευτικού όγκου που καταλαμβάνεται από τα αρχεία, ειδικά όταν η διαδικασία εφαρμόζεται μαζικά σε πολλούς φακέλους.

```
def convert_excel_to_csv(folder_path):
    exclude_sheet_substrings = [ "ΣΤΑΤΙΣΤΙΚΑ".lower(),
                                 "ΣΤΑΤ".lower()]

    for filename in os.listdir(folder_path):
        if filename.endswith(".xlsx"):
            input_path = os.path.join(folder_path, filename)

            excel_sheets = pd.read_excel(input_path,
                                         sheet_name=None)

            for sheet_name in list(excel_sheets.keys()):
                if any(substring in sheet_name.lower() for
                       substring in exclude_sheet_substrings):
                    del excel_sheets[sheet_name]

            for sheet_name, df in excel_sheets.items():
                output_filename = (os.path.splitext(filename)[0]
                                   + f"_{sheet_name}.csv")
                output_path = os.path.join(folder_path
                                           , output_filename)

                if not df.empty:
                    df.to_csv(output_path, index=False,
                              encoding='utf-8')

            os.remove(input_path)
```

Απόσπασμα Κώδικα 4.8: Μετατροπή αρχείων Excel σε CSV

Αφαίρεση κενών πριν και μετά τα κόμματα

Η συνάρτηση remove_empty_spaces_before_after_commas() αποσκοπεί στην καθολική επεξεργασία αρχείων CSV στον δοθέντα φάκελο, με στόχο την αφαίρεση τυχόν περιττών κενών χαρακτήρων πριν και μετά από τα κόμματα στα δεδομένα. Συνοπτικά, η διαδικασία υλοποιείται

ως εξής:

- **Αναγνώριση αρχείων CSV:** Ο κώδικας διατρέχει τον φάκελο, εντοπίζοντας όλα τα αρχεία που ολοκληρώνονται με το extension .csv.
- **Φόρτωση δεδομένων:** Για κάθε αρχείο CSV, κατασκευάζεται το πλήρες μονοπάτι και το αρχείο διαβάζεται σε ένα pandas DataFrame με κωδικοποίηση UTF-8. Αυτό εξασφαλίζει την ορθή ανάγνωση πιθανών ελληνικών χαρακτήρων.
- **Εφαρμογή της λειτουργίας strip():** Με τη χρήση της μεθόδου map() εφαρμόζεται η strip() σε κάθε στοιχείο του DataFrame που είναι τύπου str. Η λειτουργία αυτή αφαιρεί τα περιττά κενά πριν και μετά τα στοιχεία του κειμένου. Σε περίπτωση που προκύψει κάποιο σφάλμα κατά την εφαρμογή, ενεργοποιείται το αντίστοιχο μπλοκ except, το οποίο καταγράφει το σφάλμα μαζί με το όνομα του αρχείου για περαιτέρω διερεύνηση.
- **Αποθήκευση των τροποποιημένων δεδομένων:** Τέλος, το τροποποιημένο DataFrame αποθηκεύεται πάλι ως CSV αρχείο στον ίδιο φάκελο, αντικαθιστώντας το αρχικό αρχείο, με εξασφάλιση ότι η κωδικοποίηση παραμένει UTF-8.

Η παραπάνω διαδικασία εξασφαλίζει ότι οι εγγραφές στα αρχεία CSV δεν περιέχουν περιττά κενά που ενδεχομένως θα μπορούσαν να επηρεάσουν αρνητικά περαιτέρω επεξεργασία ή ανάλυση των δεδομένων.

Κανονικοποίηση στηλών - Αφαίρεση μη απαραίτητων πληροφοριών

Η μέθοδος normalize_columns() είναι υπεύθυνη για την απομάκρυνση συγκεκριμένων στηλών από κάθε αρχείο .csv που εντοπίζεται στον φάκελο που της δίνεται ως είσοδος. Η διαδικασία αυτή πραγματοποιείται με σκοπό την απλοποίηση του συνόλου δεδομένων και την εξάλειψη πληροφοριών που δεν είναι χρήσιμες για την περαιτέρω ανάλυση.
Αναλυτικότερα, για κάθε αρχείο:

1. Διαβάζεται το αρχείο CSV με την pandas.read_csv(), ώστε να μετατραπεί σε DataFrame.
2. Πραγματοποιούνται τρεις διαδοχικές προσπάθειες (try-except) για να διαγραφούν οι στήλες:
   - **Σχολική μονάδα τοποθέτησης**: Περιλαμβάνει πληροφορία που αναμένεται να έχει ήδη ενοποιηθεί ή θεωρείται πλεονάζουσα σε άλλα πεδία.
   - **Τύπος Πρόσληψης**: Πιθανώς κωδικοποιείται ή καλύπτεται σε άλλη στήλη, καθιστώντας την περιττή.
   - **Τρίτεκνος**: Αν και κοινωνικό κριτήριο, δεν είναι κρίσιμο για την παρούσα επεξεργασία.
3. Σε περίπτωση που κάποια από τις παραπάνω στήλες δεν υπάρχει στο αρχείο, καταγράφεται το σφάλμα στην κονσόλα με το όνομα του αρχείου, χωρίς να διακόπτεται η διαδικασία.

4. Το τροποποιημένο DataFrame αποθηκεύεται ξανά στο ίδιο αρχείο, διατηρώντας τη μορφή CSV και την κωδικοποίηση σε UTF-8.

Η ύπαρξη πολλαπλών try-except blocks επιτρέπει να εντοπιστεί το λάθος για κάθε διαφορετική εκτέλεση εντολής.

Αφαίρεση στήλης αύξοντα αριθμού (A/A)

Η μέθοδος delete_AA() καλείται με στόχο την αφαίρεση της στήλης "A/A" από κάθε αρχείο CSV που εντοπίζεται στον δοθέντα φάκελο. Για κάθε αρχείο που έχει κατάληξη ".csv", δημιουργείται η πλήρης διαδρομή του αρχείου και στη συνέχεια διαβάζεται σε μορφή DataFrame μέσω της βιβλιοθήκης pandas. Η στήλη "A/A" διαγράφεται μέσω της drop(). Η μέθοδος περικλείεται σε try-except block ώστε να καταγράφονται τυχόν σφάλματα (για παράδειγμα σε περιπτώσεις όπου η στήλη δεν υπάρχει). Μετά την τροποποίηση, το αρχείο αποθηκεύεται ξανά στην ίδια διαδρομή, διατηρώντας τη μορφοποίηση σε UTF-8, όπως φαίνεται στο απόσπασμα 4.9. Η στήλη "A/A" αποτελεί έναν απλό αύξοντα αριθμό, ο οποίος δεν προσφέρει ουσιαστική πληροφορία στη διαδικασία ανάλυσης, οπότε η αφαίρεσή της συμβάλλει στην ομοιογένεια του συνόλου των δεδομένων.

```
1 try:
2     df = df.drop('AA/', axis=1)
3 except Exception as e:
4     print(f"An error occurred during preprocessing on method
    delete_AA : {e},{filename}")
```

Απόσπασμα Κώδικα 4.9: Διαγραγή στήλης A/A

Αφαίρεση της στήλης A/A ροής

Η delete_AA_ROHS() αφαιρεί τη στήλη "A/A ΡΟΗΣ" από κάθε αρχείο CSV του φακέλου. Για κάθε αρχείο με κατάληξη ".csv", κατασκευάζεται η πλήρης διαδρομή, το αρχείο διαβάζεται σε DataFrame και επιχειρείται η διαγραφή της εν λόγω στήλης μέσω της drop(). Η διαδικασία περικλείεται σε try-except block ώστε να καταγράφεται τυχόν αποτυχία — για παράδειγμα, όταν η στήλη απουσιάζει. Το τροποποιημένο αρχείο αποθηκεύεται στην αρχική του διαδρομή με διατήρηση της UTF-8 κωδικοποίησης. Η "A/A ΡΟΗΣ" αποτελεί επίσης πληροφορία χωρίς ουσιαστικό περιεχόμενο για την ανάλυση, και η απομάκρυνσή της βοηθά στον καθαρισμό των δεδομένων.

Κανονικοποίηση του τύπου τοποθέτησης (τύπος)

Η normalize_type() στοχεύει στη δημιουργία και κανονικοποίηση της στήλης ΤΥΠΟΣ, η οποία αποτυπώνει το είδος της εκπαιδευτικής μονάδας. Για κάθε αρχείο CSV, πρώτα διασφαλίζεται η ύπαρξη της στήλης ΤΥΠΟΣ μέσω της βοηθητικής συνάρτησης check_if_column_exists(). Εάν υπάρχει η στήλη ΤΥΠΟΣ ΤΟΠΟΘΕΤΗΣΗΣ (βλέπε απόσπασμα 4.10), το περιεχόμενό της μεταφέρεται στη νέα στήλη ΤΥΠΟΣ και η αρχική στήλη διαγράφεται. Εάν η στήλη ΤΥΠΟΣ παραμένει κενή, ελέγχεται το όνομα του αρχείου για λέξεις-κλειδιά όπως γενικής, και εφόσον

εντοπιστούν, συμπληρώνεται αντίστοιχα η τιμή ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ. Στη συνέχεια, γίνεται αναγωγή τιμών: η ΓΕΝΙΚΗΣ ΠΑΙΔΕΙΑΣ αντικαθίσταται από ΓΕΝΙΚΗΣ, ενώ το ΜΟΥΣΙΚΟ ΣΧΟΛΕΙΟ μετατρέπεται σε ΜΟΥΣΙΚΟ. Όσες τιμές δεν αντιστοιχούν σε αυτές τις κατηγορίες ταξινομούνται ως ΕΙΔΙΚΗΣ. Παράλληλα, γίνεται διερεύνηση της στήλης ΠΕΡΙΟΧΗ ΤΟΠΟΘΕΤΗΣΗΣ για την ύπαρξη της φράσης Μουσικό Σχολείο, οπότε και η αντίστοιχη γραμμή χαρακτηρίζεται ως ΜΟΥΣΙΚΟ όπως στο σχήμα 4.11. Εάν για οποιονδήποτε λόγο η στήλη ΤΥΠΟΣ παραμένει άδεια, συμπληρώνεται με κενές τιμές. Η συνάρτηση διασφαλίζει έτσι μια ενιαία και συνεπή μορφή κατηγοριοποίησης τύπων σχολικών μονάδων.

```
df['ΤΥΠΟΣ'] = df['ΤΥΠΟΣ ΤΟΠΟΘΕΤΗΣΗΣ']
df = df.drop(['ΤΥΠΟΣ ΤΟΠΟΘΕΤΗΣΗΣ'], axis=1)
```

Απόσπασμα Κώδικα 4.10: Αντικατάσταση και ενοποίηση της στήλης

```
df['ΤΥΠΟΣ'] = df['ΤΥΠΟΣ'].replace(type_mapping)
df['ΤΥΠΟΣ'] = df['ΤΥΠΟΣ'].apply(lambda x: x if x in ['ΓΕΝΙΚΗΣ',
   'ΜΟΥΣΙΚΟ'] else 'ΕΙΔΙΚΗΣ')
```

Απόσπασμα Κώδικα 4.11: Κανονικοποίηση τιμών

Διαγραφή της στήλης μητρώνυμο

Η συνάρτηση delete_mitronimo() διατρέχει όλα τα αρχεία CSV στον φάκελο και επιχειρεί να αφαιρέσει τη στήλη με όνομα ΜΗΤΡΩΝΥΜΟ. Η λειτουργία αυτή περιλαμβάνεται σε try-except μπλοκ ώστε να αντιμετωπίζονται τυχόν σφάλματα που προκύπτουν σε περίπτωση που η στήλη δεν υπάρχει στο αρχείο. Μετά την ενδεχόμενη διαγραφή, το τροποποιημένο DataFrame αποθηκεύεται εκ νέου στο ίδιο αρχείο, δείτε απόσπασμα 4.12.

```
try:
    df = df.drop('ΜΗΤΡΩΝΥΜΟ', axis=1)
except Exception as e:
    print(f"An error occurred during preprocessing on method
   delete_mitronimo : {e},{filename}")
```

Απόσπασμα Κώδικα 4.12: Διαγραφή στήλης Μητρώνυμο

Διαγραφή της στήλης πίνακας

Η συνάρτηση delete_pinakas αφαιρεί τη στήλη ΠΙΝΑΚΑΣ από όλα τα αρχεία CSV του φακέλου που δίδεται ως είσοδος. Αρχικά, για κάθε αρχείο CSV κατασκευάζεται το πλήρες μονοπάτι του αρχείου και διαβάζεται το περιεχόμενό του σε ένα αντικείμενο DataFrame. Στη συνέχεια, επιχειρείται τη διαγραφή της στήλης ΠΙΝΑΚΑΣ. Εάν η στήλη δεν υπάρχει ή προκύψει οποιοδήποτε σφάλμα κατά τη διαδικασία, καταγράφεται κατάλληλα ένα μήνυμα σφάλματος που περιλαμβάνει το όνομα του αρχείου. Τέλος, το τροποποιημένο DataFrame αποθηκεύεται στο ίδιο αρχείο CSV, δείτε απόσπασμα 4.13.

```
try:
    df = df.drop('ΠΙΝΑΚΑΣ', axis=1)
```

```
except Exception as e:
    print(f"An error occurred during preprocessing on method delete_pinakas : {e},{filename}")
```

Απόσπασμα Κώδικα 4.13: Διαγραφή στήλης Πίνακας

Κανονικοποίηση επωνύμου με παύλες

Η συνάρτηση `check_and_fix_eponymo()` έχει ως στόχο να διορθώσει τη μορφή των επωνύμων σε όλα τα αρχεία CSV μέσα σε έναν φάκελο. Ο λόγος είναι ότι στο backend της εφαρμογής υπάρχει το endpoint «/onoma/:id» (/onoma/ΒΟΥΓΙΟΥΚΑ_ΜΑΡΙΑ) το οποίο περιμένει το επώνυμο και το όνομα του ατόμου να είναι χωρισμένα με χαρακτήρα _ (κάτω παύλα), και το επώνυμο να αποτελείται από μία μόνο λέξη. Για να λειτουργεί σωστά η αναζήτηση, χρειάζεται τα επώνυμα που αποτελούνται από δύο λέξεις να ενώνονται με παύλα, δείτε απόσπασμα 4.14. Η συνάρτηση ελέγχει κάθε αρχείο που τελειώνει σε `.csv`, το ανοίγει και διαβάζει τα δεδομένα. Αν εντοπίσει τη στήλη ΕΠΩΝΥΜΟ (ή ενδεχομένως ΕΠΙΘΕΤΟ), τότε εξετάζει κάθε τιμή στη στήλη ως εξής:

- Αν η τιμή είναι μία λέξη, παραμένει ως έχει.
- Αν περιέχει δύο λέξεις, αυτές ενώνονται με παύλα (π.χ. «Παπαδόπουλος Παυλίδης» γίνεται «Παπαδόπουλος-Παυλίδης»).

```
if 'ΕΠΩΝΥΜΟ' or 'ΕΠΙΘΕΤΟ' in df.columns:

    df['ΕΠΩΝΥΜΟ'] = df['ΕΠΩΝΥΜΟ'].astype(str).apply(
        lambda x: x if x.count(' ') == 0 else x.replace(' ', '-', 1) if x.count(' ') == 2 else x
    )
else:
    print(f"Column ΕΠΩΝΥΜΟ'' not found in file {filename}")
```

Απόσπασμα Κώδικα 4.14: Κανονικοποίηση επώνυμου

Κανονικοποίηση της στήλης κλάδος

Η συνάρτηση normalize_klados υλοποιεί την κανονικοποίηση της στήλης ΚΛΑΔΟΣ για όλα τα αρχεία CSV που βρίσκονται στον καθορισμένο φάκελο. Για κάθε αρχείο, το οποίο είναι σε μορφή CSV, πρώτα διαβάζεται το περιεχόμενο σε ένα DataFrame. Στη συνέχεια, ελέγχεται αν η στήλη ΚΛΑΔΟΣ υπάρχει και εάν δεν υπάρχει, πραγματοποιείται η μεταφορά των δεδομένων από άλλες στήλες που σχετίζονται με τον κλάδο, όπως οι ΚΛΑΔΟΣ/ ΕΙΔΙΚΟΤΗΤΑ, ΚΛΑΔΟΣ / ΕΙΔΙΚΟΤΗΤΑ και ΕΙΔΙΚΟΤΗΤΑ. Σε περίπτωση που η στήλη ΚΛΑΔΟΣ δεν είναι παρούσα και οι άλλες στήλες περιέχουν δεδομένα που πρέπει να μεταφερθούν, η συνάρτηση αναλαμβάνει να αντιγράψει τα δεδομένα από τη στήλη που είναι διαθέσιμη στην αντίστοιχη στήλη ΚΛΑΔΟΣ, και να αφαιρέσει τις στήλες που περιέχουν τα δεδομένα αυτά. Εάν παρουσιαστεί σφάλμα κατά τη διάρκεια της διαδικασίας, καταγράφεται μήνυμα σφάλματος με το όνομα του αρχείου που προκαλεί το πρόβλημα, αναφορά στο απόσπασμα 4.15.

```
if 'ΚΛΑΔΟΣ/ ΕΙΔΙΚΟΤΗΤΑ' in df.columns:
                    df['ΚΛΑΔΟΣ'] = df['ΚΛΑΔΟΣ/ ΕΙΔΙΚΟΤΗΤΑ']
                    df = df.drop(['ΚΛΑΔΟΣ/ ΕΙΔΙΚΟΤΗΤΑ'], axis=1)
                elif 'ΚΛΑΔΟΣ / ΕΙΔΙΚΟΤΗΤΑ' in df.columns:
                    df['ΚΛΑΔΟΣ'] = df['ΚΛΑΔΟΣ / ΕΙΔΙΚΟΤΗΤΑ']
                    df = df.drop(['ΚΛΑΔΟΣ / ΕΙΔΙΚΟΤΗΤΑ'], axis=1)
```

Απόσπασμα Κώδικα 4.15: Κανονικοποίηση στήλης κλάδος

Κανονικοποίηση των τιμών της στήλης κλάδος

Η συνάρτηση normalize_klados_values εκτελεί την κανονικοποίηση των τιμών της στήλης ΚΛΑΔΟΣ για όλα τα αρχεία CSV που βρίσκονται στον καθορισμένο φάκελο. Κάθε αρχείο διαβάζεται σε ένα DataFrame και στη συνέχεια πραγματοποιούνται αντικαταστάσεις στις τιμές της στήλης ΚΛΑΔΟΣ σύμφωνα με έναν καθορισμένο πίνακα αντικαταστάσεων. Οι αντικαταστάσεις αυτές αναφέρονται σε κωδικούς όπως ΠΕ01.00, ΠΕ02.00 κ.ά., οι οποίοι μετατρέπονται στις αντίστοιχες τιμές όπως ΠΕ01, ΠΕ02 κ.λπ. Η συνάρτηση χρησιμοποιεί το str.replace για να αντικαταστήσει τα παλιά strings με τα νέα. Εάν προκύψει κάποιο σφάλμα κατά τη διαδικασία, καταγράφεται το σφάλμα με το όνομα του αρχείου που προκαλεί το πρόβλημα. Για καλύτερη κατανόηση δείτε απόσπασμα 4.16.

```
replacements = {
                'ΠΕ01.00': 'ΠΕ01',
                'ΠΕ02.00': 'ΠΕ02',
                'ΠΕ03.00': 'ΠΕ03',
                'ΠΕ05.00': 'ΠΕ05',
                'ΠΕ06.00': 'ΠΕ06',
                'ΠΕ07.00': 'ΠΕ07',
                ....
                'ΤΕ01.10': 'ΤΕ02.04',
                'ΤΕ01.11': 'ΤΕ02.04',
                'ΤΕ01.12': 'ΤΕ02.05',
                'ΤΕ01.14': 'ΤΕ02.05',
                'ΤΕ01.17': 'ΤΕ02.05',
                'ΤΕ01.22': 'ΤΕ02.06',
                'ΤΕ01.27': 'ΤΕ02.06',
                'ΤΕ01.32': 'ΤΕ02.07',
                'ΤΕ01.33': 'ΤΕ02.07',
                ....
                }
try:
    for old_string, new_string in replacements.items():
        df['ΚΛΑΔΟΣ'] = df['ΚΛΑΔΟΣ'].str.replace(old_string,
   new_string, regex=True)
except Exception as e:
    print(
```

```
25          f"An error occurred during preprocessing on method
    normalize_klados_values: {e},{filename}")
```

Απόσπασμα Κώδικα 4.16: Κανονικοποίηση τιμών στήλης κλάδος

Ανάλυση της μεθόδου dieth_ekps

Η μέθοδος dieth_ekps αναλαμβάνει την ανάγνωση και επεξεργασία δεδομένων από αρχείο CSV. Στην αρχή, δέχεται ως είσοδο την τοποθεσία του αρχείου μέσω της παραμέτρου path_file. Μετά, το περιεχόμενο του αρχείου διαβάζεται και αποθηκεύεται σε ένα DataFrame.
Όταν τα δεδομένα φορτωθούν, η μέθοδος εφαρμόζει κάποιες απλές διαδικασίες για να αφαιρέσει γραμμές που περιέχουν περιττά ή αχρείαστα δεδομένα. Αυτά τα δεδομένα συνήθως περιλαμβάνουν:

- **Λέξεις-κλειδιά:** Υπάρχουν κάποιες λέξεις ή φράσεις, όπως Τελικό Αποτέλεσμα, που εμφανίζονται σε κάποιες γραμμές και δεν είναι χρήσιμες για την υπόλοιπη διαδικασία. Όλες οι γραμμές που περιέχουν αυτές τις λέξεις απορρίπτονται.
- **Αριθμητικά δεδομένα:** Απορρίπτονται επίσης οι γραμμές που περιέχουν αριθμούς που δεν αφορούν την ανάλυση ή δεν είναι χρήσιμοι για την επόμενη επεξεργασία.

Μετά την εφαρμογή αυτών των φίλτρων, η μέθοδος κρατά μόνο τα δεδομένα που είναι χρήσιμα και τα εξάγει με τρόπο που διευκολύνει την αναγνώριση και επαναχρησιμοποίησή τους. Αν κάτι πάει στραβά κατά τη διάρκεια της διαδικασίας, καταγράφεται το σφάλμα, έτσι ώστε να εντοπιστεί το πρόβλημα και να λυθεί.
Στο απόσπασμα 4.17 υπάρχει ένα κομμάτι από την περίπλοκη μέθοδο. Σε πρώτη φάση κανονικοποιεί το όνομα της στήλης και έπειτα διαχειρίζεται ώστε τα ΠΕ/ΔΕ και ΄ (τόνος) να βρίσκονται στην σωστή θέση, αυτό είναι σημαντικό διότι χρειάζεται να ταιριάζει με το πρότυπο που έχει τεθεί στην βάση δεδομένων.

```
1
2  df['ΔΝΣΗ/ ΕΚΠΣΗΣ/'] = df['ΔΝΣΗ/ ΑΘΜΙΑΣ/ ΕΚΠΣΗΣ/']
3  df = df.drop(['ΔΝΣΗ/ ΑΘΜΙΑΣ/ ΕΚΠΣΗΣ/'], axis=1)
4
5  df['ΔΝΣΗ/ ΕΚΠΣΗΣ/'] = df['ΔΝΣΗ/ ΠΘΜΙΑΣ/ ΕΚΠΣΗΣ/']
6  df = df.drop(['ΔΝΣΗ/ ΠΘΜΙΑΣ/ ΕΚΠΣΗΣ/'], axis=1)
7
8  df['ΔΝΣΗ/ ΕΚΠΣΗΣ/'] = df['ΔΝΣΗ/ ΠΡΩΤΟΒΑΘΜΙΑΣ ΕΚΠΣΗΣ/']
9  df = df.drop(['ΔΝΣΗ/ ΠΡΩΤΟΒΑΘΜΙΑΣ ΕΚΠΣΗΣ/'], axis=1)
10
11
12 df['Suffix'] = ''
13 for index, row in df.iterrows():
14     suffix_position = str(df.at[index, 'ΔΝΣΗ/
     ΕΚΠΣΗΣ/']).find('΄')
15     if suffix_position != -1 and suffix_position > 0:
```

```
16          df.at[index, 'Suffix'] = str(df.at[index, 'ΔΝΣΗ/
    ΕΚΠΣΗΣ/'])[suffix_position - 1]
17          df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/'] = (
18              str(df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/'])[:suffix_position
    - 1] +
19              str(df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/'])[suffix_position:]
20          ).strip()
21
22
23 df['Prefix'] = ''
24 for index, row in df.iterrows():
25     for key_phrase in key_phrases_PE_DE:
26         if key_phrase in str(df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/']):
27             matched_phrase = key_phrase
28             df.at[index, 'Prefix'] = matched_phrase
29             df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/'] = str(df.at[index,
    'ΔΝΣΗ/ ΕΚΠΣΗΣ/']).replace(matched_phrase, '').strip()
30
31
32 for index, row in df.iterrows():
33     ...
34     if 'ΔΕ ' not in value_str and 'ΠΕ ' not in value_str:
35         df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/'] = f"ΔΕ{'' if result ==
    ΔΕ'' else ΠΕ('' if result == ΠΕ'' else ΠΕ('' if
    check_klados(df) else '-'))}"
36         if df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/'] != '-':
37             df.at[index, 'ΔΝΣΗ/ ΕΚΠΣΗΣ/'] += f" {value_str}"
```

Απόσπασμα Κώδικα 4.17: Κανονικοποίηση στήλης Δ/ΝΣΗ ΕΚΠ/ΣΗΣ

Ανάλυση της μεθόδου check_moria_pinaka

Η μέθοδος check_moria_pinaka υλοποιεί μια διαδικασία που ελέγχει και τροποποιεί αρχεία CSV σε έναν καθορισμένο φάκελο. Αρχικά, η μέθοδος διαβάζει όλα τα αρχεία του φακέλου και επιλέγει τα αρχεία που έχουν κατάληξη .csv. Για κάθε αρχείο, διαβάζεται το περιεχόμενο σε ένα DataFrame χρησιμοποιώντας τη βιβλιοθήκη pandas. Στη συνέχεια, η μέθοδος ελέγχει αν η στήλη ΜΟΡΙΑ ΠΙΝΑΚΑ υπάρχει ήδη. Αν δεν υπάρχει, καλείται η συνάρτηση check_if_column_exists για να την προσθέσει. Εάν παρουσιαστεί κάποιο σφάλμα κατά την εκτέλεση αυτής της διαδικασίας, καταγράφεται το μήνυμα σφάλματος μαζί με το όνομα του αρχείου που προκάλεσε το πρόβλημα. Τέλος, το τροποποιημένο DataFrame αποθηκεύεται πίσω στο αρχείο CSV με την ίδια ονομασία, διασφαλίζοντας ότι οι αλλαγές αποθηκεύονται με τον ίδιο τρόπο και χωρίς να προστίθεται αριθμός στην ονομασία του αρχείου.

Ανάλυση της μεθόδου delete_periferia

Η μέθοδος delete_periferia υλοποιεί μια διαδικασία για την αφαίρεση της στήλης ΠΕΡΙΦΕΡΕΙΑ από όλα τα αρχεία CSV σε έναν καθορισμένο φάκελο. Η μέθοδος ξεκινά διαβάζοντας όλα τα αρχεία στον φάκελο και επιλέγοντας εκείνα που έχουν κατάληξη .csv. Για κάθε αρχείο, το περιεχόμενο διαβάζεται σε ένα DataFrame χρησιμοποιώντας τη βιβλιοθήκη pandas. Στη συνέχεια, επιχειρείται η αφαίρεση της στήλης ΠΕΡΙΦΕΡΕΙΑ. Εάν η στήλη δεν υπάρχει ή παρουσιαστεί οποιοδήποτε σφάλμα κατά τη διάρκεια της διαδικασίας, καταγράφεται το μήνυμα σφάλματος μαζί με το όνομα του αρχείου που προκάλεσε το πρόβλημα. Στο τέλος, το τροποποιημένο DataFrame αποθηκεύεται πίσω στο αρχείο CSV, διατηρώντας την ίδια ονομασία και εξασφαλίζοντας ότι οι αλλαγές εφαρμόζονται σωστά.

