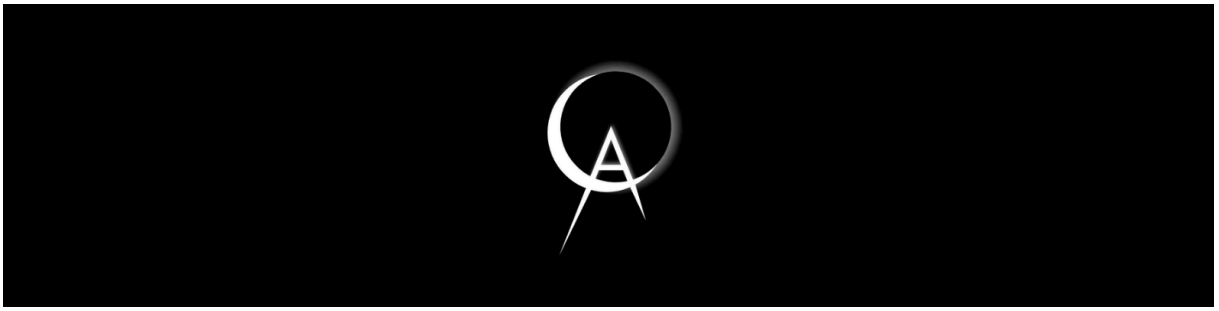
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Δημιουργία εφαρμογής για κινητές συσκευές (Android) που θα λειτουργεί ως προσωπικός βοηθός για άτομα με προβλήματα όρασης.»



Του φοιτητή
Βαρβάρη Πολυχρόνη
Αρ. Μητρώου: 185152

Επιβλέπων
Ονοματεπώνυμο Γεώργιος
Κοκκώνης
Επίκουρος Καθηγητής

Ημερομηνία Ιανουάριος 2025

Τίτλος Δ.Ε. Δημιουργία εφαρμογής για κινητές συσκευές (Android) που θα λειτουργεί ως προσωπικός βοηθός για άτομα με προβλήματα όρασης

Κωδικός Δ.Ε. 24125

Ονοματεπώνυμο φοιτητή/τών Βαρβάρης Πολυχρόνης

Ονοματεπώνυμο εισηγητή Γεώργιος Κοκκώνης

Ημερομηνία ανάληψης Δ.Ε. 19/02/2024

Ημερομηνία περάτωσης Δ.Ε. 25/01/2025

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

«Αφιέρωση»

Πρόλογος

Η επιλογή του θέματος της παρούσας διπλωματικής εργασίας προέκυψε από την προσωπική μου εμπειρία και τον έντονο προβληματισμό σχετικά με τις προκλήσεις που αντιμετωπίζουν άτομα με ακουστικές ή οπτικές δυσκολίες. Στο παρελθόν, βρέθηκα αντιμέτωπος με την πιθανότητα απώλειας της ακοής μου, γεγονός που με ώθησε να κατανοήσω βαθύτερα τις ανάγκες ατόμων που ζουν με αισθητηριακές αναπηρίες.

Παρακολουθώντας τη ραγδαία εξέλιξη της τεχνολογίας, αντιλήφθηκα τις δυνατότητες που προσφέρει για τη βελτίωση της ζωής ατόμων με δυσκολίες. Σκοπός μου μέσω αυτής της διπλωματικής ήταν να σχεδιάσω μια εφαρμογή που ενσωματώνει σύγχρονα τεχνολογικά εργαλεία, προκειμένου να διευκολύνει την καθημερινότητα ατόμων με ακουστικές ή οπτικές αναπηρίες.

Η διπλωματική μου εργασία μού έδωσε την ευκαιρία να συνδυάσω τη γνώση με την ενσυναίσθηση και την καινοτομία. Ελπίζω αυτή η εφαρμογή να αποτελέσει την αφετηρία για τη δημιουργία νέων και πιο ουσιαστικών εργαλείων, που θα συνεχίσουν να στηρίζουν την ένταξη των ανθρώπων με αναπηρίες στην κοινωνία και να προάγουν έναν κόσμο χωρίς αποκλεισμούς.

Περίληψη

Η παρούσα διπλωματική εργασία επικεντρώνεται στη δημιουργία μιας εφαρμογής σχεδιασμένης να βελτιώσει την ποιότητα ζωής ατόμων με ακουστικές ή οπτικές αναπηρίες. Η εφαρμογή αξιοποιεί τεχνολογικά εργαλεία για να παρέχει λύσεις που διευκολύνουν την επικοινωνία, την κατανόηση του χώρου, την κοινωνικη ευαισθητοποίηση και την καθημερινή λειτουργικότητα.

Κατά την ανάπτυξη της εφαρμογής, ενσωματώθηκαν σύγχρονες τεχνολογίες αναγνώρισης φωνής και εικόνας, καθώς και εργαλεία για την επεξεργασία δεδομένων σε πραγματικό χρόνο. Ο στόχος ήταν να δημιουργηθεί μια εύχρηστη και προσβάσιμη διεπαφή που να ανταποκρίνεται στις ανάγκες του χρήστη. Μέσα από ανάλυση, σχεδιασμό και δοκιμές, διασφαλίστηκε η αποδοτικότητα και η αξιοπιστία της εφαρμογής σε πραγματικές συνθήκες.

Τα αποτελέσματα έδειξαν ότι η εφαρμογή μπορεί να λειτουργήσει ως ένα πολύτιμο εργαλείο υποστήριξης και εκμάθησης για άτομα με αισθητηριακές αναπηρίες, προσφέροντας ευκολία και αυτονομία στην καθημερινότητά τους. Η χρηστικότητα και η αποτελεσματικότητα της εφαρμογής αξιολογήθηκαν μέσω πειραματικής δοκιμής με χρήστες και μέσω δοκιμαστικών βάσεων δεδομένων (test dataset), καταγράφοντας τα αποτελέσματα και προτάσεις βελτίωσης.

Η εργασία αυτή αποδεικνύει πώς η τεχνολογία μπορεί να γίνει κινητήριος δύναμη για έναν πιο προσβάσιμο και ισότιμο κόσμο. Παράλληλα, θέτει τις βάσεις για περαιτέρω ανάπτυξη της εφαρμογής, με στόχο τη δημιουργία πιο ολοκληρωμένων εργαλείων που να ανταποκρίνονται σε ευρύτερες ανάγκες, με μεγαλύτερα και πιο ποικιλόμορφα σύνολα δεδομένων (datasets), τα οποία θα ενισχύσουν τη λειτουργικότητα και την ακρίβεια των εργαλείων, διασφαλίζοντας ακόμα πιο αποδοτική στήριξη των χρηστών.

«Creation of an application for mobile devices (Android) that will function as a personal assistant for people with visual impairments.»

«Varvaris Polihronis»

Abstract

This thesis focuses on the creation of an application designed to improve the quality of life for people with hearing or visual disabilities. The application utilizes technological tools to provide solutions that facilitate communication, spatial understanding, social awareness and daily functionality.

During the development of the application, modern voice and image recognition technologies were integrated, as well as tools for real-time data processing. The goal was to create an easy-to-use and accessible interface that meets the needs of the user. Through analysis, design and testing, the efficiency and reliability of the application in real conditions were ensured.

The results showed that the application can function as a valuable support and learning tool for people with sensory disabilities, offering convenience and autonomy in their daily lives. The usability and effectiveness of the application were evaluated through experimental testing with users and through test databases (test dataset), recording the results and suggestions for improvement.

This work demonstrates how technology can be a driving force for a more accessible and equitable world. At the same time, it lays the foundation for further development of the application, aiming to create more comprehensive tools that respond to broader needs, with larger and more diverse datasets, which will enhance the functionality and accuracy of the tools, ensuring even more efficient support for users.

Ευχαριστίες

Θα ήθελα να εκφράσω τις θερμές μου ευχαριστίες σε όλους όσους συνέβαλαν στην ολοκλήρωση αυτής της διπλωματικής εργασίας.

Πρώτα απ’ όλα, ευχαριστώ τον επιβλέποντα καθηγητή μου, κύριο Γεώργιο Κοκκώνη, για την καθοδήγηση, τη στήριξη και τις πολύτιμες συμβουλές του καθ' όλη τη διάρκεια του έργου. Η εμπειρία και η υποστήριξή του ήταν καθοριστική για την πρόοδό μου.

Ευχαριστώ επίσης την οικογένειά μου για την αμέτρητη αγάπη, την υπομονή και την κατανόηση που μου παρείχαν σε κάθε στάδιο της εργασίας μου. Η ενθάρρυνση και η πίστη τους με ενδυνάμωσαν σε στιγμές αμφιβολίας.

Τέλος, θα ήθελα να ευχαριστήσω τον εαυτό μου για την επιμονή, την υπομονή και την αφοσίωση που έδειξα καθ' όλη τη διάρκεια της εργασίας. Οι προσωπικές μου προσπάθειες και η συνεχής αναζήτηση για τη βελτίωση της εφαρμογής συνέβαλαν στην επιτυχία του έργου, και είμαι υπερήφανος για την πρόοδο που έχω κάνει.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κατάλογος Εικόνων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
GSL	Ελληνική Νοηματική Γλώσσα
ASL	Αμερικάνικη Νοηματική Γλώσσα
BSL	Αγγλική Νοηματική Γλώσσα
AI	Τεχνητή Νοημοσύνη
SCTS	Subtitles Course Tracking System
OCR	Οπτική Αναγνώριση Χαρακτήρων
TTS	Κείμενο σε Ομιλία
UI	Διεπαφή Χρήστη
UX	Εμπειρία Χρήστη
URI	Uniform Resource Identifier
DB	Ντεσιμπέλ
TFLite	TensorFlow Lite
IDE	Ολοκληρωμένο Περιβάλλον Ανάπτυξης
AAC	Προηγμένος Κωδικοποιητής Ήχου
CNN	Νευρωνικά Δίκτυα

Κεφάλαιο 1ο: Σκοπός και Ανάγκες

1.1 Εισαγωγή

Το πρώτο κεφάλαιο αποσκοπεί στην ανασκόπηση των αναγκών που έχουν άτομα με αναπηρίες, εστιάζοντας στις δυσκολίες που αντιμετωπίζουν στην καθημερινότητά τους και στην ανάγκη για την κοινωνική τους ενσωμάτωση. Θα εξετάσουμε τα στατιστικά δεδομένα απο επίσημες πηγές, που αποδεικνύουν τη σημασία της ανάπτυξης τεχνολογικών λύσεων για την υποστήριξή των συγκεκριμένων ατόμων και την ενίσχυση της ανεξαρτησία απο το κοινωνικό τους περιβάλλον. Στη συνέχεια, θα αναλύσουμε τις τεχνολογίες και τα εργαλεία που χρησιμοποιούνται για την υποστήριξη τους, δίνοντας βάση στις νέες τεχνολογίες που διευκολύνουν την επικοινωνία και αλληλεπίδραση τους με το περιβάλλον.

1.2 Στατιστικά Στοιχεία

Οι οπτικές και ακουστικές αναπηρίες αποτελούν σημαντικά ζητήματα που επηρεάζουν τη ζωή πολλών ατόμων παγκοσμίως. Ειδικότερα, οι οπτικές αναπηρίες σαν κατηγορία αισθητικών αναπηριών περιλαμβάνουν οι ιδιες διάφορες επιμέρους ομάδες, όπως η τύφλωση η μερική τύφλωση, η νυκταλωπία, η δυσχρωματοψία, η φωτοφοβία, η αχρωματοψία και η στενοπραξία. Σύμφωνα με τον Παγκόσμιο Οργανισμό Υγείας (WHO)[1][2], περίπου **2,2 δισεκατομμύρια** άνθρωποι παγκοσμίως ζουν με προβλήματα όρασης, εκ των οποίων το **1 δισεκατομμύριο** θα μπορούσε να αποφευχθεί ή να αντιμετωπιστεί μέσω κατάλληλης ιατρικής φροντίδας. Παγκοσμίως, μόνο το **36%** των ατόμων με μυωπία και μόλις το **17%** των ατόμων με καταρράκτη έχουν πρόσβαση σε κατάλληλη θεραπεία ή βοηθητικα αντικειμενα οπως φακούς επαφής και διορθωτικά γυαλιά.

Παρά τις τεχνολογικές εξελίξεις τα άτομα με οπτική αναπηρία εξακολουθούν να αντιμετωπίζουν σοβαρές δυσκολίες στην καθημερινή τους ζωή, όπως στη μετακίνηση, την εκπαίδευση και την εργασία. Οι παραδοσιακές υποδομές δεν είναι πάντα προσβάσιμες, ενώ οι προκλήσεις εντείνονται για άτομα που δεν έχουν την οικονομική δυνατότητα να αποκτήσουν ειδικά βοηθήματα ή να εξασφαλίσουν εξειδικευμένη φροντίδα. Στην Ελλάδα, η Εθνική Στατιστική Υπηρεσία αναφέρει ότι πάνω από **150.000** άτομα αντιμετωπίζουν προβλήματα όρασης, με το **5%** αυτών να είναι πλήρως τυφλοί ή να αντιμετωπίζουν σοβαρές οπτικές δυσκολίες [3].

Αντίστοιχα, οι ακουστικές αναπηρίες αφορούν άτομα που έχουν απώλεια ακοής, είτε μερική είτε ολική. Ο Παγκόσμιος Οργανισμός Υγείας εκτιμά ότι περίπου **1.5 δισεκατομμύρια** άνθρωποι παγκοσμίως ζουν με κάποια μορφή ακουστικής αναπηρίας, με το **430 εκατομμύρια** από αυτούς να χρειάζονται ιατρικές μεθόδους αποκατάστασης για την απώλεια ακοής τους. Πάνω από το **30%** των περιπτώσεων απώλειας ακοής στα παιδιά οφείλονται σε ασθένειες όπως η ιλαρά, η παρωτίτιδα, η ερυθρά, η μηνιγγίτιδα και οι λοιμώξεις του αυτιού. Εκτιμάται ότι έως και **330** εκατομμύρια άνθρωποι παγκοσμίως αντιμετωπίζουν χρόνια ωτίτιδα. Επίσης **5** στα **1000** βρέφη γεννιούνται με απώλεια ακοής ή αποκτούν λίγο μετά τη γέννηση τους. [4][5].

Οι αιτίες των ακουστικών αναπηριών περιλαμβάνουν κληρονομικές διαταραχές, τραυματισμούς, έκθεση σε θόρυβο, ασθένειες και άλλους περιβαλλοντικούς παράγοντες. Οι κύριοι τύποι ακουστικών αναπηριών είναι η μερική/σοβαρή απώλεια ακοής, η ολική κώφωση (εκ γενετής ή επίκτητο) και την κωφότητα από θόρυβο (που προκαλείται από παρατεταμένη έκθεση σε δυνατούς ήχους).

Τα άτομα αυτά αντιμετωπίζουν σοβαρές δυσκολίες στην επικοινωνία, την εκπαίδευση και τη συναισθηματική τους ανάπτυξη, ενώ συχνά αντιμετωπίζουν κοινωνικό στιγματισμό. Ειδικά για τα παιδιά, η έλλειψη προσβάσιμης εκπαίδευσης μπορεί να περιορίσει σημαντικά τις μελλοντικές τους ευκαιρίες. Στην Ελλάδα, η Εθνική Στατιστική Υπηρεσία αναφέρει ότι περίπου **1,5 εκατομμύρια**

άτομα αντιμετωπίζουν κάποιου είδους ακουστικής δυσλειτουργίας, με περίπου **300.000 άτομα** να έχουν σοβαρή ή ολική απώλεια ακοής [6].

1.3 Προκλήσεις και Επιπτώσεις

1.3.1 Στο Άτομο

Τα άτομα με οπτικές και ακουστικές αναπηρίες αντιμετωπίζουν μεγάλο αριθμό προκλήσεων που επηρεάζουν όχι μόνο την καθημερινότητα τους, αλλά και την κοινωνική τους ενσωμάτωση και συμμετοχή. Οι δυσκολίες αυτές εκτείνονται σε διάφορους τομείς του περιβάλλοντος τους, όπως η εκπαίδευση, η μετακίνηση, η κοινωνικές σχέσεις, η επαγγελματική τους πορεία και η συναισθηματική υποστήριξη.

- **Εκπαίδευση** Στην εκπαίδευση, οι μαθητές με οπτικές και ακουστικές αναπηρίες αντιμετωπίζουν 3 κύρια προβλήματα. Την δυσκολία επικοινωνίας με τους υπόλοιπους μαθητές, την αργή μεταφορά πληροφορίας από τον εκπαιδευτή προς το μαθητευόμενο άτομο και τα μη επαρκή εργαλεία αποθήκευσης της πληροφορίας. Το πρόβλημα της επικοινωνίας είναι το βασικότερο πρόβλημα για άτομα με ακουστικές αναπηρίες, ειδικά σε μικρες ηλικίες, που του άτομο αποκτά την πρώτη του επαφή με μια κοινωνική ομάδα και είναι επικίνδυνο να περιθωριοποιηθεί. Οι μαθητες πολλες φορες δεν μπορουν να παρακολουθούν και να συμμετέχουν ενεργά σε σχολεία ,τα οποια δεν εχουν την δυνατότητα να παρέχουν μεταφραστες ή καθηγητές με ειδίκευση στην νοηματική γλώσσα. Αυτο είναι και το πρόβλημα της μεταφοράς πληροφορίας που αναφέρθηκε προηγουμένως. Τελος η αποθηκευση της πληροφορίας ασχολείται με τον τρόπο ο μαθητής μπορεί να αποθηκευση πληροφορια για μελλοντική χρήση. Για παράδειγμα, οι συγκεκριμένοι μαθητες για να ασχοληθούν με τις σχολικές τους υποχρεώσεις, χρειαζονται καποια οπτικοακουστικα μέσα όπως ταμπλετ, ακουστικά, ειδικό στυλό (Braille Pen) και κωδικοποιημένα βιβλία σε γλώσσα Μπράιγ. Το κόστος αυτών των εργαλείων μπορεί να είναι ιδιαίτερα υψηλό και να επιβαρύνει ακόμα περισσοτερό τις ευαισθητες κοινωνικές ομάδες.
- **Μετακίνηση και Πρόσβαση στους Δημόσιους Χώρους** Η καθημερινή μετακίνηση είναι μια από τις πιο απαιτητικές πτυχές της ζωής των ατόμων με αναπηρίες όρασης ή ακοής. Οι ανεπαρκείς υποδομές και η έλλειψη προσβάσιμων μέσων μεταφοράς και δημόσιων χώρων καθιστούν δύσκολη έως και αδύνατη την αυτόνομη πλοήγηση τους. Η έλλειψη ηχητικών και οπτικών σημάνσεων, όπως και η απουσία τεχνολογικών βοηθημάτων, μειώνουν την αυτονομία τον προαναφερθέντων ατόμων και καθιστούν επικίνδυνη καθε τους προσπαθεια για την πλήρης αυτονομία στη μετακίνηση τους.
- **Κοινωνική Ζωή και Σχέσεις** Τα άτομα με αναπηρίες, είτε όρασης είτε ακοής, αντιμετωπίζουν σημαντικές προκλήσεις στην επικοινωνία και τη διατήρηση κοινωνικών σχέσεων. Η δυσκολία επικοινωνίας, η έλλειψη κοινωνικής αποδοχής και ο κοινωνικός στιγματισμός οδηγούν συχνά το άτομο σε απομόνωση. Η έλλειψη κατανόησης για τις ανάγκες τους με συνδυασμό την καθημερινή εξέλιξη της τεχνολογίας, έχει σαν αντίκτυπο να μεγαλώνει το χάσμα μεταξύ των κοινωνικών ομάδων. Για τον λόγο αυτό, οι τεχνολογίες που αναπτύσσονται πρέπει να έχουν ως κύριο στόχο τη βελτίωση της καθημερινότητας όλων των ατόμων, εμβαθύνοντας στην ενσωμάτωση εργαλείων και λειτουργιών που βασίζονται στις ανάγκες τους.
- **Εργασία και Οικονομική Αξιοπρέπεια** Στην αγορά εργασίας, τα άτομα αυτά συχνά αντιμετωπίζουν διακρίσεις και περιορισμένες ευκαιρίες. Οι ελλιπείς υποδομές, η έλλειψη

προσβάσιμων εργαλείων και η αδυναμία ένταξης στο επαγγελματικό του περιβάλλον λόγω δυσκολίας επικοινωνίας, δυσκολεύουν την επαγγελματική τους αποκατάσταση. Παρά τις νομοθετικές προστασίες, η πραγματική ένταξή τους παραμένει περιορισμένη λόγω προκαταλήψεων και ελλιπούς κατανόησης των αναγκών τους.

- **Συναισθηματική και Ψυχολογική Υποστήριξη** Η καθημερινή αντιμετώπιση των δυσκολιών και η αίσθηση απομόνωσης μπορούν να προκαλέσουν άγχη, κατάθλιψη και απογοήτευση. Η ανάγκη για κατάλληλη ψυχολογική υποστήριξη και επαγγελματίες που κατανοούν τις ιδιαίτερες ανάγκες των ατόμων με αισθητηριακές δυσκολίες είναι κρίσιμη για τη βελτίωση της ποιότητας ζωής τους.

1.3.2 Στην Κοινωνία

Οι προκλήσεις που αντιμετωπίζουν τα άτομα με οπτικές ή ακουστικές αναπηρίες δεν επηρεάζουν μόνο τα ίδια τα άτομα, αλλά έχουν και ευρύτερες κοινωνικές επιπτώσεις. Η κοινωνική απομόνωση, η έλλειψη πρόσβασης στην εκπαίδευση και στην αγορά εργασίας, καθώς και η περιορισμένη συμμετοχή σε κοινωνικές και πολιτιστικές δραστηριότητες, δημιουργούν μια κοινωνία που δεν αξιοποιεί πλήρως τις δυνατότητες όλων των μελών της. Σύμφωνα με άρθρο του 2023 του Παγκόσμιο Οργανισμό Υγείας (WHO), η μη αξιοποίηση των ατόμων αυτόν επιβαρύνει σημαντικά την παγκόσμια οικονομία, με το ετήσιο κόστος να εκτιμάται κοντά στα 411 δισεκατομμύρια δολάρια (United States dollar)[1].

1.4 Υπάρχουσες Λύσεις και Εργαλεία Υποστήριξης

Όπως αναφέρθηκε προηγουμένως η αναπτυξη της τεχνολογιας τα τελευταια χρονια ειναι ραγδαια, με μεγάλες εταιρείες όπως η Samsung, η Apple, η Google και η Huawei να επενδύουν πολλα χρηματα και μεγάλο εργατικό δυναμικό για την δημιουργία νέων κινητών λειτουργιών. Το έγγραφο αυτό βασίζεται στην δημιουργία Android εφαρμογής, οπότε δεν θα αναλυθούν τεχνολογίες iOS. Τα τελευταία 15 χρόνια έχουν αναπτυχθεί διάφορες τεχνολογίες που υποστηρίζουν άτομα με οπτικες ή ακουστικές αναπηρίες. Κάποιες από αυτές τις τεχνολογίες έχουν ενσωματωθεί στα λογισμικά των κινητών συσκευών σαν “features”, ενώ άλλες τεχνολογίες έχουν μετατραπεί σε εφαρμογές. Παρακάτω παρατίθενται δύο πίνακες, οι οποίοι απεικονίζουν τεχνολογιες που εχουν ενσωματωθει στο λογισμικο και τις τεχνολογίες που έχουν μετατραπεί σε εφαρμογές (αναφορά στους πίνακες 1.4 & 1.5)[7].

Λειτουργία	Τρόπος Λειτουργίας	Περιγραφή
Google assistant,	Βασισμένη στον ήχο	Κάντε μια ερώτηση. Δώσε εντολές να πραγματοποιήσει.
TalkBack	Βασισμένη στον ήχο	Αναγνώστης οθόνης (screen reader).
Text to speech/Voice recognition	Βασισμένη στον ήχο	Μετατροπή κειμένου σε ομιλία.
Select to speak	Βασισμένη στον ήχο	Ανάγνωση επιλεγμένου κειμένου.