Προσθήκη της στήλης ημερομηνία από το όνομα του αρχείου

Η μέθοδος add_hmeromhnia προσθέτει μια νέα στήλη ΗΜΕΡΟΜΗΝΙΑ σε όλα τα αρχεία CSV που βρίσκονται σε έναν καθορισμένο φάκελο. Η διαδικασία ξεκινά με την ανάγνωση όλων των αρχείων CSV στο φάκελο και την εξαγωγή της ημερομηνίας(εάν αυτή υπάρχει) από το όνομα του αρχείου. Η ημερομηνία αναζητείται χρησιμοποιώντας κανονική έκφραση (regex) για τον εντοπισμό της μορφής YYYY-MM-DD. Εάν η ημερομηνία βρεθεί στο όνομα του αρχείου, η μέθοδος διαβάζει το περιεχόμενο του αρχείου σε ένα DataFrame και προσθέτει μια νέα στήλη ΗΜΕΡΟΜΗΝΙΑ με την ημερομηνία που εξήχθη από το όνομα του αρχείου. Εάν προκύψει κάποιο σφάλμα κατά την εκτέλεση της διαδικασίας, καταγράφεται το μήνυμα του σφάλματος, μαζί με το όνομα του αρχείου στο οποίο συνέβη το πρόβλημα. Στο τέλος, το τροποποιημένο DataFrame αποθηκεύεται πίσω στο αρχείο CSV, διατηρώντας την ίδια ονομασία και εξασφαλίζοντας ότι η νέα στήλη προστέθηκε σωστά. Στο απόσπασμα κώδικα 4.18 φαίνεται το ουσιώδης σημείο της μεθόδου.

```
1 match = re.search(r'\d{4}-\d{2}-\d{2}', filename)
2 date_from_filename = match.group() if match else None
3
4 if date_from_filename:
5     input_path = os.path.join(folder_path, filename)
6     df = pd.read_csv(input_path, encoding='utf-8')
7     df['ΗΜΕΡΟΜΗΝΙΑ'] = date_from_filename
```

Απόσπασμα Κώδικα 4.18: Κανονικοποίηση στήλη ημερομηνία

Προσθήκη της στήλης ωράριο με την τιμή ΑΠΩ

Η μέθοδος add_orario_values προσθέτει τη στήλη ΩΡΑΡΙΟ σε όλα τα αρχεία CSV ενός καθορισμένου φακέλου που περιέχουν την λέξη ΑΠΩ στο όνομα του αρχείου. Η διαδικασία ξεκινά με την ανάγνωση όλων των αρχείων που πληρούν την προϋπόθεση αυτή και την επεξεργασία τους. Για κάθε αρχείο, ελέγχεται αν η στήλη ΩΡΑΡΙΟ υπάρχει ήδη. Εάν δεν υπάρχει, η μέθοδος προσθέτει τη στήλη ΩΡΑΡΙΟ και αναθέτει σε όλα τα στοιχεία αυτής την τιμή ΑΠΩ. Στο τέλος, το τροποποιημένο DataFrame αποθηκεύεται πίσω στο αρχείο CSV, διατηρώντας την ίδια ονομασία και εξασφαλίζοντας ότι η στήλη ΩΡΑΡΙΟ έχει προστεθεί σωστά σε όλα τα δεδομένα.

Δημιουργία της στήλης έτος

Η μέθοδος create_sxoliko_etos δημιουργεί τη στήλη ΕΤΟΣ για αρχεία CSV. Η διαδικασία ξεκινά με την αναζήτηση και εξαγωγή της ημερομηνίας και του έτους από το όνομα του αρχείου. Ειδικότερα, η μέθοδος χρησιμοποιεί εκφράσεις κανονικών εκφράσεων (regex) για να εντοπίσει τις ημερομηνίες και τα έτη που περιλαμβάνονται στο όνομα του αρχείου και να τα αποθηκεύσει σε μεταβλητές. Στη συνέχεια, αναγνωρίζει το σχολικό έτος με βάση τον μήνα της ημερομηνίας. Εάν ο μήνας είναι από Σεπτέμβριο έως Δεκέμβριο, το σχολικό έτος ορίζεται ως το τρέχον έτος και το επόμενο, ενώ αν ο μήνας είναι από Ιανουάριο έως Αύγουστο, το σχολικό έτος ορίζεται ως το προηγούμενο έτος και το τρέχον. Τέλος, η μέθοδος προσθέτει τη στήλη ΕΤΟΣ στο DataFrame και αποθηκεύει το τροποποιημένο αρχείο πίσω στον φάκελο, διατηρώντας την ίδια ονομασία για το αρχείο CSV. Το απόσπασμα 4.19 παρέχει τον έλεγχο καθορισμού σχολικού έτους.

```
1 if 9 <= date_object.month <= 12:
2     df['ΕΤΟΣ'] = f"{year_from_filename}-{int(year_from_filename)
    + 1}"
3 elif 1 <= date_object.month <= 8:
4     df['ΕΤΟΣ'] = f"{int(year_from_filename) -
    1}-{year_from_filename}"
```

Απόσπασμα Κώδικα 4.19: Κανονικοποίηση στήλης έτος

Προσθήκη μουσικών οργάνων στα σχόλια

Η μέθοδος add_mousika_organa_to_sxolia προσθέτει τις πληροφορίες για τα μουσικά όργανα στα σχόλια των δεδομένων, για όλα τα αρχεία CSV που βρίσκονται σε έναν καθορισμένο φάκελο. Αρχικά, η μέθοδος διαβάζει τα δεδομένα του αρχείου και στη συνέχεια ενσωματώνει τις τιμές της στήλης ΜΟΥΣΙΚΗ ΕΙΔΙΚΕΥΣΗ στη στήλη ΣΧΟΛΙΑ, προσθέτοντας την πληροφορία για την μουσική ειδικότητα στον ήδη υπάρχοντα κείμενο της στήλης ΣΧΟΛΙΑ. Στη συνέχεια, η μέθοδος αφαιρεί τις στήλες ΜΟΥΣΙΚΗ ΕΙΔΙΚΕΥΣΗ και ΩΡΑΡΙΟ, αφού τα δεδομένα τους πλέον περιλαμβάνονται στην στήλη ΣΧΟΛΙΑ. Τέλος, τα δεδομένα αποθηκεύονται πίσω στο ίδιο αρχείο CSV με τις τροποποιήσεις που έχουν γίνει. Εάν παρουσιαστεί κάποιο σφάλμα κατά τη διάρκεια της διαδικασίας, το μήνυμα σφάλματος καταγράφεται, επισημαίνοντας το αρχείο που προκαλεί το πρόβλημα. Ο κώδικας είναι διαθέσιμος στο απόσπασμα 4.20

```
1 df['ΣΧΟΛΙΑ'] = df['ΣΧΟΛΙΑ'] + ', ' + df['ΜΟΥΣΙΚΗ ΕΙΔΙΚΕΥΣΗ']
2                 df.drop(columns=['ΜΟΥΣΙΚΗ ΕΙΔΙΚΕΥΣΗ'],
    inplace=True)
3                 df.drop(columns=['ΩΡΑΡΙΟ'], inplace=True)
```

Απόσπασμα Κώδικα 4.20: Κανονικοποίηση μουσικών οργάνων

Αφαίρεση κενών χαρακτήρων πριν και μετά από κόμματα

Η μέθοδος remove_empty_spaces_before_after_commas καθαρίζει τα δεδομένα από κενά που μπορεί να υπάρχουν πριν ή μετά από κόμματα σε όλα τα αρχεία CSV που βρίσκονται σε έναν καθορισμένο φάκελο. Αρχικά, η μέθοδος διαβάζει το περιεχόμενο του κάθε αρχείου και στη

συνέχεια εφαρμόζει μια συνάρτηση που αφαιρεί τα κενά από τα δεδομένα σε κάθε στήλη του αρχείου, εφόσον αυτά τα δεδομένα είναι συμβολοσειρές. Όλες οι τιμές που δεν είναι συμβολοσειρές παραμένουν αμετάβλητες. Εάν παρουσιαστεί κάποιο σφάλμα κατά τη διάρκεια της διαδικασίας, καταγράφεται το μήνυμα σφάλματος, επισημαίνοντας το αρχείο που προκαλεί το πρόβλημα. Στο τέλος, το επεξεργασμένο αρχείο αποθηκεύεται πίσω στο ίδιο αρχείο CSV.

Επαναταξινόμηση στηλών

Η μέθοδος re_order αναλαμβάνει την επαναταξινόμηση των στηλών ενός αρχείου CSV σύμφωνα με μια επιθυμητή σειρά στηλών. Αρχικά, η μέθοδος ορίζει τη σειρά των στηλών που πρέπει να υπάρχουν στο αρχείο και ελέγχει εάν κάποιες στήλες λείπουν ή είναι επιπλέον σε κάθε αρχείο. Στη συνέχεια, προστίθενται οι απουσιάζουσες στήλες με κενές τιμές και αφαιρούνται οι επιπλέον στήλες που δεν ανήκουν στη ζητούμενη σειρά. Μετά από αυτές τις τροποποιήσεις, τα δεδομένα επαναταξινομούνται σύμφωνα με τη νέα σειρά και το αρχείο αποθηκεύεται με τις αλλαγές. Εάν παρουσιαστεί κάποιο σφάλμα κατά τη διάρκεια της διαδικασίας, καταγράφεται το μήνυμα σφάλματος. Ο κώδικας υπάρχει στο απόσπασμα 4.21

```
1  desired_order = ['ΤΥΠΟΣ', 'ΕΠΩΝΥΜΟ', 'ΟΝΟΜΑ', 'ΠΑΤΡΩΝΥΜΟ',
       'ΚΛΑΔΟΣ', 'ΣΕΙΡΑ ΠΙΝΑΚΑ',
2                      'ΜΟΡΙΑ ΠΙΝΑΚΑ',
3                      'ΠΕΡΙΟΧΗ ΤΟΠΟΘΕΤΗΣΗΣ', 'ΔΝΣΗ/ ΕΚΠΣΗΣ/',
       'ΗΜΕΡΟΜΗΝΙΑ', 'ΕΤΟΣ', 'ΣΧΟΛΙΑ']
4  ....
5  missing_columns = set(desired_order) - set(df.columns)
6                  for missing_column in missing_columns:
7                      df[missing_column] = None
8
9                  extra_columns = set(df.columns) -
       set(desired_order)
10                 df = df.drop(columns=extra_columns,
       errors='ignore')
11
12                 df = df[desired_order]
13
14 ....
```

Απόσπασμα Κώδικα 4.21: Επαναταξινόμηση στηλών

Κανονικοποίηση ονομάτων στηλών

Η μέθοδος normalize_all_columns_names αναλαμβάνει την κανονικοποίηση των ονομάτων των στηλών ενός αρχείου CSV. Αρχικά, η μέθοδος δημιουργεί έναν χάρτη (mapping) που αντιστοιχεί τα παλιά ονόματα των στηλών στα νέα, σύμφωνα με μια προκαθορισμένη αναφορά. Στη συνέχεια, διαβάζει το περιεχόμενο του αρχείου και αναδιοργανώνει τις στήλες σύμφωνα με τον χάρτη. Οι στήλες που δεν αντιστοιχούν στον χάρτη αφαιρούνται. Εάν παρουσιαστεί κάποιο σφάλμα κατά τη διαδικασία, καταγράφεται το μήνυμα του σφάλματος. Μετά την κανονι-

κοποίηση των ονομάτων των στηλών, το αρχείο αποθηκεύεται με τις αλλαγές(Στο απόσπασμα 4.22 παρέχεται η χαρτογράφηση των ονομάτων).

```
column_mapping = {
                'ΤΥΠΟΣ': 'Typos',
                'ΕΠΩΝΥΜΟ': 'Eponymo',
                'ΟΝΟΜΑ': 'Onoma',
                'ΠΑΤΡΩΝΥΜΟ': 'Patronymo',
                'ΚΛΑΔΟΣ': 'Klados',
                'ΣΕΙΡΑ ΠΙΝΑΚΑ': 'Seira_Pinaka',
                'ΜΟΡΙΑ ΠΙΝΑΚΑ': 'Moria_Pinaka',
                'ΠΕΡΙΟΧΗ ΤΟΠΟΘΕΤΗΣΗΣ': 'Perioxh_Topothethshs',
                'ΔΝΣΗ/ ΕΚΠΣΗΣ/': 'Dieytynsh_Ekpaideyshs',
                'ΗΜΕΡΟΜΗΝΙΑ': 'Hmeromnia',
                'ΕΤΟΣ': 'Etos',
                'ΣΧΟΛΙΑ': 'Sxolia',
            }
...
```

Απόσπασμα Κώδικα 4.22: Ονόματα στηλών

Αφαίρεση γραμμών με άδεια ονόματα

Η μέθοδος remove_rows_with_empty_names υλοποιεί την αφαίρεση των γραμμών από τα αρχεία CSV που περιέχουν κενά πεδία στα ονόματα. Συγκεκριμένα, για κάθε αρχείο CSV, η μέθοδος ελέγχει αν οι στήλες Eponymo, Onoma και Patronymo είναι κενές. Εάν όλες αυτές οι στήλες περιέχουν κενές τιμές (null), η γραμμή διαγράφεται. Το επεξεργασμένο αρχείο αποθηκεύεται ξανά, διατηρώντας τις αλλαγές. Οι τριάδα Επώνυμο, Όνομα και Πατρώνυμο εγγυώνται πως το πρόσωπο είναι μοναδικό μιας και είναι αρκετά σπάνιο να βρεθεί 2 ή παραπάνω άτομα με αυτά τα τρία πεδία ίδια. Η γραμμή ελέγχου και διαγραφής μπορούν να παρατηρηθούν στο παράρτημα 4.23

```
condition = df['Eponymo'].isnull() & df['Onoma'].isnull() &
   df['Patronymo'].isnull()
df = df[~condition]
```

Απόσπασμα Κώδικα 4.23: Έλεγχος γραμμών

Συγχώνευση αρχείων CSV με full outer join

Τέλος, η μέθοδος full_outer_join_csv_files υλοποιεί τη συγχώνευση πολλών αρχείων CSV που βρίσκονται στον καθορισμένο φάκελο, χρησιμοποιώντας τη μέθοδο full outer join. Κάθε αρχείο CSV διαβάζεται και επεξεργάζεται πριν από τη συγχώνευση. Στη διαδικασία αυτή, οι στήλες με τύπο δεδομένων object γεμίζουν με κενές τιμές, ενώ οι στήλες Moria_Pinaka και Seira_Pinaka μετατρέπονται σε αριθμητικές τιμές και γεμίζουν με NaN όπου είναι απαραίτητο. Επιπλέον, οι στήλες Perioxh_Topothethshs και Klados μετατρέπονται σε τύπο object. Στη συνέχεια, όλα τα αρχεία συγχωνεύονται με τη μέθοδο outer, διατηρώντας όλα τα δεδομένα από κάθε αρχείο,

ακόμα και εκείνα που δεν υπάρχουν σε κάθε αρχείο. Εάν παρουσιαστούν διπλότυπες γραμμές, αυτές αφαιρούνται πριν από την αποθήκευση του τελικού αρχείου. Το τελικό αρχείο αποθηκεύεται με ένα μοναδικό όνομα, το οποίο περιλαμβάνει την τρέχουσα ημερομηνία και έναν τυχαίο αριθμό. Εάν παρουσιαστεί κάποιο σφάλμα κατά τη διάρκεια της διαδικασίας, καταγράφεται το μήνυμα του σφάλματος.