Zoom magnification/Font size	Βασισμένη στο βίντεο	Εργαλείο μεγέθυνσης.
Contrast	Βασισμένη στο βίντεο	Διαφοροποίηση ενός αντικειμένου από το φόντο του.
Voice Inputs Keyboards	Βασισμένη στον ήχο	Πληκτρολόγηση μέσω φωνητικών εντολών.

Πίνακας 1.4: Συχνές Ενσωματωμένες Λειτουργίες

Εφαρμογή	**Τρόπος Λειτουργίας**	**Περιγραφή**
Kibo	Βασισμένη στον ήχο	Ανάγνωση εικόνων, PDF, e-books, doc, αναγνώστης Χίντι και Αγγλικών, καταγραφή φωνής.
Be My Eyes	Βασισμένη στον ήχο	Βοήθεια από άτομα με όραση (εθελοντές) μέσω βιντεοκλήσεων.
Supersense	Βασισμένη στον ήχο	Πληροφορίες για το περιβάλλον, Τεχνητή Νοημοσύνη (AI).
Visor	Βασισμένη στο βίντεο	Μεγεθυντής (για κοντινά αντικείμενα).
Binoculars	Βασισμένη στο βίντεο	Μεγεθυντής σε απόσταση (κάμερα με σούπερ ζουμ).
Google Pay	Βασισμένη στον ήχο	Εύκολος και αξιόπιστος τρόπος για online πληρωμές, ηλεκτρονικές συναλλαγές.
KNFB Reader	Βασισμένη στον ήχο	Εργαλεία μετατροπής κειμένου σε ομιλία, Braille και επισήμανσης κειμένου.
Audible	Βασισμένη στον ήχο	Παροχή ηχητικών βιβλίων.
Color Grab	Βασισμένη στον ήχο	Σημειώστε το κινητό σε οτιδήποτε και λάβετε το ακριβές χρώμα, συμπεριλαμβανομένης της εξαγωγής του δεκαεξαδικού.
Signily Keyboard	βασισμένη στο βίντεο	Οπτικό πληκτρολόγιο που επιτρέπει στους χρήστες να επικοινωνούν χρησιμοποιώντας την Αμερικανική Νοηματική Γλώσσα (ASL)

Google Live Transcribe	Βασισμένη στον ήχο	Παροχή μετατροπής συνομιλιών από ομιλία σε κείμενο, σε πραγματικό χρόνο

Πίνακας 1.5: Δημοφιλείς Εφαρμογές

Η ανάπτυξη του συγκεκριμένου επιστημονικού κλάδου, έχει καθεστίσει απλή και εύκολη την ενσωμάτωση των τεχνολογιών αυτον σε νέες εφαρμογές και πρότζεκτ. Αυτό και η λεπτομερή καταγραφεί (documentation) των νέων τεχνολογιών, αποτέλεσαν τον κύριο παράγοντα για την προσέγγιση νέων προγραμματιστών και επιστημόνων να δημιουργήσουν καινοτόμες εφαρμογες, μερικες απο τις οποιες θα αναλυθουν ακολούθως.

SCTS (Subtitles Course Tracking System): Το SCTS είναι εφαρμογή η οποία έχει σχεδιαστεί για την επικοινωνία μεταξύ ατόμων με προβλήματα ακοης και ατόμων που δεν γνωρίζουν νοηματική. Η εφαρμογή μετατρέπει προφορικές λέξεις σε κείμενο και είναι βασισμένη στην AJAX. Επίσης υποστηρίζει εκτος απο Android και την χρήση της σαν πλατφόρμα σε ιστοσελίδα, εκμεταλευόμενη τις τεχνολογίες που παρεχουν οι PHP, MySQL, AJAX, HTML και η CSS. Αυτό, διότι οι δημιουργοί έχουν σαν στόχο η εφαρμογή/πλατφόρμα να χρησιμοποιείτε στο μελλον απο καθηγητές, οπου με την χρήση της συγκεκριμένης τεχνολογίας θα μπορούν να επικοινωνούν με τους μαθητές, χωρίς να είναι απαραίτητη η γνώση της νοηματικής γλώσσας και χωρις να τίθενται θέματα οικονομικής δυσχέριας για τους μαθητες, οι οποίοι δεν θα χρειαζεται στο μελλον να προμηθευτούν κινητή συσκευή[8].

Ai Εφαρμογή (Universiti Teknikal Malaysia Melaka): Η εφαρμογή με την χρήση CNN (νευρωνικού δικτύου) παρέχει στους χρήστες την δυνατότητα με την κάμερα τους και χωρίς πρόσβαση στο διαδίκτυο, να βγαλουν μια φωτογραφία και να ενημερωθούν για τα αντικείμενα που απεικονίζονται στην φωτογραφία και για τα χρώματα τους. Η εφαρμογή εκμεταλλεύεται έτοιμα (pretrained) μοντέλα τεχνητής νοημοσύνης, τα οποία παρέχει το Tensorflow[9].

Εφαρμογή Αναγνώρισης Χαρακτήρων (OCR): Optical Character Recognition ειναι λειτουργία στα κινητά, που επιτρέπει την αναγνώριση χαρακτήρων με την χρήση καμερας. Η εφαρμογή βελτίωσε την τεχνολογία OCR προσθέτοντας τεχνολογία text to speech (TTS), κανοντας την ετσι προσβάσιμη για άτομα με οπτικές αναπηρίες. Με αυτόν τον τρόπο προσφέρετε στους μια αρκετά οικονομική λύση, με περιορισμό όμως την αναγνώρισή μόνο της Αγγλικής γλώσσας. [10].

MonVoix: Αρχικά, η εφαρμογή MonVoix όπως και η εφαρμογή **SCTS ,**η οποία αναφέρθηκε προηγουμένως, αποσκοπεί να δώσει λύσεις και βοηθητικα εργαλεία, στα προβλήματα εκπαίδευσης που αντιμετωπίζουν άτομα με προβλήματα ακολης. Στην συγκεκριμένη εφαρμογή, οι δημιουργεί ανεπτυξαν ενα AI μοντέλο αναγνώρισης της νοηματικής γλώσσας, χρισιμοποιόντας τεχνικες όπως Otsu, Gaussian και Canny για την επεξεργασία των εικόνων. Για την αναγνώριση της νοηματικής χρησιμοποιήθηκε η βιβλιοθήκη OpenCV, ενώ για την διαχείριση της βάσης δεδομένων χρησιμοποιήθηκε η SQLite [11].

Οι τεχνολογίες αυτές έχουν χρησιμοποιηθεί σαν πηγή έμπνευσης για την δημιουργία της εφαρμογής Aather, αξιοπιοντας υπάρχουσες τεχνολογίες και ενσωματώνοντας καινουργιες. Επίσης στην εφαρμογή χρησιμοποιήθηκαν διαφορετικές τεχνικές από αυτές που αναφέρθηκαν, για την περαιτέρω βελτίωση και ανάπτυξη των τεχνολογιών.

Κεφάλαιο 2ο: Εκπαιδευτική Αναζήτηση και Καθοδηγητικές Γραμμές για την Ανάπτυξη της Εφαρμογής

2.1 Εισαγωγή

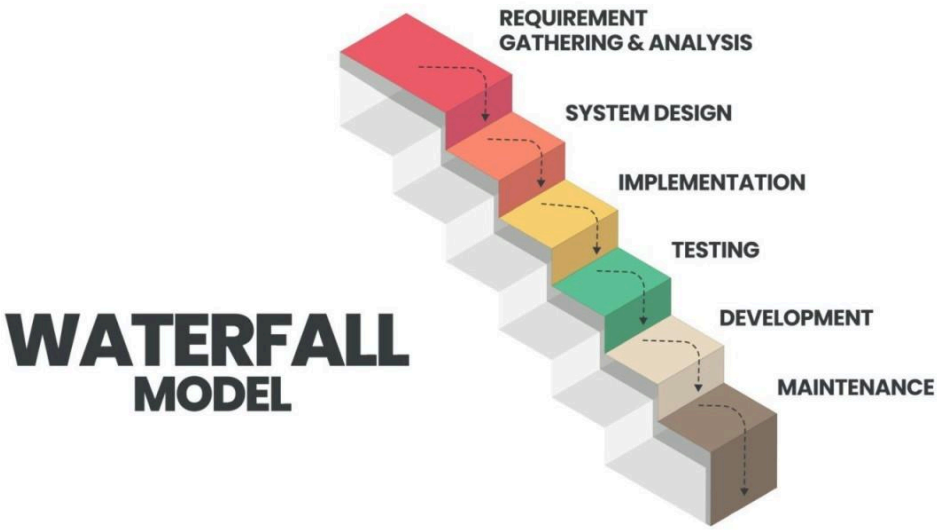
Η ανάπτυξη μιας βοηθητικής εφαρμογής για άτομα με αναπηρίες απαιτεί προσεκτική έρευνα και σχεδιασμό. Πέρα από την τεχνική πλευρά της ανάπτυξης λογισμικού, είναι απαραίτητο να ληφθούν υπόψη οι ανάγκες των χρηστών και να οριστούν στόχοι για το πρότζεκτ. Σε αυτό το κεφάλαιο θα αναλυθεί η μεθοδολογία που επιλέχθηκε για την ανάπτυξη του λογισμικού και οι βασικοί τομείς που πρέπει να μελετηθούν κατά την ανάπτυξη της εφαρμογής, όπως τα κατάλληλα γράμματα, τα χρώματα, οι απτικές διεπαφές, εργαλεία, και η δομή κάθε δραστηριότητας.

2.2 Επιλογή Στρατηγικής Υλοποίησης

Η υλοποίηση της εφαρμογής βασίστηκε στη μεθοδολογία **Waterfall**. Το Waterfall είναι μεθοδολογία γραμμικής ανάπτυξης λογισμικού, που το χαρακτηριστικό των προκαθορισμένων απαιτήσεων την καθιστά κατάλληλη για την ανάπτυξη της εφαρμογής. Το μοντέλο διαχωρίζει ανάπτυξη της εφαρμογής σε στάδια, κάθε ένα από το οποίο πρέπει να ολοκληρωθεί για να ξεκινησει η έναρξη του επόμενου, διασφαλίζοντας έτσι την προσεκτική υλοποίηση κάθε βήματος. Για την περαιτέρω κατανοηση της μεθοδολογιας συμβουλευτείτε στην Εικόνα 2.2

Πλεονεκτήματα του μοντέλου **Waterfall** :

- Ξεκάθαροι στόχοι.
- Σταθερές απαιτήσει.
- Η πρόοδος του συστήματος είναι εύκολα μετρήσιμη.
- Εύκολα κατανοητή ανάπτυξη λογισμικού.
- Παράγει επίσημη τεκμηρίωση προδιαγραφών.
- Καλύτερη κατανομή πόρων.
- Βελτιώνει την ποιότητα. Η έμφαση στις απαιτήσεις και τον σχεδιασμό πριν την έναρξη υλοποίησης του κώδικα εξασφαλίζει ελάχιστη σπατάλη χρόνου και προσπάθειας, μειώνοντας τον κίνδυνο καθυστερήσεων στο χρονοδιάγραμμα.



Εικόνα 2.2: Μεθοδολογία Waterfall

2.2.1 Ανάλυση Απαιτήσεων

Στο πρώτο στάδιο καθορίζεται ποιοι θα ειναι οι στόχοι της εφαρμογής. Σε αυτο το στάδιο είναι πολύ βασικό να δοθεί βαρύτητα στο ερευνητικό κομμάτι, καθώς αποτελεί και την κεντρική ιδεα του προτζεκτ. Με την σωστή έρευνα και τον ορισμό πραγματοποιούμενων στόχων, η δημιουργία της εφαρμογής γίνεται πιο ομαλη.

2.2.2 Σχεδιασμός

Ακολουθώντας, το δεύτερο στάδιο είναι ο σχεδιασμός της εφαρμογής, ο οποίος περιλαμβάνει:

- Τη δημιουργία του λογοτύπου.
- Το σχεδιασμό της διεπαφής χρήστη (UI/UX), βασισμένο στις ανάγκες των χρηστών.
- Τον καθορισμό των βιβλιοθηκών και των εργαλείων που θα χρησιμοποιηθούν για την υλοποίηση της εφαρμογής.

2.2.3 Υλοποίηση

Το στάδιο της υλοποίησης περιλαμβάνει την αρχή της κατασκευής του κώδικα και της ενσωμάτωσης των δύο προηγούμενων σταδίων. Ο κώδικας βασίζεται πάνω στο σχεδιαστικό κομμάτι που υλοποιήθηκε προηγουμένως, χρησιμοποιώντας τις εξαρτήσεις (**dependencies**) και τις βιβλιοθήκες που οργανώθηκαν στο προαναφερθέν σταδιο, για την επίτευξη των στόχων.

2.2.4 Δοκιμές

Μετά την ολοκλήρωση της υλοποίησης, ακολουθεί το στάδιο που πρέπει να πραγματοποιηθούν δοκιμές για να διασφαλιστεί η σωστή λειτουργία της εφαρμογής. Οι δοκιμές περιλάμβαναν:

- Τον έλεγχο λειτουργιών, όπως η καταγραφή και η προβολή δεδομένων.
- Δοκιμές προσβασιμότητας από χρήστες.
- Διορθώσεις σφαλμάτων και βελτιστοποίηση της απόδοσης.

2.2.5 Παράδοση

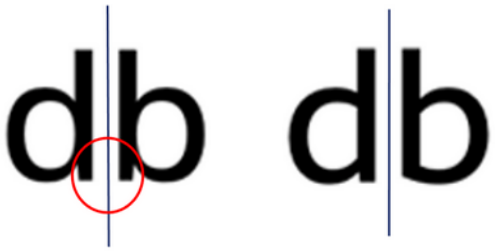
Η τελευταία φάση της διαδικασίας περιλάμβανε την τεκμηρίωση της εφαρμογής και την παρουσίασή της. Το τελικό προϊόν παραδόθηκε πλήρως λειτουργικό, ανταποκρινόμενο στις αρχικές απαιτήσεις και προσφέροντας λύσεις στα προβλήματα που τέθηκαν.

2.3 Επιλογή κατάλληλης Γραμματοσειράς

Η τυπογραφία επηρεάζει σημαντικά την αναγνωσιμότητα μιας εφαρμογης. Λόγο των μικρών οθονών και των χρηστών που θα φιλοξενήσει η εφαρμογή, η σημασία της κατάλληλης επιλογής γραμματοσειράς αποκτά ακόμα μεγαλύτερη σημασία.

Συμφωνα με το μπλογκ [12] τα κύρια σημεία που πρέπει να ελεγχθούν για την επιλογή της σωστης γραμματοσειράς είναι :

- Να έχουν μεγαλύτερο τον άξονα x (να ειναι πλατιά)
- Να υπάρχουν κενα μεταξύ των γραμμάτων
- Να υπάρχουν μεγάλα κενα μεταξύ των λέξεων
- Να είναι απλή
- Να μήν είναι ομοιόμορφη (ξεκάθαρες διαφορές μεταξύ d και b), διαφορά στην Εικόνα 2.3



Εικόνα 2.3: Διαφορά μεταξύ των γραμμάτων d και b

Λόγω όλων των παραπάνω ως βασική γραμματοσειρά επιλέχθηκε η χρήση της **Helvetica**, ενώ σαν δεύτερες και τρίτες γραμματοσειρές για περαιτέρω ευαναγνωσία επιλέχθηκαν η **Arial** και η **Open Sans.** Σαν μέγεθος για τις γραμματοσειρές, επιλέχθηκε να χρησιμοποιηθεί λίγο μεγαλύτερο από τα στάνταρ της βιομηχανίας που ειναι **11px**. Η απεικόνιση των γραμματοσειρών σε μέγεθος **14px**, δικαιολογείται από τις οπτικές αδυναμίες που μπορεί να έχουν αρκετοί χρήστες.

2.4 Χρώματα και Αντίθεση

Η κατανόηση των κατηγοριών αχρωματοψίας είναι σημαντική για την επιλογή χρωμάτων για την εφαρμογή. Πρέπει να χρώματα να μην κουράζουν και να μην δυσκολεύουν τους χρήστες ανεξάρτητα της οπτικής τους αναπηρίας. Για την διαφοροποίηση των αντικειμένων στην οθόνη, βασικό είναι η

χρήση χρωμάτων με μεγάλη αντίθεση και η αποφυγή χρωμάτων που επηρεαζονται απο την αχρωματοψία

Υπάρχουν διάφοροι τύποι αχρωματοψίας αλλα οι πιο σύνηθες είναι η προτανωπία και η δευτερανοπία. Οι δυο αυτες κατηγορίες δυσκολεύουν την αναγνώριση των **πράσινων** και των **κόκκινων** χρωμάτων, οπότε η αποφυγή τους είναι σπάνια χρήση τους είναι σημαντική στην ανάπτυξη του πρότζεκτ. Αντίθετα καλές πρακτικές για την ανάπτυξη παρόμοιων εφαρμογών είναι η χρήση του **μαύρου** και του **λευκού**, λόγω της υψηλής τους αντίθεσης.

Επιπλέον, η προσθήκη σχεδίων (patterns) στα χρώματα μπορεί να βοηθήσει στη διαφοροποίηση των στοιχείων μεταξύ τους. Για παράδειγμα, τα μοτίβα ή οι γραμμές μπορούν να προσθέσουν επιπλέον αντίθεση και να κάνουν τα στοιχεία πιο ευδιάκριτα [13].

Η επιλογή των χρωμάτων έγινε με τη βοήθεια του εργαλείου **Adobe Color**, το οποίο είναι ένα online εργαλείο που υποστηρίζει τη δημιουργία συνδυασμών χρωμάτων συμβατά με τις ανάγκες των χρηστών. Για την εφαρμογή, χρησιμοποιήθηκαν τα χρώματα αναφερόμενα στον Πίνακα 3.4, τα οποία είναι γνωστά ως **flat χρώματα**, λόγω της απλότητας και της αισθητικής τους. Τα χρώματα αυτά επιλέχθηκαν λόγω των συχνοτήτων φωτός που εκπέμπουν, έχοντας ως αποτέλεσμα να μην είναι κουραστικά στα μάτια λόγω υπερβολικά έντονων χρωμάτων.

Χρώμα	**Κωδικός Hex**
Μπλε	#1f7bbd
Μωβ	#d067ca
Πράσινο	#27ae60
Κόκκινο	#ca2a1e
Κίτρινο	#f7db4d
Ροζ	#f7584d
Μαύρο	#000000
Λευκό	#ffffff

Πίνακας 3.4: Χρώματα εφαρμογης Aather (brand colors)

2.5 Ενσωμάτωση Απτικών Διεπαφών

Απτικές επαφές ή αλλιως **haptics** είναι τεχνολογίες που χρησιμοποιούν την αίσθηση της αφής για την αλληλεπίδραση με συσκευές. Μέσω δονήσεων, πίεσης ή άλλων φυσικών αισθήσεων, επιτρέπουν στους χρήστες να "αισθάνονται" δεδομένα ή λειτουργίες. Οι συγκεκριμένες τεχνολογίες βοηθούν σημαντικα την επικοινωνία μεταξύ χρηστών με αισθησιακές αναπηρίες και μηχανών.

Η ενσωμάτωση απτικών διεπαφών σε συσκευές, όπως κινητά τηλέφωνα ή φορητές συσκευές, επιτρέπει την παροχή πληροφοριών μέσω διαφορετικών μοτίβων δόνησης, που οι χρήστες μπορούν να

αναγνωρίσουν και να ερμηνεύσουν για να κατανοήσουν το περιβάλλον τους. Για παράδειγμα, μια συγκεκριμένη δόνηση μπορεί να σηματοδοτήσει την ενεργοποίηση ενός κουμπιού.

2.6 Εργαλεία που Πρέπει να Ενσωματωθούν

Για την ανάπτυξη εφαρμογών οπως το **Aather**, είναι απαραίτητη η χρήση βασικών τεχνολογιών υποστήριξης ατόμων με οπτικές η ακουστικές αναπηρίες. Οι τεχνολογίες αυτες παρουσιάζεται παρακάτω:

- **Οθόνη Ανάγνωσης (Screen Reader)**: Είναι το πιο δημοφιλές εργαλείο για άτομα με προβλήματα όρασης. Το λογισμικό διαβάζει το περιεχόμενο μιας οθόνης και το μετατρέπει σε σε ομιλία.
- **Μεγέθυνση Οθόνης (Screen Magnification)**: Για άτομα με μειωμένη όραση, εργαλεία μεγέθυνσης εικόνας και κειμένου.
- **Ακουστικά Εργαλεία (Speech-to-Text, Text-to-Speech)**: Τα εργαλεία φωνητικής αναγνώρισης σε Android επιτρέπουν στους χρήστες να αλληλεπιδρούν με συσκευές μέσω της ομιλίας τους.
- **Εφαρμογές Υποστήριξης Ομιλίας**: Εφαρμογές που επιτρέπουν στους χρήστες με δυσκολίες στην ομιλία να επικοινωνούν μέσω εικονιδίων και φωνητικής ανατροφοδότησης

Πολλα απο τα παραπανω εργαλεια έχουν εφαρμοστεί στην εφαρμογη και θα αναλυθούν λεπτομερως στην συνεχεια.

Κεφάλαιο 3ο: Αρχικά Στάδια Υλοποίησης της Εφαρμογής

3.1 Εισαγωγή

Στο κεφάλαιο αυτό, περιγράφονται τα βασικά στάδια της υλοποίησης του πρότζεκτ, ξεκινώντας από τη σχεδίαση της οπτικής ταυτότητας του έργου, όπως το λογότυπο και στη συνέχεια παρουσιάζοντας τη διαδικασία ανάπτυξης της εφαρμογής .

Συγκεκριμένα, εξετάζεται η σημασία του σωστού σχεδιασμού, οι τεχνολογίες που επιλέχθηκαν, οι στρατηγικές ανάπτυξης και η διαχείριση των εξαρτήσεων (dependencies). Παράλληλα, δίνεται έμφαση στη δομή του Firebase για την υποστήριξη της εφαρμογής, στους ρόλους των χρηστών και στις απαιτήσεις των permissions που απαιτούνται για τη σωστή λειτουργία.

3.2 Σχεδιασμός Λογότυπου

Η εμπνευση του σχεδιασμού του λογοτύπου βασίστηκε στο όνομα της εφαρμογής, **Aather**, το οποίο προέρχεται από την αρχαία ελληνική μυθολογία. Ο Αιθέρας (Aether) ήταν το παιδί της Νύχτας (Nyx) και του Ερέβους (Erebus), συσχετισμένος με το αιώνιο φως και τον καθαρό ουρανό. Το όνομα επιλέχθηκε για να αποτυπώσει την έννοια της ελπίδας, της καθαρότητας και της καθοδήγησης, καθως ειναι οι εννοιες με τις οποιες συνδεόταν στην αρχαιότητα ο θεος **Aather**.

3.2.1 Φιλοσοφία Σχεδιασμού

Παρακάτω γίνεται αναφορά στην Εικόνα 3.3

Το λογότυπο σχεδιάστηκε λαμβάνοντας υπόψη την ανάγκη ενός εύκολα αναγνωρίσιμου συμβόλου, το οποίο πρέπει ταυτόχρονα να είναι και συμβολικά ισχυρό. Η χρήση **μαύρου φόντου** και **λευκού φωτεινού σχεδίου** αναδεικνύει τη συμβολική μετάβαση από το σκοτάδι στο φως, παραπέμποντας στη φιλοσοφία της εφαρμογής: να φέρει φως και καθοδήγηση σε όσους έχουν ανάγκη. Το λογότυπο, πέρα από την αισθητική του αξία, αποτελεί ένα ισχυρό μήνυμα του στόχου της εφαρμογής: τη βελτίωση της ποιότητας ζωής για άτομα με οπτικές ή ακουστικές αναπηρίες.