Αφαίρεση ελλιπών διπλοτύπων γραμμών

Η μέθοδος remove_inferior_duplicates εκτελεί μια διαδικασία καθαρισμού δεδομένων σε αρχεία CSV, με σκοπό την αφαίρεση των "κατώτερων" ή ελλιπών διπλοτύπων γραμμών, διατηρώντας την καλύτερη εγγραφή από κάθε ομάδα διπλοτύπων. Η διαδικασία ξεκινά διαβάζοντας το αρχείο CSV σε ένα DataFrame, χρησιμοποιώντας κωδικοποίηση utf-8 και τύπο δεδομένων str, και γεμίζει κάθε κενό πεδίο με κενή συμβολοσειρά. Η βοηθητική συνάρτηση is_missing χρησιμοποιείται για να εντοπίσει τις "ελλιπείς" τιμές, οι οποίες θεωρούνται οι τιμές που είναι κενές ή έχουν τιμές όπως '-', 'null', 'nan' ή 'κενό'. Στη συνέχεια, ορίζεται μια λίστα comparison_columns που περιλαμβάνει τις στήλες που θα χρησιμοποιηθούν για τη σύγκριση των γραμμών, όπως τα πεδία Klados, Seira_Pinaka, Moria_Pinaka, Perioxh_Topothethshs, Dieytynsh_Ekpaideyshs και Sxolia. Ελέγχεται αν οι στήλες που καθορίζονται στη λίστα comparison_columns υπάρχουν στο DataFrame και, αν κάποια από αυτές απουσιάζει, προστίθεται με κενές τιμές. Το DataFrame ομαδοποιείται με βάση τις στήλες που αποτελούν ταυτότητες (identity_columns), οι οποίες περιλαμβάνουν τα πεδία Typos, Eponymo, Onoma, Patronymo, Hmeromnia και Etos. Κάθε ομάδα εξετάζεται για να διαπιστωθεί αν περιέχει μόνο μία γραμμή ή περισσότερες, και αν περιέχει μόνο μία, αυτή προστίθεται στο τελικό σύνολο δεδομένων. Για τις ομάδες με περισσότερες από μία γραμμές, υπολογίζεται μια "βαθμολογία" για κάθε γραμμή, η οποία βασίζεται στον αριθμό των μη κενών τιμών στις στήλες σύγκρισης. Οι γραμμές ταξινομούνται με βάση αυτή τη βαθμολογία σε φθίνουσα σειρά, και η καλύτερη εγγραφή διατηρείται, ενώ οι "κατώτερες" γραμμές αφαιρούνται και αποθηκεύονται για μελλοντική αναφορά. Όλες οι στήλες του DataFrame επαναταξινομούνται για να αντιστοιχούν σε μια καθορισμένη σειρά στηλών (ordered_columns), και αν κάποιες στήλες δεν υπάρχουν, προστίθεται μια στήλη με κενές τιμές. Τέλος, το καθαρισμένο DataFrame αποθηκεύεται σε ένα νέο αρχείο CSV, με ένα όνομα που περιλαμβάνει την τρέχουσα ημερομηνία και έναν τυχαίο αριθμό για να διασφαλιστεί η μοναδικότητα του αρχείου, ενώ τα δεδομένα αποθηκεύονται με κωδικοποίηση utf-8 και τα κενά πεδία αναπαρίστανται ως NULL.

Στον πυρήνα της μεθόδου, οι εγγραφές ομαδοποιούνται με βάση πεδία ταυτότητας όπως τύπος, ονοματεπώνυμο και ημερομηνία (identity_columns), ώστε να εντοπιστούν διπλότυπα. Σε κάθε ομάδα, υπολογίζεται μια βαθμολογία πληρότητας (__score) με βάση τον αριθμό μη κενών τιμών σε επιλεγμένες στήλες σύγκρισης, και διατηρείται η πληρέστερη εγγραφή. Αν λείπουν στήλες από το αρχείο, προστίθενται αυτόματα με κενές τιμές για να διατηρηθεί η συνοχή. Τέλος, το καθαρισμένο αποτέλεσμα αποθηκεύεται σε νέο αρχείο με μοναδικό όνομα, που περιλαμβάνει την ημερομηνία και έναν τυχαίο αριθμό (Δείτε απόσπασμα 4.24).

```
identity_columns = [
    "Typos", "Eponymo", "Onoma", "Patronymo",
    "Hmeromnia", "Etos"
]
```

```
 5
 6 grouped = df.groupby(identity_columns, sort=False)
 7 ....
 8 group['__score'] = group[comparison_columns].apply(
 9     lambda row: sum(not is_missing(val) for val in row), axis=1
10 )
11 group_sorted = group.sort_values('__score', ascending=False)
12 best_row = group_sorted.iloc[0]
13 ....
14 for col in comparison_columns:
15     if col not in df.columns:
16         df[col] = ''
17 ...
18 current_date = date.today().strftime("%Y-%m-%d")
19 random_number = random.randint(10000, 99999)
20 cleaned_filename =
     f"cleaned_output-{current_date}-{random_number}.csv"
21 cleaned_df.to_csv(cleaned_path, index=False, encoding='utf-8',
     na_rep='NULL')
22 ...
23 removed_rows.append(row.drop('__score'))
24 ...
```

Απόσπασμα Κώδικα 4.24: Αφαίρεση ελλιπών διπλοτύπων γραμμών

4.3.1 Github Repository

Ο open source κώδικας της εφαρμογής είναι ανεβασμένος σε ένα github repository το οποίο βρίσκεται στον σύνδεσμο `https://github.com/mikeApostolidis/ScrapAndPreProcess`.

Κεφάλαιο 5

Ανάπτυξη εφαρμογής android

Η εφαρμογή για κινητές συσκευές δημιουργήθηκε με ReactJs Expo έχοντας κρατήσει παρόμοιο UI/UX, έτσι ο χρήστης εξοικειώνοντας με την μια, θα μπορεί εύκολα να πλοηγηθεί και στην άλλη. Επίσης, η εφαρμογή κάνει χρήση των ήδη υπάρχων API που δημιουργήθηκαν στην αρχική πτυχιακή για να εμφανίσει τα αποτελέσματα.

Δομή εφαρμογής

Η εφαρμογή διαχειρίζεται την πλοήγηση μεταξύ των διαφόρων οθονών μέσω της βιβλιοθήκης «Stack Navigator». Η υλοποίηση της «Stack Navigator» γίνεται στο αρχείο AppNavigation, όπου καθορίζονται οι πέντε βασικές οθόνες της εφαρμογής: «Home Screen2», «Result Screen», «Error Screen», «Api Document» και «General Result Screen». Η ιδιότητα headerShown έχει τεθεί σε false για όλες τις οθόνες, καθώς το προεπιλεγμένο header που παρέχει η βιβλιοθήκη περιλαμβάνει επιπλέον στοιχεία, όπως αυτόματο τίτλο, κουμπί επιστροφής και άλλα εργαλεία, τα οποία δεν εντάσσονται στον σχεδιασμό της παρούσας εφαρμογής. Ο κώδικας παρουσιάζεται στο Απόσπασμα Κώδικα 5.1.

```
const AppNavigator = () => (
  <Stack.Navigator>
    <Stack.Screen name="Home Screen2" component={HomeScreen2}
      options={{ headerShown: false }}/>

    <Stack.Screen name="Result Screen" component={ResultScreen}
      options={{ headerShown: false }}/>

    <Stack.Screen name="Error Screen" component={ErrorScreen}
      options={{ headerShown: false }}/>

    <Stack.Screen name="Api Document" component={
   ApiDocumentScreen}
      options={{ headerShown: false }}/>

    <Stack.Screen name="General Result Screen" component={
   GeneralResultScreen}
      options={{ headerShown: false }}/>

  </Stack.Navigator>
);
```

Απόσπασμα Κώδικα 5.1: Ορισμός του AppNavigator

Αρχική οθόνη

Η HomeScreen2 αποτελεί το αρχικό σημείο αλληλεπίδρασης του χρήστη με την εφαρμογή και παρέχει δύο κύριες λειτουργικότητες αναζήτησης: Προσωπική και Γενική. Η εναλλαγή μεταξύ των δύο γίνεται μέσω ενός διακόπτη Switch, ο οποίος διαχειρίζεται την κατάσταση isEnabled. Η έτοιμη βιβλιοθήκη DropDownPicker χρησιμοποιήθηκε ως πεδίο εισαγωγής κειμένου (input text field) μιας και προσφέρει απαραίτητες λειτουργίες όπως: αναζήτηση, πολλαπλή επιλογή και drop down list.

Χρήση του DropDownPicker

Για την υλοποίηση των φίλτρων αναζήτησης στην αρχική οθόνη επιλέχθηκε η βιβλιοθήκη react-native-dropdown-picker, η οποία προσφέρει ένα ευέλικτο και προσαρμόσιμο dropdown μενού, ειδικά σχεδιασμένο για εφαρμογές React Native.
Κάθε ένα από τα φίλτρα υλοποιείται ως ξεχωριστό instance του DropDownPicker και διαχειρίζεται τη δική του κατάσταση open, value και items. Οι λίστες των δεδομένων (items) ανακτώνται από τις αντίστοιχες κλήσεις API και αποθηκεύονται στο αντίστοιχο state. Χαρακτηριστικό παράδειγμα αποτελεί το dropdown για την επιλογή περιοχής τοποθέτησης, το οποίο διαφέρει σημαντικά από τα υπόλοιπα:

- Υποστηρίζει πολλαπλή επιλογή (multiple = true) μέσω badges.

- Κάθε επιλεγμένη τιμή εμφανίζεται ως έγχρωμο badge, με δυνατότητα διαγραφής.
- Ενεργοποιείται η αναζήτηση στο dropdown (searchable = true), διευκολύνοντας την εύρεση περιοχών από μεγάλες λίστες.
- Ορίζονται έως και 10 επιλογές ταυτόχρονα μέσω της ιδιότητας max.

Ο σχετικός κώδικας για την περιοχή τοποθέτησης έχει ως εξής, όπως παρουσιάζεται στο Απόσπασμα Κώδικα 5.2:

```
<DropDownPicker
  open={openPeriox_Topothet}
  setOpen={(isOpen) => {
                handleOpenPeriox_Topothet(isOpen);
                setActiveDropdown(isOpen ? '
   Periox_Topothet' : '');
          }}
  items={periox_TopothetList}
  setItems={setPeriox_TopothetList}
  value={searchValuePeriox_Topothet}
  setValue={setSearchValuePeriox_Topothet}
  placeholder="Περιοχή Τοποθέτησης"
  containerStyle={styles.dropdownContainer}
  style={styles.dropdownStyle}
  dropDownStyle={styles.dropdownDropDownStyle}
  searchable={true}
  searchPlaceholder="Αναζήτηση..."
  closeOnBackPressed={true}
  multiple={true}
  zIndex={998}
  min={0}
  max={10}
  mode="BADGE"
  badgeDotColors={[ 'rgba(255, 99, 132, 1)',
                    'rgba(54, 162, 235, 1)',
                    'rgba(255, 206, 86, 1)',
                    'rgba(75, 192, 192, 1)',
                    'rgba(153, 102, 255, 1)',
                    'rgba(255, 159, 64, 1)',
```

```
                    'rgba(255, 0, 255, 1)',
                    'rgba(128, 0, 0, 1)',
                    'rgba(0, 128, 128, 1)',
                    'rgba(128, 0, 128, 1)']}
                />
```

Απόσπασμα Κώδικα 5.2: DropDownPicker με πολλαπλή επιλογή περιοχής τοποθέτησης

Η υλοποίηση περιλαμβάνει τα ακόλουθα βασικά σημεία:

- Την ιδιότητα mode=”BADGE” που καθορίζει τον τρόπο εμφάνισης πολλαπλών επιλογών.
- Τον ορισμό χρωματιστών κουκκίδων μέσω του badgeDotColors[] ώστε κάθε επιλογή να ξεχωρίζει οπτικά.
- Την κλήση setActiveDropdown που διαχειρίζεται ποιο dropdown είναι ανοιχτό κάθε στιγμή, αποτρέποντας επικάλυψη.

Η γενική υλοποίηση για ένα από τα υπόλοιπα όπως ο κλάδος γίνεται ως εξής:

- Το dropdown υποστηρίζει πολλαπλή επιλογή (multiple = true), η οποία εμφανίζεται ως badges.
- Ενεργοποιείται η αναζήτηση (searchable = true), επιτρέποντας στο χρήστη να βρει ευκολότερα την επιθυμητή επιλογή.
- Η διαχείριση της κατάστασης (open, value, items) γίνεται μέσω των React Hooks (useState).