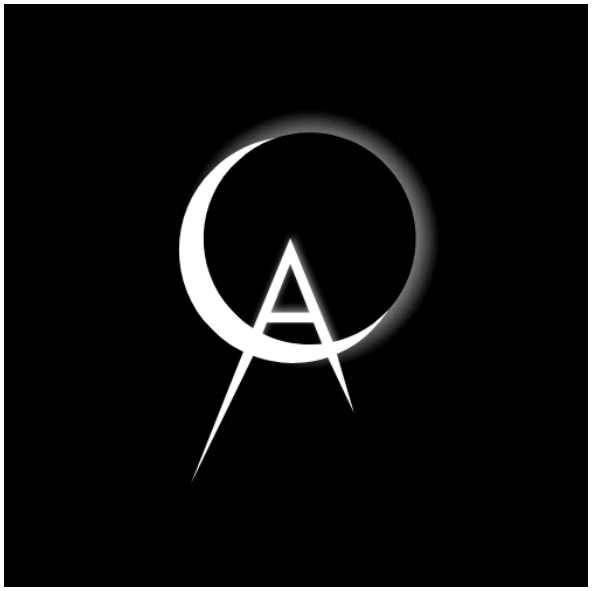
3.2.2 Στοιχεία του Λογότυπου

- **Ο Κύκλος**: Ο κύκλος στο λογότυπο αντιπροσωπεύει τη νύχτα (την μητέρα του Aather), το φεγγαρι και ο ήλιος βρισκονται απο πίσω της, ενώ η λευκή λάμψη γύρο του υποδηλώνει το φως του Αιθέρα που ξεπροβάλλει από το σκοτάδι.
- **Το γράμμα "Α"**: Το γράμμα "Α" τοποθετήθηκε κεντρικά, ώστε να παραπέμπει άμεσα στο όνομα της εφαρμογής.
- **Η Αντίθεση Μαύρου και Λευκού**: Αυτή η επιλογή χρωμάτων εξασφαλίζει υψηλή αντίθεση, καθιστώντας το λογότυπο εύκολα αναγνωρίσιμο, ακόμα και από άτομα με μειωμένη όραση.
- **Απλότητα Σχεδίασης**: Η λιτότητα στις γραμμές και τα σχήματα του λογότυπου αποτρέπει την οπτική σύγχυση.
- **Συμβολισμός**: Το λογότυπο αντανακλά την ελπίδα για ένα καλύτερο αύριο, αποπνέοντας θετικά συναισθήματα, εμπιστοσύνη και γαλήνη.

3.2.3 Τεχνική Υλοποίηση

Το λογότυπο δημιουργήθηκε χρησιμοποιώντας δύο κύρια εργαλεία:

1. **Canva**: Για την αρχική σύλληψη και απεικόνιση της ιδέας.
2. **Adobe Illustrator**: Για την τελική επεξεργασία και τον καθαρισμό των γραμμών, εξασφαλίζοντας υψηλή ποιότητα και επαγγελματικότητα.



Εικόνα 3.3: Λογότυπο εφαρμογης Aather

3.3 Υλοποίηση του UI και του UX

Η υλοποίηση του UI (User Interface) και του UX (User Experience) στην εφαρμογή αποτελεί ένα από τα πιο σημαντικά βήματα για την εξασφάλιση μιας ευχάριστης και εύκολης πλοήγησης για τον χρήστη. Στο πλαίσιο της ανάπτυξης της εφαρμογής, δόθηκε ιδιαίτερη έμφαση στη σωστή επιλογή χρωμάτων (αναφορά στο κεφάλαιο 2.4), στον σχεδιασμό των κουμπιών, και στην υλοποίηση μηχανισμών πλοήγησης που να διευκολύνουν τη χρήση για όλους τους χρήστες, συμπεριλαμβανομένων αυτών με οπτικές δυσκολίες.

3.3.1 Διαφοροποίηση Κουμπιών

Για τη οπτική διευκόλυνση, στα κουμπιά της εφαρμογής προστέθηκαν:

- **Διαφορετικά σχήματα** (π.χ., στρογγυλά, τετράγωνα, ορθογώνια) για τον διαχωρισμό των λειτουργιών.
- **Οπτικοί δείκτες (indicators)**, οι οποίοι ενισχύουν την εμπειρία πλοήγησης, ειδικά σε περιπτώσεις όπου το χρώμα δεν επαρκεί ως διακριτικό στοιχείο.

Οι διαφοροποιήσεις αυτές προσφέρουν καλύτερη κατανόηση για το ποια κουμπιά εξυπηρετούν συγκεκριμένες λειτουργίες, μειώνοντας την πιθανότητα σύγχυσης για τον χρήστη. Αξιοσημείωτο είναι επίσης το μέγεθος και η τοποθεσία των κουμπιών. Η τοποθεσία τους παραμένη συγκεκριμένη στις περισσότερες περιπτώσεις σε όλη την εφαρμογή, ενώ το μεγεθος τους ειναι αρκετα μεγαλο για τις οπτικές ανάγκες των χρηστών.

3.3.2 Πλοήγηση

Για την πλοήγηση στην εφαρμογή χρησιμοποιήθηκε ο **ViewPager2**, ο οποίος δίνει τη δυνατότητα στον χρήστη να μεταβαίνει από τη μία ενότητα της εφαρμογής στην άλλη απλά με μία κίνηση **slide**. Για καλύτερη κατανόηση μπορείτε να κοιτάξετε στην Εικόνα 3.4. Αυτή η μέθοδος πλοήγησης είναι ιδανική, καθώς:

- Ελαχιστοποιεί την ανάγκη χρήσης μικρών κουμπιών, τα οποία μπορεί να είναι δύσκολα εντοπίσημα απο χρήστες με προβλήματα όρασης.
- Παρέχει φυσική και αισθητική εμπειρία χρήσης, κάνοντας την εφαρμογή ομαλή και σύγχρονη.

3.3.3 Εισαγωγική Οθόνη και Ροή Ταυτοποίησης Χρήστη

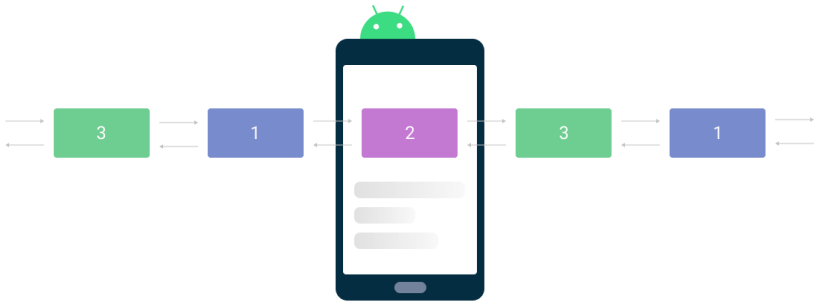
Στην αρχική οθόνη της εφαρμογής, ο χρήστης έχει την επιλογή είτε να **συνδεθεί (Log In)** είτε να **δημιουργήσει νέο λογαριασμό (Sign Up)**. Ανάλογα με την επιλογή του, ενεργοποιείται ένα διαφορετικό animation, προσφέροντας μια οπτικά ευχάριστη εμπειρία.

Εάν ο χρήστης επιλέξει να δημιουργήσει νέο λογαριασμό, καλείται να επιλέξει τον **ρόλο** του. Οι διαθέσιμες επιλογές ρόλων, που σχεδιάστηκαν για να καλύπτουν διαφορετικές ανάγκες, είναι οι εξής:

- **Εθελοντής (Volunteer):** Χρήστες που επιθυμούν να υποστηρίξουν τα άτομα με ειδικές ανάγκες, παρέχοντας βοήθεια μέσω της εφαρμογής.
- **Άτομο με Ακουστική Αναπηρία (Hearing Impairment):** Χρήστες που αντιμετωπίζουν προβλήματα ακοής και χρειάζονται λειτουργίες που διευκολύνουν την επικοινωνία και την πρόσβασή τους σε πληροφορίες.
- **Άτομο με Οπτική Αναπηρία (Visual Impairment):** Χρήστες με δυσκολίες όρασης, οι οποίοι επωφελούνται κυρίως σε θέματα πρόσβασης πληροφορίας.

Η επιλογή του ρόλου γίνεται μέσα από ένα **Activity** με τρία ξεκάθαρα κουμπιά, κάθε ένα από τα οποία οδηγεί σε διαφορετική εμπειρία χρήστη. Αφού ο χρήστης επιλέξει τον ρόλο του, εμφανίζεται το κυρίως μενού της εφαρμογής, το οποίο αποτελείται από **ViewPager2 Fragments**. Τα fragments αυτά είναι προσαρμοσμένα στις ανάγκες του εκάστοτε ρόλου και περιλαμβάνουν διαφορετικές λειτουργίες. Οι λειτουργίες αυτες και τα Activities που τα χαρακτηρίζουν μαζι με τον σκοπός τους θα αναλυθούν λεπτομερώς στο επόμενο κεφάλαιο (κεφάλαιο 4)

```
            when (userUsage) {
                "eyes" -> {
                    lol = 1
                    Toast.makeText(holder.itemView.context, text: "usage = eyes", Toast.LENGTH_SHORT).show()
                }
                "volunteer" -> {
                    lol = 2
                    Toast.makeText(holder.itemView.context, text: "usage = volunteer", Toast.LENGTH_SHORT).show()
                }
                else -> {
                    lol = 3
                    Toast.makeText(holder.itemView.context, text: "usage = ears", Toast.LENGTH_SHORT).show()
                }
            }
        }
    }
}

Polihronis Varvaris *
inner class Pager2ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
    val itemTitle: TextView = itemView.findViewById(R.id.tvTitle)
    val imagePop: ImageView = itemView.findViewById(R.id.imagepop)

    Polihronis Varvaris *
    init {
        imagePop.setOnClickListener { v: View ->
            val position: Int = adapterPosition
            val context = itemView.context

            val intent: Intent = when (lol) {
                1 -> handleEyesClick(position, context)
                2 -> handleVolunteerClick(position, context)
                3 -> handleEarsClick(position, context)
                else -> handleEarsClick(position, context)
            }

            // Start the chosen activity
            context.startActivity(intent)
        }
    }
```



Εικόνα 3.4: Menu πλοήγησης εφαρμογης με slide και μέρος υλοποίησης

3.3.4 Δομή UI/UX για τα Activities Κάθε Ρόλου

Η σχεδίαση του UI (User Interface) και του UX (User Experience) για τα **Activities** κάθε ρόλου έγινε με έμφαση στον οπτικό τους διαχωρισμό, απεικονίζοντας ταυτόχρονα τις λειτουργίες που παρέχει καθε **Activity**. Παρακάτω παρουσιάζεται η δομή των **Activities** που έχουν σχεδιαστεί για κάθε ρόλο, περιγράφοντας τις δυνατότητες και τα χαρακτηριστικά τους.

Άτομα με Δυσκολία στην Όραση:

- **Scale**
  Εργαλείο μεγέθυνσης εικόνων μέσω της κάμερας, επιτρέποντας στους χρήστες να βλέπουν λεπτομέρειες αντικειμένων με μεγαλύτερη ευκρίνεια. Αναφορά στην Εικόνα 3.12
- **Color Picker**
  Δυνατότητα ανίχνευσης και ανάλυσης χρωμάτων από την κάμερα. Περιλαμβάνει λειτουργία μετατροπής χρωμάτων για προσομοίωση γνωστών διαταραχών αντίληψης του χρώματος, όπως η αχρωματοψία. Αναφορά στην Εικόνα 3.13
- **Be My Eyes**
  Περιγραφή αντικειμένων μέσω της κάμερας με τη χρήση τεχνολογίας επεξεργασίας εικόνας, προσφέροντας άμεση υποστήριξη στον χρήστη. Αναφορά στην Εικόνα 3.14
- **Connect Eyes**
  Διασύνδεση του χρήστη με εθελοντές για βοήθεια σε πραγματικό χρόνο, επιτρέποντας την αλληλεπίδραση μέσω βίντεο και ήχου. Αναφορά στην Εικόνα 3.8

Άτομα με Δυσκολία στην Ακοή

- **Edu Sign**
  Ανίχνευση χειρονομιών σε πραγματικό χρόνο, μέσω της κάμερας, για εκμάθηση της νοηματικής γλώσσας. Αναφορά στην Εικόνα 3.9

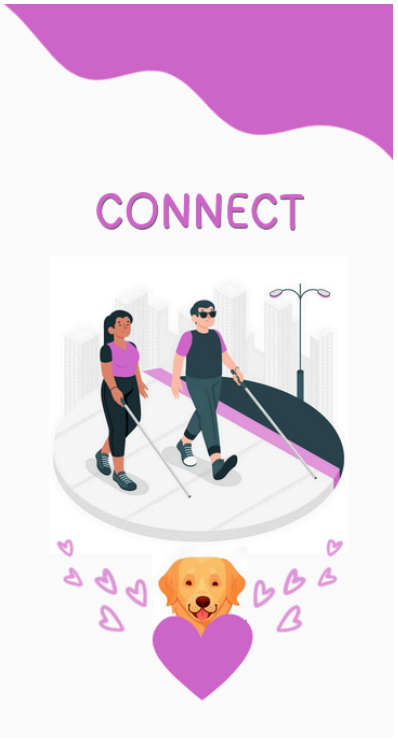
- **Edu Recorder**
  Λειτουργία μετατροπής ομιλίας σε κείμενο, σχεδιασμένη για να βοηθά τους χρήστες να καταγράφουν συνομιλίες ή σημειώσεις σε πραγματικό χρόνο. Αναφορά στην Εικόνα 3.10
- **Noise Meter**
  Εργαλείο μέτρησης θορύβου από το περιβάλλον, που ειδοποιεί τον χρήστη με την βοηθεια της δόνησης. Αναφορά στην Εικόνα 3.11
- **Connect Ears**
  Λειτουργία σύνδεσης του χρήστη με εθελοντές, παρέχοντας βοήθεια σε πραγματικό χρόνο μέσω μηνυμάτων ή βίντεο, με δυνατότητα μετατροπής ομιλίας σε κείμενο. Αναφορά στην Εικόνα 3.6

Εθελοντές

- **Edu Sign**
  Λειτουργία εκμάθησης νοηματικής γλώσσας για εθελοντές, με στόχο τη διευκόλυνση της επικοινωνίας με άτομα με δυσκολία στην ακοή. Αναφορά στην Εικόνα 3.9
- **Color Picker**
  Δυνατότητα ανίχνευσης και ανάλυσης χρωμάτων από την κάμερα. Περιλαμβάνει λειτουργία μετατροπής χρωμάτων για προσομοίωση γνωστών διαταραχών αντίληψης του χρώματος, το οποίο μπορεί να χρησιμοποιηθεί για να κατανοηθούν οι δυσκολίες ατόμων με τις συγκεκριμένες διαταραχές. Αναφορά στην Εικόνα 3.13
- **Settings**
  Επιλογή ρυθμίσεων για τις διεπαφές χρήστη εφαρμογής. Αναφορά στην Εικόνα 3.17
- **Connect Hands**
  Εργαλείο που επιτρέπει στους εθελοντές να συνδέονται με χρήστες που χρειάζονται βοήθεια σε πραγματικό χρόνο, προσφέροντας υποστήριξη μέσω βίντεο και συνομιλίας. Αναφορά στην Εικόνα 3.6
- **Manual**
  Επιτρέπει στους εθελοντές επιλέξουν άτομα που θα βοηθήσουν και τους παρέχει βοηθητικό υλικό για την καλύτερη επικοινωνία μαζί τους. Αναφορά στην Εικόνα 3.15

Εικόνα 3.6 Connect Ears

Εικόνα 3.7 Connect Hands



Εικόνα 3.8 Connect Eyes

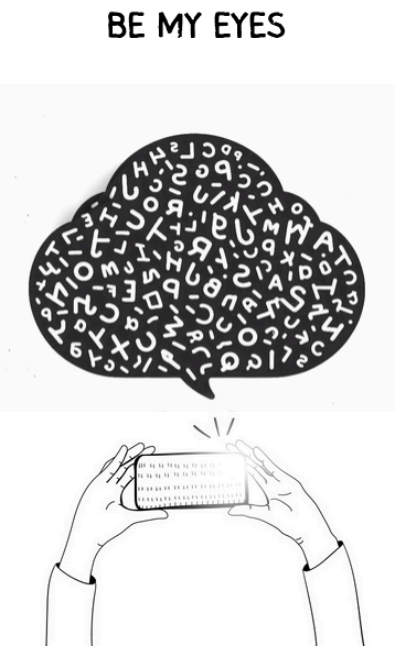
Εικόνα 3.9 Edu Sign

Εικόνα 3.10 Edu Recorder

Εικόνα 3.11 Noise Meter

Εικόνα 3.12 Scale

Εικόνα 3.13 Color Picker



Εικόνα 3.14 Be My Eyes

Εικόνα 3.15 Manual

Εικόνα 3.16 Settings

Κεφάλαιο 4ο: Ανάλυση Εξαρτήσεων, Modules και Δικαιωμάτων

Σε αυτό το κεφάλαιο, θα πραγματοποιηθεί αναλυτική παρουσίαση των εξαρτήσεων (**dependencies**), των **modules** και του κώδικα που αναπτύχθηκε για τη δημιουργία της εφαρμογής. Στόχος είναι να κατανοηθεί η δομή και η λειτουργία τους και να εξηγηθεί η επιλογή των εργαλείων και τεχνολογιών που χρησιμοποιήθηκαν.Επισης θα αναφερθούν τα frameworks που χρησιμοποιήθηκαν και ο σκοπός τους.

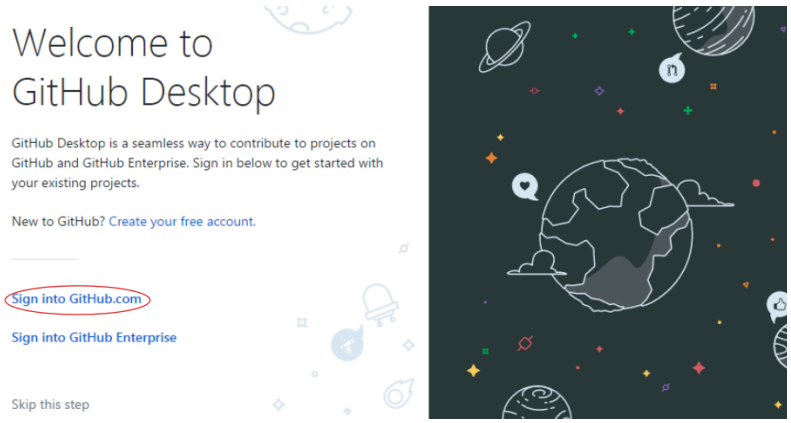
4.1 Επιλογή frameworks και Τεχνολογιών Ανάπτυξης

Όσον αφορά τα frameworks, χρησιμοποιήθηκε το **Android Studio** και το **Visual Studio Code** για την ανάπτυξη της εφαρμογής, ενώ το **GitHub** χρησιμοποιήθηκε για την αποθήκευση και τη διαχείριση του κώδικα. Το Android Studio είναι το επίσημο IDE για την ανάπτυξη εφαρμογών **Android** και προσφέρει ένα πλούσιο περιβάλλον με εργαλεία για τη δημιουργία, τη δοκιμή και τη ανάπτυξη εφαρμογών. Παράλληλα, χρησιμοποιήθηκε το Visual Studio Code για την επεξεργασία κώδικα και τη διαχείριση έργων σε γλώσσες όπως η **Python**. Συγκεκριμένα στην εφαρμογή αναπτύχθηκαν μοντέλα AI (τεχνητής νοημοσύνης) τα οποία είναι βασισμένα στην Python και θα αναλυθούν στο κεφάλαιο 5.

4.1.1 Σύνδεση Github με Android Studio

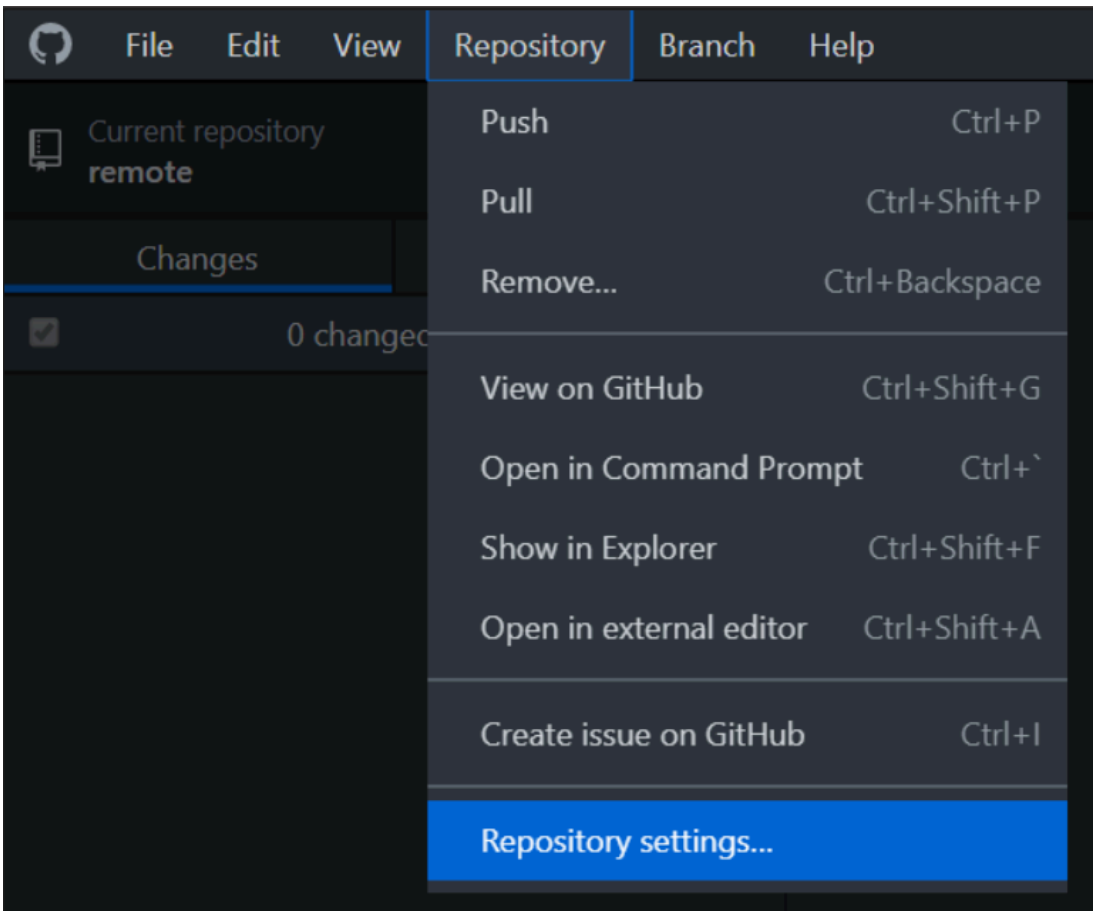
Η ενσωμάτωση του Git με το GitHub και το Android Studio μπορεί να βελτιώσει σημαντικά τη ροή ανάπτυξης εργασιών, παρέχοντας έλεγχο στις εκδόσεις του κώδικα. Παρουσιάζεται ο οδηγός καθοδηγήσεις στη διαδικασία σύνδεσης του Git, του GitHub και του Android Studio. Για τη κατανοητή ροή των βημάτων γίνεται αναφορά στην Εικόνα 4.1 με ολα τα βηματα στην σειρα.