Παρακάτω δίνεται χαρακτηριστικό παράδειγμα υλοποίησης του DropDownPicker για την επιλογή κλάδου, όπως χρησιμοποιείται στην οθόνη:

```
<DropDownPicker
  open={openKlados}
  setOpen={(isOpen) => {
    handleOpenKlados(isOpen);
    setActiveDropdown(isOpen ? 'Klados' : '');
  }}
  items={kladosList}
  setItems={setKladosList}
  value={searchValueKlados}
  setValue={setSearchValueKlados}
  placeholder="Κλάδος"
  containerStyle={styles.dropdownContainer}
  style={styles.dropdownStyle}
  dropDownStyle={styles.dropdownDropDownStyle}
  searchable={true}
  searchPlaceholder="Αναζήτηση..."
```

```
17   closeOnBackPressed={true}
18   zIndex={999}
19 />
```

Απόσπασμα Κώδικα 5.3: DropDownPicker με επιλογή κλάδου

Κάθε instance του DropDownPicker έχει την ακόλουθη βασική δομή στον κώδικα της εφαρμογής:

- **open**: Κατάσταση που δείχνει αν το dropdown είναι ανοιχτό.
- **setOpen**: Handler που ενεργοποιεί ή απενεργοποιεί το dropdown.
- **value**: Κατάσταση που διατηρεί τις επιλεγμένες τιμές.
- **setValue**: Handler που αποθηκεύει τις επιλογές του χρήστη.
- **items**: Κατάσταση που περιέχει τις δυναμικά φορτωμένες επιλογές από το API.

Η αρχικοποίηση του dropdown γίνεται με αντίστοιχο useEffect hook κατά την φόρτωση της οθόνης, το οποίο εκτελεί το αίτημα προς το αντίστοιχο endpoint:

```
1 useEffect(() => {
2   fetchDataKlados();
3   fetchDataPeriox_Topothet();
4   fetchDataTypos();
5   fetchDataDieythynsh();
6 }, []);
```

Απόσπασμα Κώδικα 5.4: Ανάκτηση δεδομένων κατά την αρχικοποίηση της οθόνης

Προσωπική αναζήτηση

Στην περίπτωση ενεργοποίησης της προσωπικής αναζήτησης, ενεργοποιείται ένα πεδίο τύπου DropDownPicker με searchable λειτουργία. Καθώς ο χρήστης πληκτρολογεί, η συνάρτηση fetchData αποστέλλει αίτημα τύπου POST προς το endpoint /api/anaplirotes, με σώμα που περιλαμβάνει την ιδιότητα name. Τα αποτελέσματα επιστρέφονται με τη μορφή λίστας και φορτώνονται στο picker με δυναμική ανανέωση.

Κατά την επιλογή ενός ονόματος και την ενεργοποίηση του κουμπιού Αναζήτηση, η εφαρμογή αποστέλλει αίτημα GET προς το endpoint /api/onoma/{formattedSearchValue}. Το αποτέλεσμα αυτής της κλήσης υφίσταται επεξεργασία με χρήση της βοηθητικής συνάρτησης groupData, η οποία ομαδοποιεί τα δεδομένα με βάση το πατρώνυμο (Patronymo). Στη συνέχεια:

- Αν υπάρχουν πολλαπλές εγγραφές, ενεργοποιείται ένα Modal, μέσω του οποίου ο χρήστης επιλέγει το πατρώνυμό του.
- Αν υπάρχει μία εγγραφή, ο χρήστης μεταφέρεται απευθείας στην οθόνη Result Screen.

- Αν δεν υπάρχουν εγγραφές, εμφανίζεται η Error Screen.

Η διαχείριση του modal γίνεται μέσω της κατάστασης isModalVisible, ενώ η πλοήγηση υλοποιείται με χρήση της useNavigation() του React Navigation.

```
<View style={styles.modalView}>
  <Text style={styles.modalTextΣυνωνυμία}></Text>

  <Text>{searchValue}</Text>
  {fatherNames.map((name, index) => (
    <TouchableOpacity key={index} style={styles.button} onPress
   ={() => handleNamePress(name)}>
      <Text style={styles.buttonText}>{name}</Text>
    </TouchableOpacity>
  ))}
</View>
```

Απόσπασμα Κώδικα 5.5: Εμφάνιση modal για συνωνυμία ονομάτων

Γενική αναζήτηση

Όταν η τιμή της μεταβλητής isEnabled είναι false, ενεργοποιείται η Γενική Αναζήτηση, η οποία βασίζεται σε τέσσερα διαδοχικά DropDownPicker components:

- Τύπος (typos)
- Κλάδος (klados)
- Περιοχή τοποθέτησης (perioxh_topothetisis)
- Διεύθυνση εκπαίδευσης (dieythynsh_ekpaideyshs)

Οι επιλογές αντλούνται δυναμικά κατά το component mount, μέσω των συναρτήσεων fetchDataTypos, fetchDataKlados, fetchDataPeriox_Topothet και fetchDataDieythynsh, οι οποίες εκτελούν κλήσεις τύπου GET στα αντίστοιχα endpoints.

Κατά την υποβολή της φόρμας, η συνάρτηση handleGenikiPress δημιουργεί δυναμικό query string με χρήση της encodeURIComponent() για κάθε φίλτρο και το αποστέλλει ως αίτημα τύπου GET προς το /api/search. Εφόσον τα δεδομένα περιλαμβάνουν αποτελέσματα, ο χρήστης μεταφέρεται στην οθόνη General Result Screen. Διαφορετικά, προβάλλεται η Error Screen.

Header & Footer

Η εφαρμογή περιλαμβάνει δύο κοινά οπτικά στοιχεία που ενσωματώνονται σχεδόν σε όλες τις οθόνες: το HeaderComponent και το FooterComponent. Το HeaderComponent εμφανίζεται στο άνω μέρος της οθόνης και περιλαμβάνει το όνομα της εφαρμογής «eAnaplirotes», το οποίο λειτουργεί και ως πλήκτρο επιστροφής στην αρχική σελίδα (Home Screen2), μέσω της μεθόδου navigation.navigate().

Αντίστοιχα, το FooterComponent τοποθετείται στο κάτω μέρος και παρέχει συνδέσμους για χρήσιμες εξωτερικές ενέργειες. Πιο συγκεκριμένα, προσφέρει:

- Πλοήγηση προς την οθόνη τεκμηρίωσης API μέσω της εντολής navigation.navigate('Api Document'),
- Σύνδεση με τον αποθετήριο του κώδικα στο GitHub,
- Άμεση επικοινωνία μέσω email με τον δημιουργό της εφαρμογής.

Τα δύο αυτά στοιχεία προσφέρουν συνοχή στην πλοήγηση της εφαρμογής και διασφαλίζουν ότι ο χρήστης έχει εύκολη πρόσβαση τόσο στις βασικές λειτουργίες όσο και σε πληροφορίες υποστήριξης.

Οθόνη Σφάλματος

Η οθόνη ErrorScreen ενεργοποιείται όταν ένα αίτημα επιστρέψει κενό αποτέλεσμα. Παρουσιάζει στον χρήστη ένα κατάλληλο μήνυμα μαζί με μια εικόνα και του προσφέρει επιλογή επιστροφής στην αρχική οθόνη.

```
<Image
  source={require('../../assets/error-image.webp')}
  style={styles.image}
/>
<Text style={styles.text}>
  Δεν βρέθηκαν αποτελέσματα για την αναζήτησή σας, επιστρέψτε
   στην{' '}
  <Text style={styles.link} onPress={() => navigation.navigate('
   Home Screen2')}>
    αρχική
  </Text>.
</Text>
```

Απόσπασμα Κώδικα 5.6: Οθόνη σφάλματος όταν δεν υπάρχουν αποτελέσματα

Οθόνη αποτελεσμάτων γενικής αναζήτησης

Όταν ο χρήστης πατάει το κουμπί "Αναζήτηση", ξεκινάει η εκτέλεση της συνάρτησης handleGenikiPress. Η εφαρμογή ελέγχει τι στοιχεία έχει βάλει ο χρήστης για την περιοχή τοποθέτησης. Αν έχει επιλέξει μόνο μία περιοχή, τότε το κείμενο που χρειάζεται για το URL φτιάχνεται με ειδικό τρόπο. Αν έχει διαλέξει περισσότερες περιοχές, η εφαρμογή φτιάχνει το URL έτσι ώστε να στείλει σωστά όλες τις επιλογές μαζί.

```
perioxhParams = searchValuePeriox_Topothet.length === 1
  ? `perioxh=${encodeURIComponent(searchValuePeriox_Topothet[0]
   || '')}&perioxh=`
```

```
3   : searchValuePeriox_Topothet.map(p => `perioxh=${
      encodeURIComponent(p || '')}`).join('&');
```

Απόσπασμα Κώδικα 5.7: Δημιουργία query string για πολλαπλές περιοχές τοποθέτησης

Αφού οργανωθούν όλα τα στοιχεία αναζήτησης (τύπος εκπαιδευτικού, κλάδος, περιοχή και διεύθυνση), δημιουργείται η διεύθυνση URL για να γίνει η αναζήτηση. Όλα τα δεδομένα κωδικοποιούνται σωστά, για να είναι έτοιμα να σταλούν στο διαδίκτυο χωρίς πρόβλημα. Στη συνέχεια, η εφαρμογή στέλνει ένα αίτημα μέσω του axios για να ζητήσει αποτελέσματα από το απομακρυσμένο API.

Μόλις έρθει η απάντηση, η εφαρμογή ελέγχει αν υπάρχουν αποτελέσματα. Αν δεν βρεθεί τίποτα, τότε ο χρήστης μεταφέρεται στην οθόνη σφάλματος (”Error Screen”) που τον ενημερώνει ότι δεν υπάρχουν αποτελέσματα. Αν όμως επιστραφούν δεδομένα, τότε ο χρήστης μεταφέρεται στην οθόνη ”General Result Screen”, όπου μπορεί να δει τα αποτελέσματα της αναζήτησής του. Μαζί με τη μεταφορά, η εφαρμογή στέλνει και τα στοιχεία που είχε βάλει ο χρήστης, ώστε να εμφανιστούν σωστά τα αποτελέσματα.

```
1  try {
2    const url = `https://eanaplirotes.iee.ihu.gr/api/search?typos=$
       {
3    encodeURIComponent(searchValueTypos || '')}&klados=${
       encodeURIComponent(searchValueKlados || '')}&${perioxhParams}&
       dieythynsh=${encodeURIComponent(
4    searchValueDieythynsh || '')}`;
5
6    const response = await axios.get(url);
7
8    if (response.data.anaplirotesCount.datasets.length === 0 &&
       response.data.anaplirotesCount.labels.length === 0) {
9      navigation.navigate('Error Screen');
10   } else {
11     navigation.navigate('General Result Screen', {
12       typos: searchValueTypos,
13       klados: searchValueKlados,
14       perioxh: searchValuePeriox_Topothet,
15       dieythynsh: searchValueDieythynsh
16     });
17   }
18 } catch {
19   console.error('Error fetching data: ', error);
20 }
```

Απόσπασμα Κώδικα 5.8: Αίτημα αναζήτησης και διαχείρισης αποτελεσμάτων ή σφάλματος

Το component GeneralResultScreen είναι υπεύθυνο για την απεικόνιση των αποτελεσμάτων που προκύπτουν από τη Γενική Αναζήτηση της αρχικής οθόνης. Η συγκεκριμένη οθόνη δέχεται

παραμέτρους μέσω της route.params, που περιλαμβάνουν τον τύπο εκπαιδευτικού (typos), τον κλάδο (klados), την περιοχή τοποθέτησης (perioxh) και τη διεύθυνση (dieythynsh).

Αμέσως μετά την απόδοση της οθόνης, εκτελείται ασύγχρονη αίτηση μέσω του axios προς το endpoint:

GET /api/search

Το url δημιουργείται δυναμικά και περιλαμβάνει ως «query string» τις επιλεγμένες τιμές των φίλτρων από τον χρήστη. Η παράμετρος perioxh γίνεται με ειδική διαχείριση ώστε να υποστηρίζεται η επιλογή πολλαπλών περιοχών τοποθέτησης.
Το αντικείμενο απόκρισης περιέχει δύο βασικά datasets:

- **anaplirotesCount**: περιλαμβάνει τις χρονιές (labels) και τα αντίστοιχα πλήθη αναπληρωτών (datasets).
- **minMoria**: περιλαμβάνει τα ελάχιστα μόρια εισαγωγής για την ίδια χρονική περίοδο, οργανωμένα κατά περιοχή.

Πριν την εμφάνιση των γραφημάτων, οι περιοχές (ή άλλες κατηγορίες) αποτυπώνονται σε ένα δυναμικό πλαίσιο. Ο χρήστης έχει τη δυνατότητα να ενεργοποιεί ή να απενεργοποιεί συγκεκριμένες περιοχές, ελέγχοντας ποια δεδομένα θα απεικονιστούν γραφικά. Η ενεργή κατάσταση κάθε περιοχής συνοδεύεται από μικρό έγχρωμο τετράγωνο που αντιστοιχεί στο χρώμα της γραμμής στο γράφημα. Όταν η περιοχή είναι ανενεργή, το όνομά της εμφανίζεται με διακριτή διαγράμμιση (textDecorationLine: line-through).
Οι παραπάνω πληροφορίες αποδίδονται γραφικά μέσω του πακέτου react-native-chart-kit και του component LineChart. Πιο συγκεκριμένα, τα δεδομένα μορφοποιούνται κατάλληλα ώστε:

- Κάθε γραμμή να αντιστοιχεί σε μία περιοχή (ή σε άλλο διαχωριστικό χαρακτηριστικό).
- Με το onDataPointClick προβάλλεται αναδυόμενο παράθυρο (Dialog) που εμφανίζει τα στοιχεία του συγκεκριμένου σημείου στο γράφημα.

Πλοήγηση στα γραφήματα

Εφόσον η αναζήτηση πραγματοποιήθηκε με τύπο, κλάδο και περιοχή τοποθέτησης, δημιουργείται αυτόματα και ένα δεύτερο γράφημα τύπου LineChart, το οποίο απεικονίζει τα ελάχιστα μόρια εισαγωγής ανά περιοχή και έτος. Ο χρήστης μπορεί να το δει πατώντας το ειδικό βελάκι (που ενεργοποιείται μόνο στην συγκεκριμένη περίπτωση) ώστε να σύρει τον γράφο ελάχιστα μόρια προς την οθόνη και αντίστοιχα να σύρει μακριά από αυτήν το Πλήθος Αναπληρωτών.