1. Το GitHub Desktop είναι διαθέσιμο για την πλατφόρμα Windows το εδώ.
2. Εγκαταστήστε το GitHub Desktop τρέχοντας το πρόγραμμα εγκατάστασης και περιμένετε να ολοκληρωθεί η εγκατάσταση .
3. Κάντε κλικ στην επιλογή Είσοδος στο Github.com στην οθόνη καλωσορίσματος και συνδεθείτε με τον λογαριασμό σας στο GitHub. (Εικόνα 4.1)



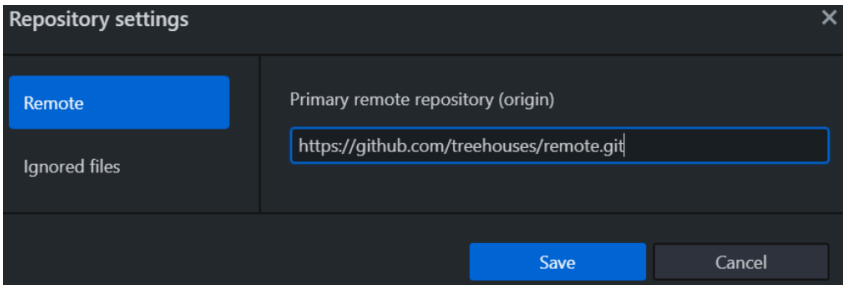
Εικόνα 4.1 Σύνδεση στο Github

4. Κάντε κλικ στο Repository στο μενού και κάντε κλικ στην επιλογή Repository Settings. (Εικόνα 4.2)



Εικόνα 4.2 Ρυθμίσεις Repository

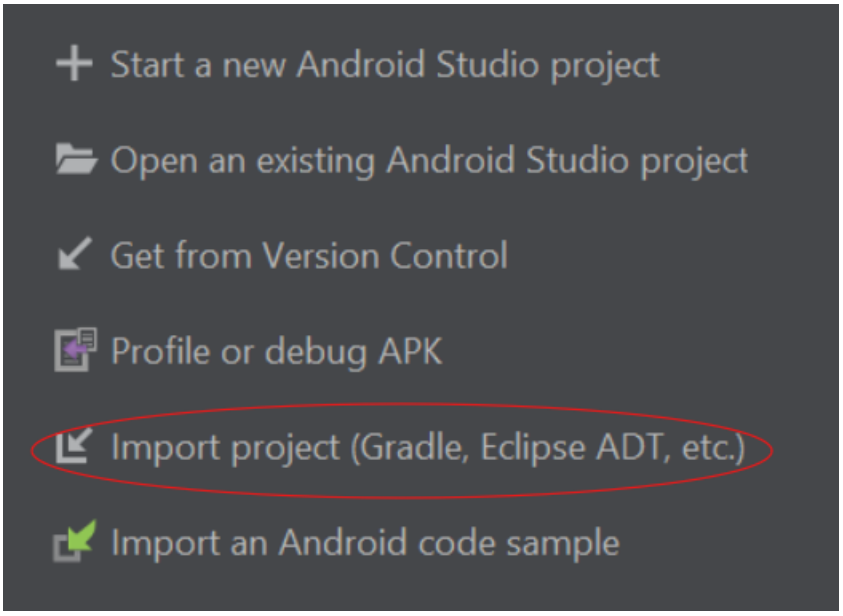
5. Επιλέξτε την επιλογή Remote και εισάγεται στο πεδίο το URL του Repository. Π.χ αν το ονομα σας ειναι treehouses και το Repository έχει όνομα remote το URL θα είναι έτσι: **https://github.com/treehouses/remote.git** στην περίπτωση της διπλωματικής ειναι το: **https://github.com/PolihronisVarvaris/aatherGit**. (Εικόνα 4.3)



Εικόνα 4.3 URL Repository

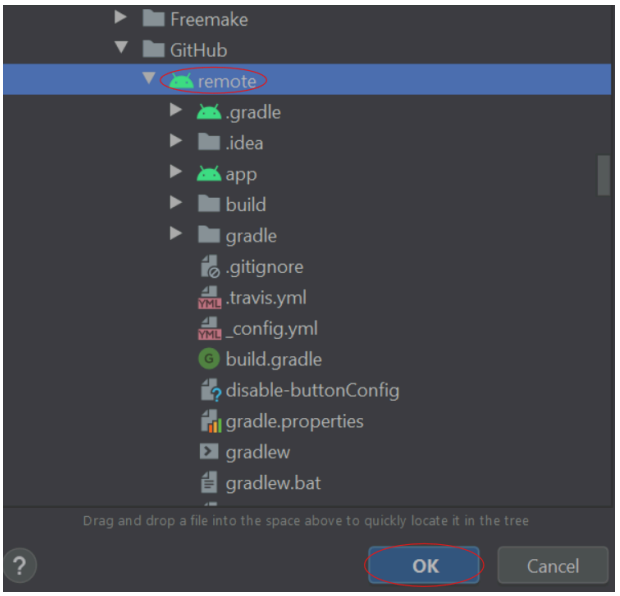
6. Ανοίξτε το Android Studio.

7. Κάντε κλικ στο Import Project (Gradle, Eclipse ADT, etc.). (Εικόνα 4.4)



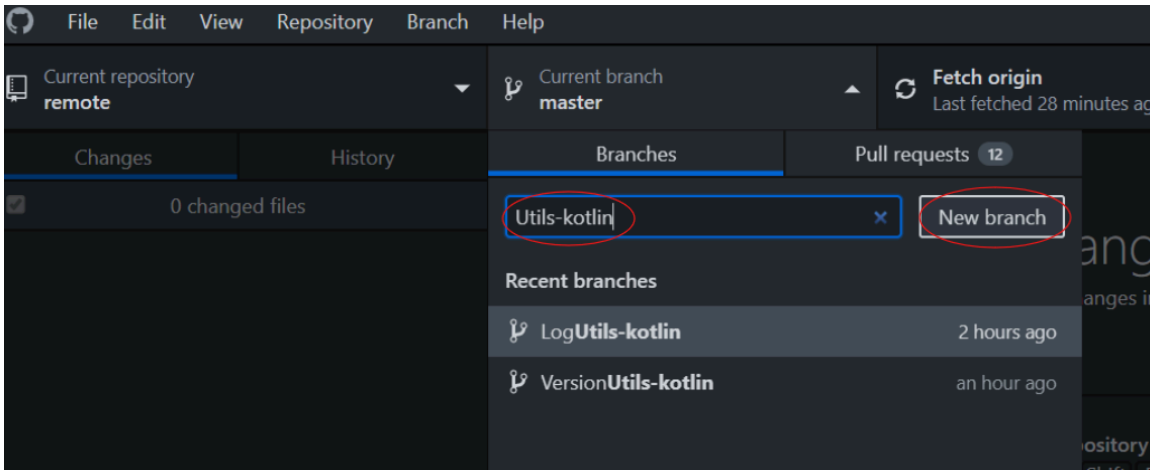
Εικόνα 4.4 Import Project

8. Μεταβείτε στο σημείο όπου το GitHub Desktop αποθήκευσε το Remote Repository τοπικά. Είναι προεπιλογένο να βρίσκεται στον κατάλογο GitHub\remote μέσα στο φάκελο Documents. Επιλέξτε τον απομακρυσμένο φάκελο με το εικονίδιο Android και κάντε κλικ στο OK. (Εικόνα 4.5)



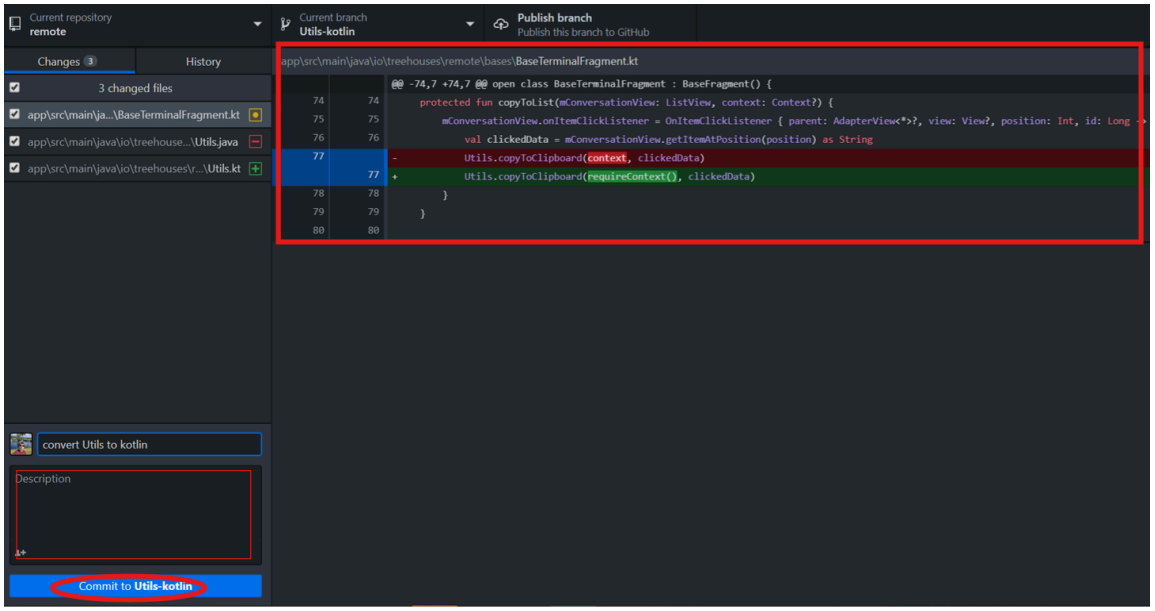
Εικόνα 4.5 Local Remote Repository Path

9. Έπειτα θα βρίσκεστε στον κλάδο Master, θα πρέπει να δημιουργήσετε έναν νέο κλάδο για να κάνετε τις αλλαγές σας. Δώστε του ένα όνομα και κάντε κλικ στο New Branch. Π.χ Utils-kotlin. (Εικόνα 4.6)



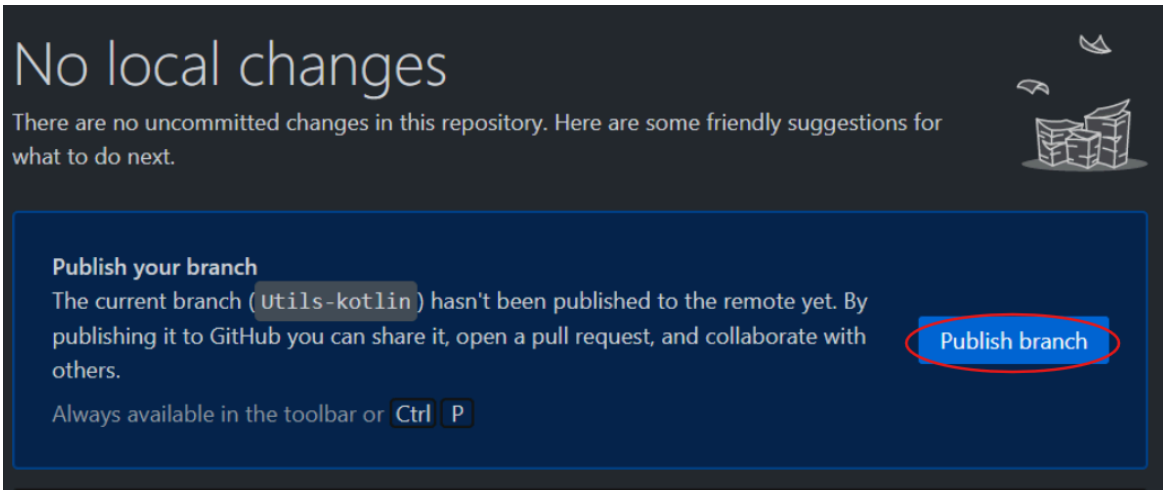
Εικόνα 4.6 Δημιουργία νέου κλάδου

10. Το έργο στο Android Studio θα αλλάξει αυτόματα σε αυτόν του νέο κλάδο. Κάντε τις αλλαγές που επιθυμείτε και όταν επιστρέψετε στο git θα εντοπιστούν αυτόματα. Δώστε μια περιγραφή της αλλαγής σας και κάντε κλικ στο Commit to "το όνομα του repository". (Εικόνα 4.7)