Πίνακας δεδομένων (DataTable)

Στο κάτω μέρος της οθόνης, εμφανίζεται ένας διαδραστικός πίνακας με τα ίδια δεδομένα. Ο πίνακας αποδίδεται μέσω του component DataTable από το react-native-paper και περιλαμβάνει:

- Τις χρονιές

- Το πλήθος αναπληρωτών
- (Προαιρετικά) τα ελάχιστα μόρια, εάν η αναζήτηση περιλαμβάνει κλάδο και περιοχή.

Όλα τα κελιά είναι component TouchableOpacity, επιτρέποντας την εμφάνιση λεπτομερούς πληροφορίας μέσω ενός Dialog, είναι χρήσιμο διότι η πληροφορία μπορεί να κρύβεται ή να έχει μικρό μέγεθος στις συσκευές με μικρότερες οθόνες.

Οθόνη αποτελεσμάτων προσωπικής αναζήτησης

Το component ResultScreen είναι υπεύθυνο για την παρουσίαση των αποτελεσμάτων που προκύπτουν από την επιλογή συγκεκριμένου ονοματεπωνύμου στην Προσωπική Αναζήτηση. Μέσω της παραμέτρου route.params, το component λαμβάνει το Eponymo, Onoma και Patronymo του αναπληρωτή.
Με την εμφάνιση της οθόνης, εκτελείται αυτόματα αίτημα προς το endpoint:

GET /api/user/Eponymo/Onoma/Patronymo

Η απάντηση του API περιλαμβάνει ένα πίνακα με όλες τις χρονιές στις οποίες έχει εργαστεί ο εκπαιδευτικός, συνοδευόμενες από τα αντίστοιχα μόρια, σειρά στον πίνακα και λοιπά στοιχεία τοποθέτησης.

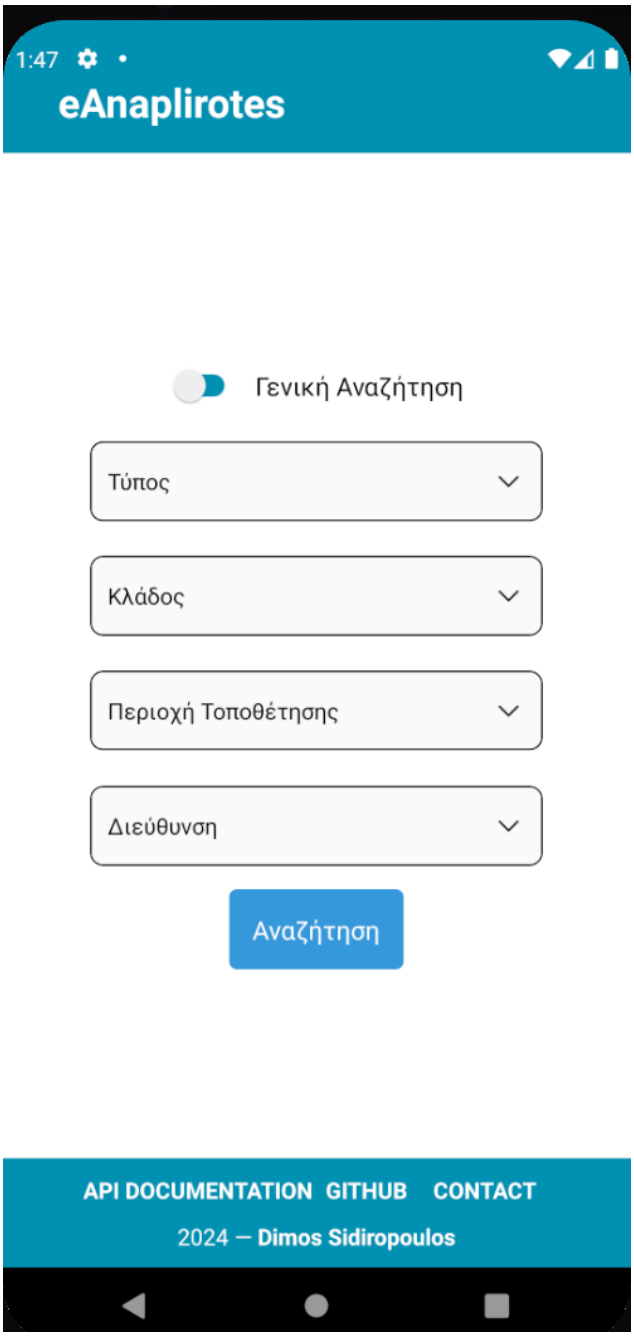
5.0.1 Github Repository

Ο open source κώδικας της Android εφαρμογής είναι ανεβασμένος σε ένα github repository το οποίο βρίσκεται στον σύνδεσμο `https://github.com/mikeApostolidis/Android_App_For_eAnaplirotes`.

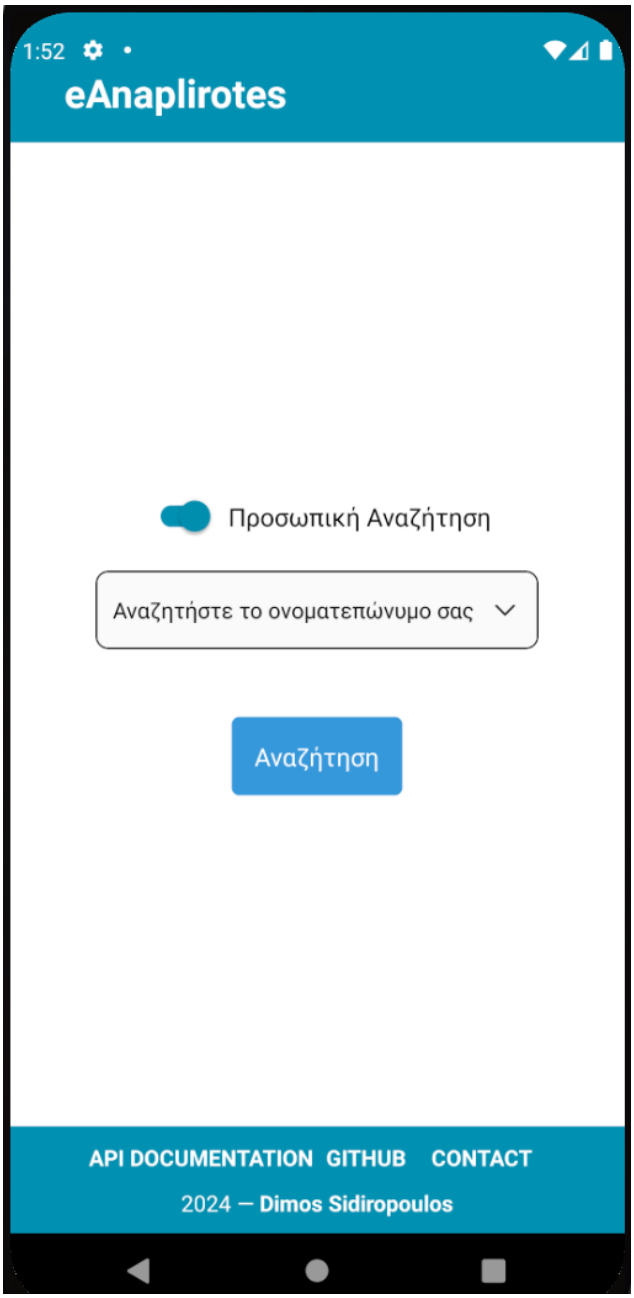
5.1 Παρουσίαση εφαρμογής

Σε αυτό το κεφάλαιο παρουσιάζεται το λειτουργικό και οπτικό κομμάτι της εφαρμογής, όπως αυτό προβάλλεται στον τελικό χρήστη. Έχοντας αναλύσει τον κώδικα στο προηγούμενο κεφάλαιο, η παρούσα ενότητα μιλάει για την διεπαφή χρήστη (UI), τον τρόπο πλοήγησης και την αλληλεπίδραση με τα βασικά στοιχεία της εφαρμογής.

Ανοίγοντας την εφαρμογή, η πρώτη οθόνη περιλαμβάνει την γενική αναζήτηση με όλα τα δικά της πεδία-φίλτρα, όπως φαίνεται στο Σχήμα 5.1. Επίσης, με το κουμπί switch ο χρήστης μπορεί να αλλάξει την αναζήτηση με βάση ονοματεπώνυμο η οποία έχει και αυτή το αντίστοιχο πεδίο της, όπως παρουσιάζεται στο Σχήμα 5.2.



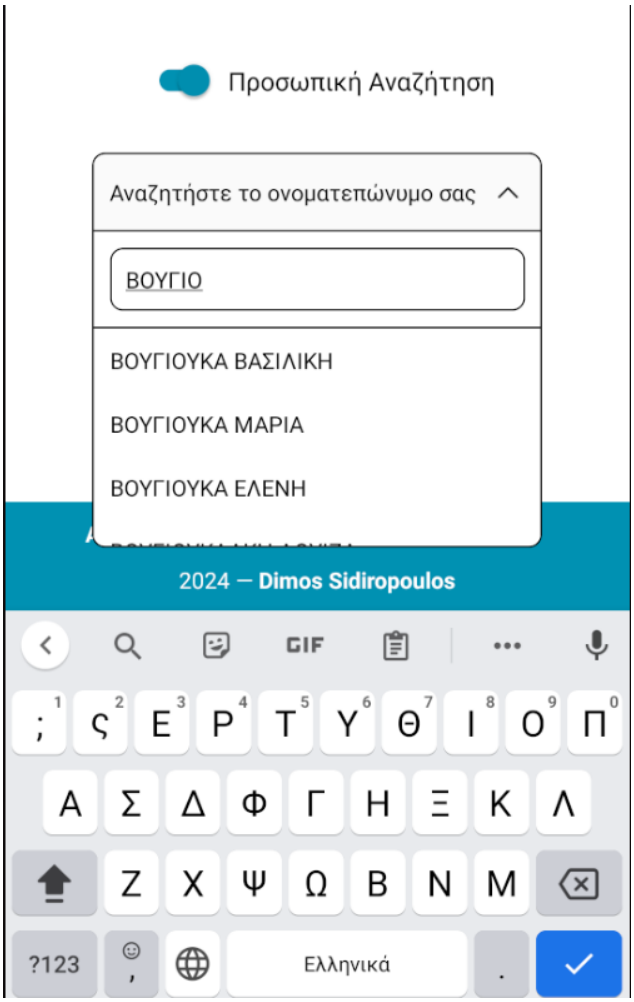
Σχήμα 5.1: Γενική αναζήτηση



Σχήμα 5.2: Προσωπική αναζήτηση

5.1.1 Προσωπική αναζήτηση

Στο Σχήμα 5.3 φαίνεται η δυνατότητα αυτόματης συμπλήρωσης ονόματος κατά την διάρκεια πληκτρολόγησης ώστε να βοηθήσει στην εύρεση του επιθυμητού προσώπου.



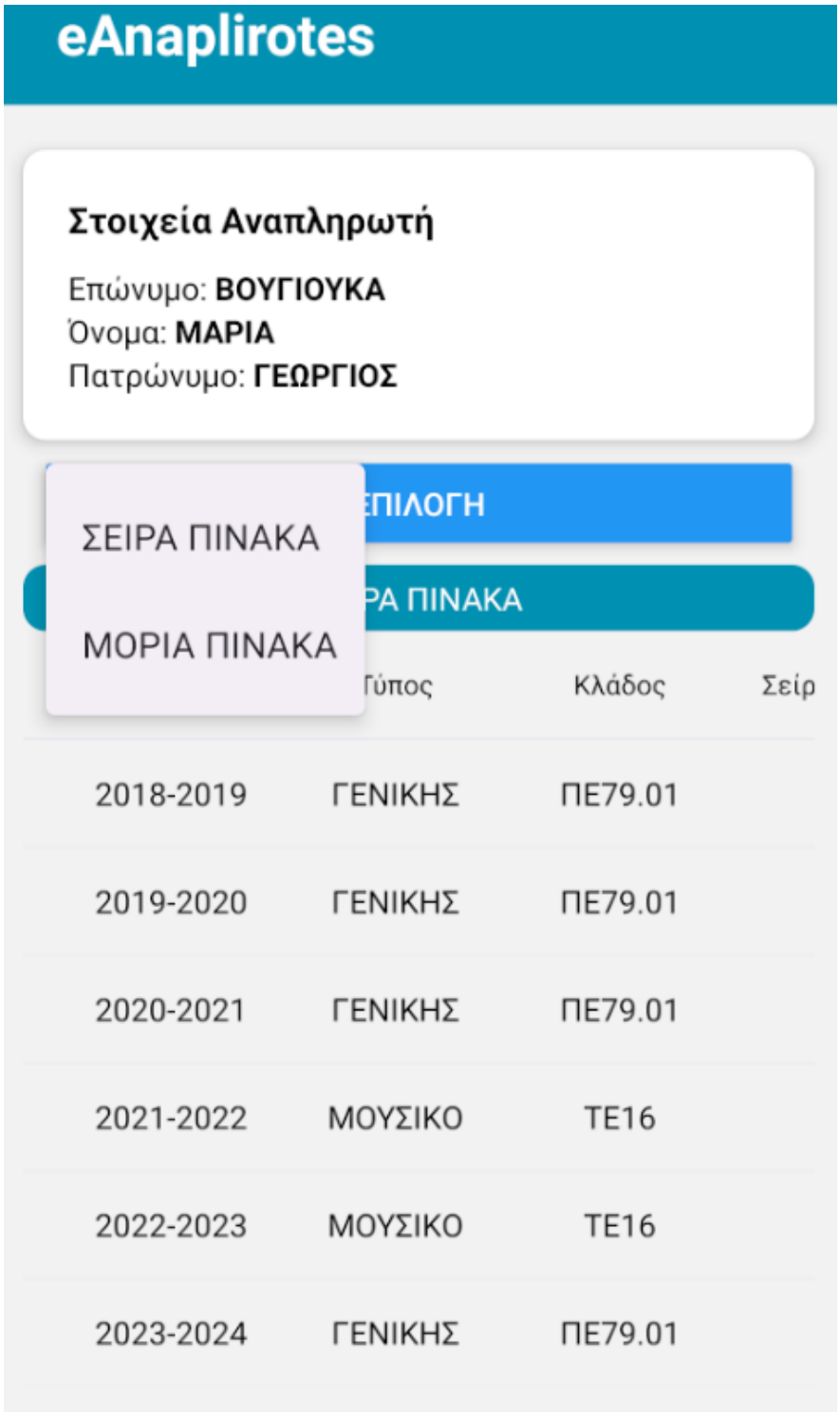
Σχήμα 5.3: Αυτόματη συμπλήρωση ονόματος

Στο Σχήμα 5.4 παρουσιάζεται ο διάλογος που εμφανίζεται στις περιπτώσεις που υπάρχει συνωνυμία, ώστε να επιλέξει ο χρήστης το όνομα του πατέρα.

Σχήμα 5.4: Διάλογος συνωνυμίας

5.1.2 Οθόνη αποτελεσμάτων προσωπικής αναζήτησης

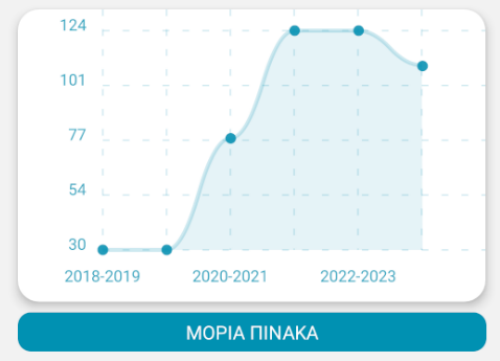
Η αναζήτηση βάση ονοματεπώνυμου επιστρέφει απο το backend την παρακάτω οθόνη. Αυτή αποτελείται από τα εξής στοιχεία: το πλαίσιο με τα στοιχεία του καθηγητή, ένα κουμπί που εμφανίζει τον επιθυμητό πίνακα (είτε Σειρά, είτε Μόρια) και τέλος, ένας πίνακας με χρήσιμες πληροφορίες, όπως απεικονίζεται στο Σχήμα 5.5.



Σχήμα 5.5: Επιλογή πίνακα

Η εφαρμογή προσφέρει γραφική απεικόνιση των δεδομένων με χρήση του component LineChart που ανήκει στο πακέτο react-native-chart-kit. Ο χρήστης μπορεί να επιλέξει μεταξύ δύο γράφων: Σειρά Πίνακα (Σχήμα 5.6) και Μόρια Πίνακα (Σχήμα 5.7).

Σχήμα 5.6: Σειρά πίνακα



Σχήμα 5.7: Μόρια πίνακα

Εφόσον επιλεχθούν οι κουκίδες στους γράφους, εμφανίζεται ένα παράθυρο με αναλυτικά στοιχεία του άξονα Χ και Υ, όπως αναπαρίσταται στο Σχήμα 5.8.

Σχήμα 5.8: Λεπτομέρειες στοιχείων

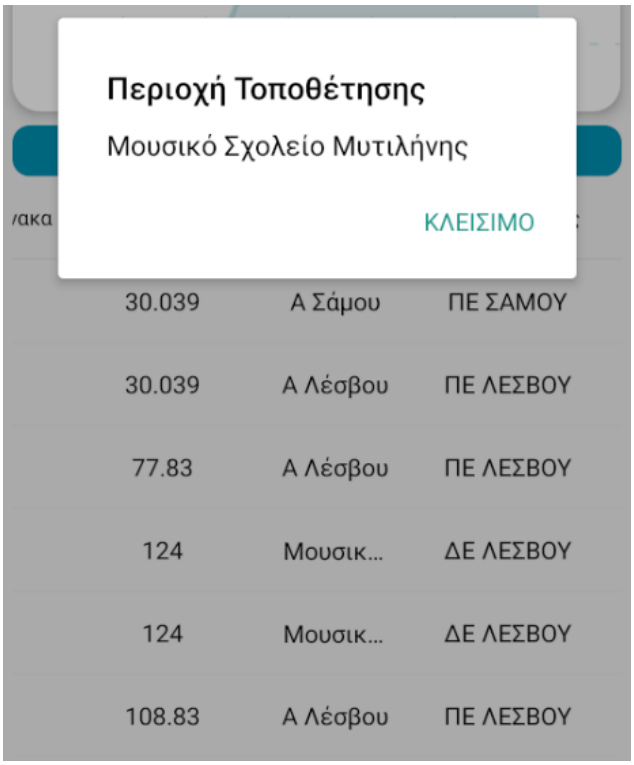
Τα ίδια δεδομένα εμφανίζονται και σε μορφή πίνακα μέσω του component DataTable από το πακέτο react-native-paper. Ο πίνακας περιλαμβάνει τις εξής στήλες:

- Έτος
- Τύπος
- Κλάδος
- Σειρά Πίνακα
- Μόρια Πίνακα
- Περιοχή Τοποθέτησης
- Διεύθυνση Εκπαίδευσης

Κάθε κελί είναι αλληλοεπιδράσημο, το οποίο σημαίνει πως μπορεί να επιλεχθεί απο τον χρήστη, η επιλογή του ενεργοποιεί έναν διάλογο που εμφανίζει αναλυτικά το περιεχόμενο του συγκεκριμένου πεδίου. Αυτό καθιστά την εμπειρία φιλική σε κινητές συσκευές όπου η πληροφορία μπορεί να εμφανίζεται αποκομμένη λόγω μικρής οθόνης. Το Σχήμα 5.9 παρουσιάζει τον πίνακα με τα διαδραστικά κελιά, ενώ το Σχήμα 5.10 αποτυπώνει τον διάλογο που εμφανίζεται με τις αναλυτικές πληροφορίες μετά την επιλογή ενός κελιού.

ακα	Μόρια Πίνακα	Π.Τοποθέτησης	Δ.Εκπαίδευσης
	30.039	Α Σάμου	ΠΕ ΣΑΜΟΥ
	30.039	Α Λέσβου	ΠΕ ΛΕΣΒΟΥ
	77.83	Α Λέσβου	ΠΕ ΛΕΣΒΟΥ
	124	Μουσικ...	ΔΕ ΛΕΣΒΟΥ
	124	Μουσικ...	ΔΕ ΛΕΣΒΟΥ
	108.83	Α Λέσβου	ΠΕ ΛΕΣΒΟΥ

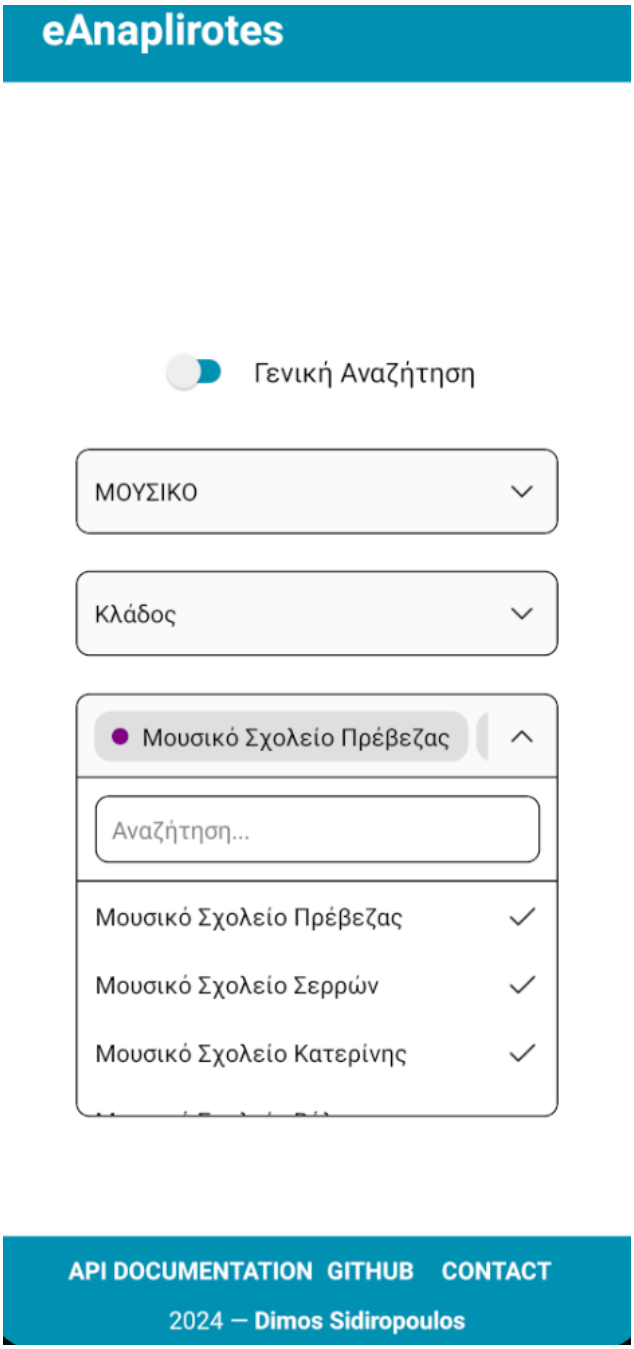
Σχήμα 5.9: Πίνακας αποτελεσμάτων



Σχήμα 5.10: Λεπτομέρειες πίνακα

5.1.3 Λεπτομέρειες πληροφοριών

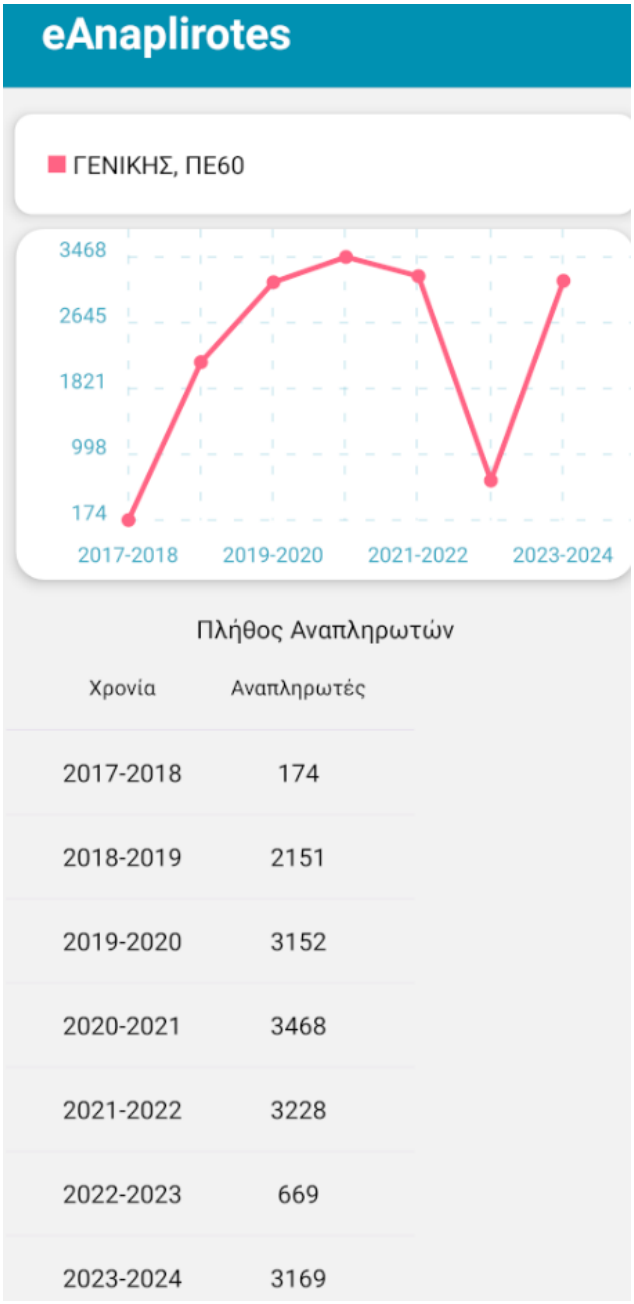
Αντίστοιχα, η οθόνη της Γενικής Αναζήτησης περιλαμβάνει τα δικά της πεδία συμπλήρωσης, τα οποία είναι: τύπος, κλάδος, περιοχή τοποθέτησης και διεύθυνση. Η αναζήτηση μπορεί να πραγματοποιηθεί με την επιλογή ενός ή περισσότερων από αυτά, σε οποιονδήποτε συνδυασμό. Σημειώνεται ότι μόνο η περιοχή τοποθέτησης υποστηρίζει την επιλογή πολλαπλών τιμών, ενώ τα υπόλοιπα πεδία δέχονται αυστηρά μία ή καμία τιμή, όπως φαίνεται στο Σχήμα 5.11.



Σχήμα 5.11: Παράδειγμα γενικής αναζήτησης

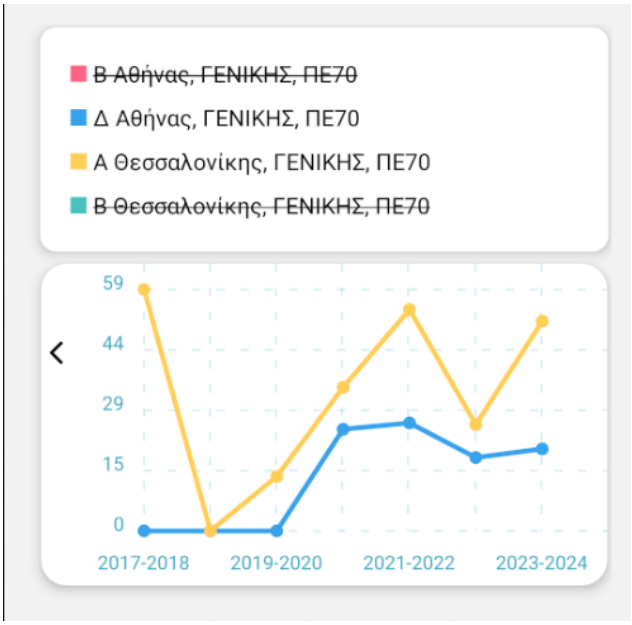
5.1.4 Οθόνη αποτελεσμάτων γενικής αναζήτησης

Όταν έχουν επιλεγεί τα κατάλληλα φίλτρα, η οθόνη αυτή ακολουθεί παρόμοια δομή με εκείνη της προσωπικής αναζήτησης. Περιλαμβάνει ένα τμήμα με τις επιλεγμένες πληροφορίες από την αρχική οθόνη, έναν γράφο και έναν πίνακα δεδομένων, όπως μπορούμε να δούμε στο Σχήμα 5.12.



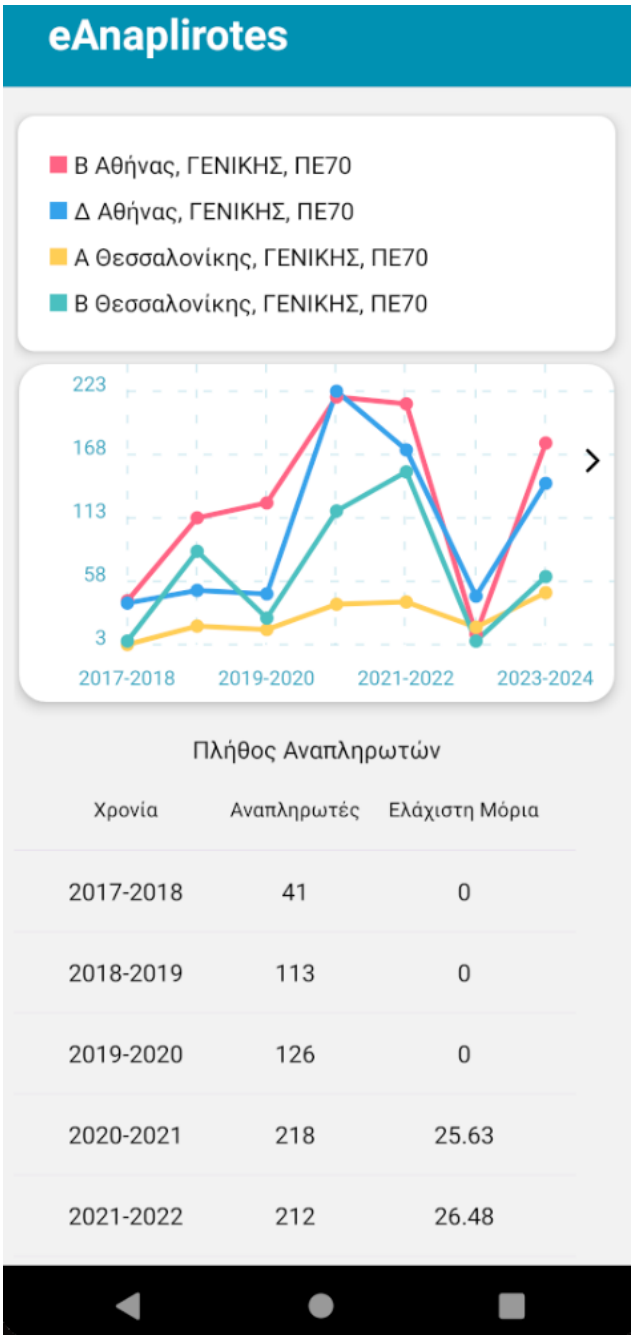
Σχήμα 5.12: Παράδειγμα γενικής αναζήτησης

Όταν εμφανίζονται δύο ή περισσότερες διευθύνσεις, ο χρήστης έχει τη δυνατότητα να αποκρύψει μία ή περισσότερες από αυτές, όπως παρουσιάζεται στο Σχήμα 5.13. Με αυτόν τον τρόπο μπορεί να απομονώσει τις πληροφορίες που τον ενδιαφέρουν, διευκολύνοντας την ανάγνωση του γραφήματος όταν περιλαμβάνει πολλούς τόπους.

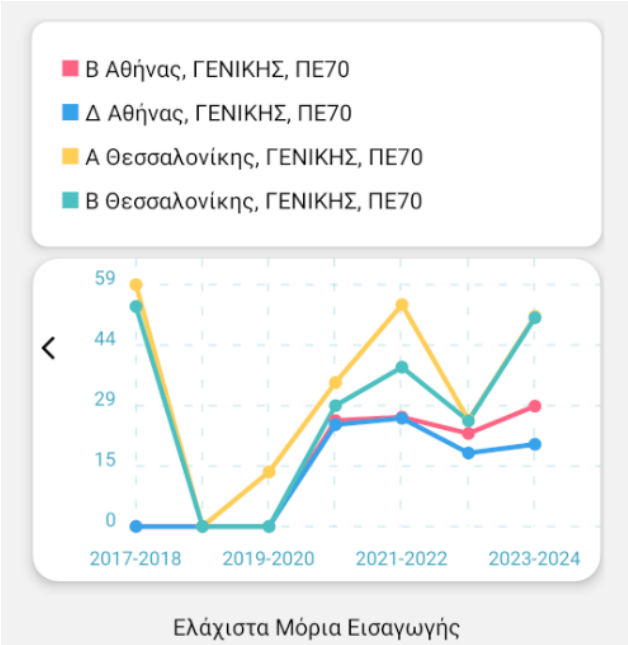


Σχήμα 5.13: Απομόνωση πληροφορίας

Όταν ο χρήστης έχει επιλέξει τόσο φίλτρο κλάδου όσο και φίλτρο περιοχής, εμφανίζεται ένας δεύτερος γράφος που απεικονίζει τα Ελάχιστα Μόρια Εισαγωγής για τις αντίστοιχες επιλογές. Το Σχήμα 5.14 παρουσιάζει τη συνολική εικόνα της οθόνης, ενώ το Σχήμα 5.15 προσφέρει μια πιο λεπτομερή απεικόνιση του γράφου μορίων.

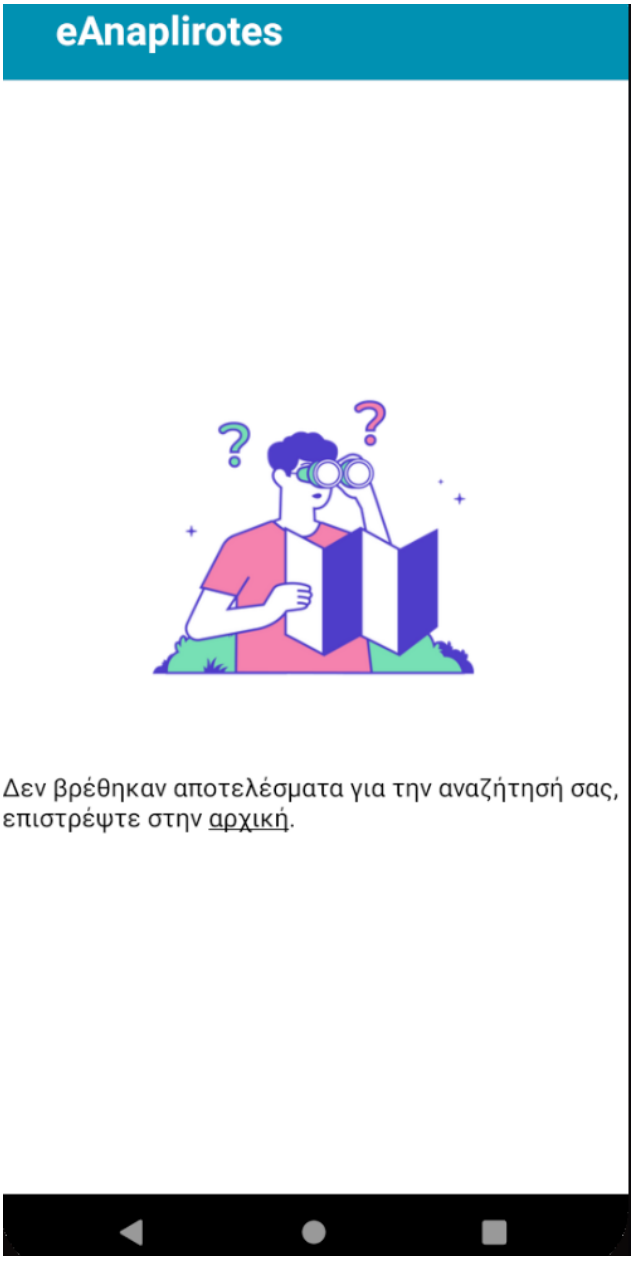


Σχήμα 5.14: Οθόνη αποτελεσμάτων γενικής αναζήτησης



Σχήμα 5.15: Παράδειγμα ελάχιστα μόρια εισαγωγής

Τέλος, σε περίπτωση που ο χρήστης επιλέξει έναν συνδυασμό φίλτρων που δεν αποδίδει κανένα αποτέλεσμα, για παράδειγμα, τύπο μουσικής σε συνδυασμό με κωδικό κλάδου εκπαιδευτικού ειδικής αγωγής, εμφανίζεται ένα κατάλληλο ενημερωτικό μήνυμα που τον ειδοποιεί για την απουσία αποτελεσμάτων, όπως απεικονίζεται στο Σχήμα 5.16.



Σχήμα 5.16: Οθόνη σφαλμάτων

Κεφάλαιο 6

Συμπεράσματα και Μελλοντικές επεκτάσεις

6.1 Συμπεράσματα

Η δημιουργία αυτού του έργου αποτελεί ένα σημαντικό μέρος της αρχικής πτυχιακής ολοκληρώνοντας την. Το σύστημα σχεδιάστηκε ώστε να λειτουργεί με ελάχιστη ανθρώπινη παρέμβαση, ξεκινώντας από την αυτόματη σύνδεση με την ιστοσελίδα του Υπουργείου Παιδείας, την αναγνώριση και λήψη των κατάλληλων αρχείων, έως και την πλήρη κανονικοποίηση και καθαρισμό των δεδομένων σε μορφή κατάλληλη για ανάλυση και αποθήκευση.

Με την δημιουργία της Android εφαρμογής δύνεται μια παραπάνω επιλογή για τον καθηγητή ως προς την χρήση της υπηρεσίας. Η Android εφαρμογή eAnaplirotes προσφέρει έναν εύχρηστο και ξεκάθαρο τρόπο αναζήτησης και παρουσίασης δεδομένων για αναπληρωτές εκπαιδευτικούς.

6.2 Μελλοντικές επεκτάσεις

Μία μελλοντική επέκταση της πτυχιακής θα μπορούσε να είναι η προσθήκη ενός μηχανισμού ειδοποίησης των χρηστών μέσω email. Συγκεκριμένα, μετά την καταχώρηση νέων δεδομένων πρόσληψης στη βάση, το σύστημα θα μπορούσε να ενημερώνει αυτόματα τους καθηγητές που έχουν εγγραφεί δηλώνοντας το email τους, αποστέλλοντάς τους σχετική ειδοποίηση. Με αυτόν τον τρόπο, θα μειωνόταν η ανάγκη για συνεχή έλεγχο από μέρους των χρηστών, διευκολύνοντας την άμεση και έγκαιρη ενημέρωσή τους. Παράλληλα, μία ακόμη χρήσιμη επέκταση θα ήταν η ανάπτυξη της εφαρμογής και για συσκευές iOS, προκειμένου να καλύπτεται μεγαλύτερο εύρος χρηστών και να αυξηθεί η διάδοση και χρηστικότητα του συστήματος.

Βιβλιογραφία

[1] N. Nikolakakos and D. Vergidis, "Substitute teachers in greece: The pursuit for academic qualifications and its consequences in everyday life," *European Journal of Education Studies*, vol. 11, no. 8, 2024.

[2] "Προσλήψεις εκπαιδευτικών - Πίνακες ΑΣΕΠ: Τι μοριοδοτείται για τη σειρά κατάταξης — alfavita.gr." https://www.alfavita.gr/node/406234. [Accessed 04-11-2024].

[3] "opendatahandbook.org." https://opendatahandbook.org/guide/el/what-is-open-data/. [Accessed 25-04-2025].

[4] "opendata.ellak.gr." https://opendata.ellak.gr/2017/07/14/open-data-ti-ine-ke-pos-mporoun-na-allaxoun-ti-zoi-sou/. [Accessed 25-04-2025].

[5] "Open data and its usability: an empirical view from the Citizen&x2019;s perspective — link.springer.com." https://link.springer.com/content/pdf/10.1007/s10796-016-9679-1.pdf. [Accessed 25-04-2025].

[6] "The economic benefits of Open Data | data.europa.eu — data.europa.eu." https://data.europa.eu/en/publications/datastories/economic-benefits-open-data. [Accessed 25-04-2025].

[7] "data.gov.gr — data.gov.gr." https://data.gov.gr. [Accessed 25-04-2025].

[8] "Τα ανοικτά δημόσια δεδομένα βασικός πυλώνας του ψηφιακού μετασχηματισμού της χώρας." https://www.gsis.gr/sites/default/files/Secdigital/Researches/SR_Open_Data_8_10_2018.pdf. [Accessed 25-04-2025].

[9] M. Lutz, *Learning Python*. Sebastopol, CA: O'Reilly Media, 5 ed., 2013.

[10] "TIOBE Index - TIOBE — tiobe.com." https://www.tiobe.com/tiobe-index/. [Accessed 26-11-2024].

[11] "Technology | 2024 Stack Overflow Developer Survey — survey.stackoverflow.co." https://survey.stackoverflow.co/2024/technology#most-popular-technologies-language-other. [Accessed 04-11-2024].

[12] D. Flanagan, *JavaScript: The Definitive Guide: Master the World's Most-Used Programming Language*. O'Reilly Media, 7 ed., 2020.

[13] "Introduction - JavaScript | MDN — developer.mozilla.org." https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction. [Accessed 05-05-2025].

[14] "React – A JavaScript library for building user interfaces — legacy.reactjs.org." legacy.reactjs.org/. [Accessed 05-05-2025].

[15] "React Native · Learn once, write anywhere — reactnative.dev." reactnative.dev/. [Accessed 05-05-2025].

[16] "Introduction · React Native — reactnative.dev." https://reactnative.dev/docs/getting-started. [Accessed 05-05-2025].

[17] "Introduction — docs.expo.dev." https://docs.expo.dev/get-started/introduction/. [Accessed 05-05-2025].

[18] "DB-Engines Ranking — db-engines.com." https://db-engines.com/en/ranking. [Accessed 05-05-2025].

[19] "MySQL :: MySQL 8.0 Reference Manual :: 1.2.3 History of MySQL — dev.mysql.com." https://dev.mysql.com/doc/refman/8.0/en/history.html. [Accessed 26-05-2025].

[20] "1.2.2&xA0;The Main Features of MySQL — docs.oracle.com." https://docs.oracle.com/cd/E17952_01/mysql-8.0-en/features.html. [Accessed 26-05-2025].