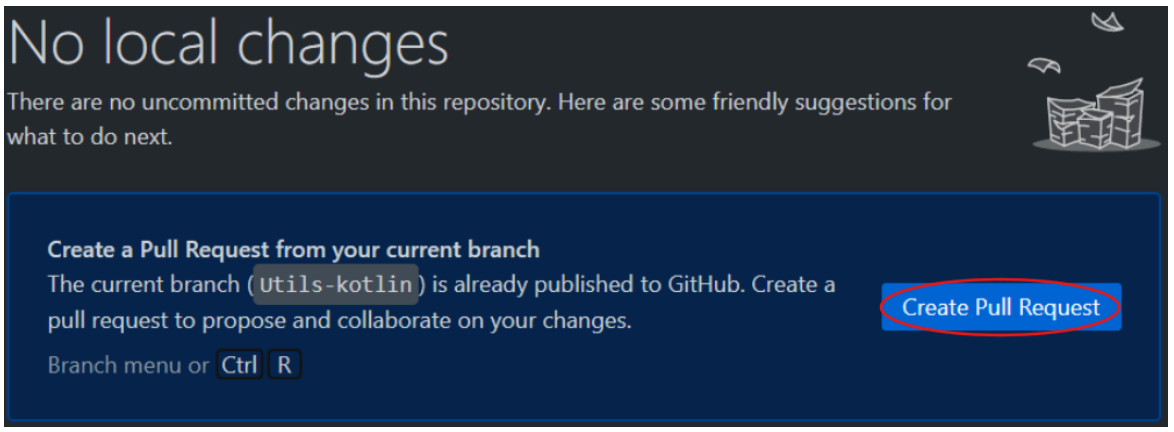
Εικόνα 4.7 Αποθήκευση του ανανεωμένου κώδικα

11. Κάντε κλικ στο Publish Branch για την δημοσίευση του πρότζεκτ. (Εικόνα 4.8)



Εικόνα 4.8 Δημοσίευση του πρότζεκτ

12. Τέλος κάντε κλικ στο Create Pull Request για να ολοκληρώσετε την διαδικασία σύνδεσης. (Εικόνα 4.9)



Εικόνα 4.9 Ολοκλήρωση διαδικασίας

4.2 Plugins

Παρακάτω γίνεται αναφορά στην Εικόνα 4.10.

Τα plugins είναι κομμάτια λογισμικού που παρέχουν πρόσθετες λειτουργικότητες στο σύστημα κατασκευής. Λόγω των plugins δεν χρειάζεται η ανάπτυξη κάθε εργαλείου από την αρχή και προσφέρουν την δυνατότητα να αναπτυχθεί η εφαρμογή σε πιο μοντέρνες γλώσσες προγραμματισμού, όπως η Kotlin.

Η **Kotlin** είναι μια ανοιχτού κώδικα, στατικά τυποποιημένη γλώσσα προγραμματισμού που υποστηρίζει τόσο αντικειμενοστραφή όσο και λειτουργικό προγραμματισμό. Η Kotlin βασίζεται σε παρόμοια σύνταξη και έννοιες με άλλες γλώσσες, όπως η **C#**, η **Java** και η **Scala**, μεταξύ άλλων [14].

Οι κύριοι λόγοι επιλογής της ειναι η υψηλή συμβατότητα που εχει με την java, η παροχή άμεσης υποστήριξης από την Google και η μεγάλη κοινότητα προγραμματιστών που τη χρησιμοποιούν ενεργά.

Plugins που Χρησιμοποιήθηκαν στο Έργο:

- **com.android.application**: Plugin για την ανάπτυξη εφαρμογών Android, υπεύθυνο για τη δημιουργία APK ή AAB (γρήγορη εγκατάσταση και εγκατάσταση με χρήση Google play αντίστηχα).
- **com.android.library**: Χρησιμοποιείται για την ανάπτυξη βιβλιοθηκών Android.
- **org.jetbrains.kotlin.android**: Plugin για την υποστήριξη της γλώσσας Kotlin.
- **com.google.gms.google-services**: Ενσωματώνει τις υπηρεσίες Google (όπως το Firebase) για λειτουργίες όπως αποθήκευση και ειδοποιήσεις push.
- **com.google.dagger.hilt.android**: Plugin για την ενσωμάτωση του Hilt, ενός framework για τη διαχείριση εξαρτήσεων που κατασκευάστηκε πάνω στο **Dagger**.
- **androidx.navigation.safeargs**: Επιτρέπει ασφαλή πέρασμα δεδομένων μεταξύ των fragment.

```
plugins { this: PluginDependenciesSpecScope
    id("com.android.application") version "8.2.2" apply false
    id("com.android.library") version "8.2.2" apply false
    id("org.jetbrains.kotlin.android") version "1.8.0" apply false
    id("com.google.gms.google-services") version "4.3.15" apply false
    id("com.google.dagger.hilt.android") version "2.44" apply false
    id("androidx.navigation.safeargs") version "2.6.0" apply false
}
```

Εικόνα 4.10 Plugins του Aather

4.3 Ρυθμίσεις και Διαμόρφωση του Android Project

Σε αυτό το υποκεφάλαιο παρουσιάζονται οι βασικές ρυθμίσεις του αρχείου **build.gradle**. που ορίζουν τη δομή, τη λειτουργικότητα, και τις εξαρτήσεις του Android project. Οι ρυθμίσεις αυτές είναι κρίσιμες για τη σωστή κατασκευή και εκτέλεση της εφαρμογής. Οι παρακατω υποενότητες αναφέρονται στην Εικόνα 4.11.

Το **Gradle** είναι ένα εργαλείο αυτοματοποιημένης κατασκευής (build automation tool) που περιέχει πληροφορίες και οδηγίες για τη διαδικασία κατασκευής (build) της εφαρμογής. Υπάρχουν δύο διαφορετικοί φάκελοι Gradle. Το **build.gradle (:Aather)** που βρίσκεται στο επίπεδο project, έχει της ρυθμίσεις όλου του πρότζεκτ, όπως τα **Plugins**, τις εξαρτήσεις (**dependencies**) και τις εκδόσεις που χρησιμοποιουνται. Αντίθετα, το **build.gradle (:app)** βρίσκεται μέσα στο φάκελο του κάθε module και περιλαμβάνει πληροφορίες για το module και την κατασκευή του.

4.3.1 Namespace και Εκδόσεις SDK

Η εφαρμογή σαν namespace έχει δηλομένο: **com.example.aather**, το οποίο λειτουργεί ως μοναδικό αναγνωριστικό όνομα για την εφαρμογή, αποκτώντας έτσι μια διαχωριστικη ταυτοτητα απο τις υπολοιπες Android εφαρμογές. Η παράμετρος **compileSdk** = 35 ορίζει την έκδοση του Android SDK

που χρησιμοποιείται κατά τη διαδικασία της μεταγλώττισης, ενώ οι παράμετροι **minSdk** = 30 και **targetSdk** = 34 καθορίζουν την ελάχιστη και την προτεινόμενη έκδοση SDK για τη κινητή συσκευή.

4.3.2 Ρυθμίσεις

Το block **sourceSets** καθορίζει τη δομή του project και τις τοποθεσίες των βασικών αρχείων:

- **jniLibs.srcDirs("src/main/jniLibs")**: Κατάλογος για τις βιβλιοθήκες C/C++ (JNI).
- **java.srcDirs("src/main/kotlin")**: Κατάλογος για αρχεία Kotlin.
- **res.srcDirs("src/main/res")**: Κατάλογος για τους πόρους της εφαρμογής (layouts και drawables).
- **manifest.srcFile("src/main/AndroidManifest.xml")**: Τοποθεσία αρχίου manifest της εφαρμογής.

4.3.3 Default Config

Το block **defaultConfig** περιέχει τις βασικές ρυθμίσεις της εφαρμογής:

- **applicationId**: Ο μοναδικός αναγνωριστικός κωδικός της εφαρμογής.
- **versionCode** και **versionName**: Ορίζουν την έκδοση της εφαρμογής.
- **testInstrumentationRunner**: Υποδεικνύει τη βασική κλάση για την εκτέλεση των instrumented tests.

4.3.4 Build Types, Compile και Kotlin Options

Το block **buildTypes** καθορίζει τις ρυθμίσεις για την έκδοση κυκλοφορίας της εφαρμογής:

- **isMinifyEnabled** = **false**: Απενεργοποίηση της μεθόδου ελαχιστοποίησης, η οποία αφαιρεί τα περιττά τμήματα του κώδικα.

Οι επιλογές μεταγλώττισης είναι ρυθμισμένες να χρησιμοποιούν την έκδοση **Java 17**:

- **sourceCompatibility** και **targetCompatibility**: Ορίζουν τη συμβατότητα με **Java 17**.
- **jvmTarget** = "**17**": Εξασφαλίζει ότι η **Kotlin** χρησιμοποιεί **JVM 17**.

4.3.5 Kapt και Build Features

Ενεργοποίηση βασικών λειτουργιών για την ανάπτυξη της εφαρμογής:

- **viewBinding** = **true**: Διευκολύνει την ασφαλή πρόσβαση των κλάσεων στα xml αρχεία τους μέσω binding .
- **mlModelBinding** = **true**: Παρέχει υποστήριξη για ενσωμάτωση μοντέλων μηχανικής μάθησης (ai).
- **kapt**: Παρέχει πρόσθετες ρυθμίσεις για την παραγωγή κώδικα μέσω annotation processors (χρήση για **Hilt**).

```
android { this: BaseAppModuleExtension
    namespace = "com.example.aather"
    compileSdk = 35

    sourceSets { this: NamedDomainObjectContainer<out AndroidSourceSet>
        getByName( name: "main") { this: AndroidSourceSet
            jniLibs.srcDirs("src/main/jniLibs")
            java.srcDirs("src/main/kotlin")
            res.srcDirs("src/main/res")
            manifest.srcFile( srcPath: "src/main/AndroidManifest.xml")
        }
    }

    defaultConfig { this: ApplicationDefaultConfig
        applicationId = "com.example.aather"
        minSdk = 30
        targetSdk = 34
        versionCode = 1
        versionName = "1.0"

        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes { this: NamedDomainObjectContainer< ApplicationBuildType>
        release { this: ApplicationBuildType
            isMinifyEnabled = false
            proguardFiles(
                getDefaultProguardFile( name: "proguard-android-optimize.txt"),
                "proguard-rules.pro"
            )
        }
    }

    compileOptions { this: CompileOptions
        sourceCompatibility = JavaVersion.VERSION_17
        targetCompatibility = JavaVersion.VERSION_17
    }

    kotlinOptions { this: KotlinJvmOptions
        jvmTarget = "17"
    }
```

```
java { this: JavaPluginExtension
    toolchain { this: JavaToolchainSpec
        languageVersion = JavaLanguageVersion.of( version: 17)
    }
}

kapt { this: KaptExtension
    javacOptions { this: KaptJavacOption
        option( name: "-source", value: "17")
        option( name: "-target", value: "17")
    }
}
```

Εικόνα 4.11 Build.gradle (:app)

4.4 Ενσωματωμένες Βιβλιοθήκες και Λειτουργίες του Project

Παρακάτω παρουσιάζονται οι βασικές κατηγορίες εξαρτήσεων (dependencies) του πρότζεκτ και οι δυνατότητές τους:

- **Διαχείριση Τοπικών Βιβλιοθηκών**:
  - Χρήση αρχείων .jar και .aar από τον κατάλογο libs.
- **Επεξεργασία Εικόνας**:
  - **PhotoView**: Παρέχει λειτουργίες ζουμ και περιστροφής εικόνων.
- **Διεπαφή Χρήστη**:
  - **Material Components**: Για τη δημιουργία σύγχρονων διεπαφών UI.
  - **Constraint Layout**: Διαχείριση διάταξης στοιχείων με περιορισμούς.
  - **CircleIndicator**: Οπτική ένδειξη σελιδοποίησης για ViewPager και ViewPager2.
- **Πλοήγηση**:
  - Χρήση του **Navigation Component** για πλοήγηση μεταξύ οθονών.
- **Δικτύωση**:
  - **Retrofit**: Διαχείριση αιτημάτων HTTP και αποστολή δεδομένων.
  - **OkHttp**: Γρήγορη και ασφαλή διαχείριση αιτημάτων και συνδέσεων HTTP.
  - **Gson**: Μετατροπή JSON δεδομένων σε αντικείμενα Kotlin.
- **Υποστήριξη Κάμερας**:
  - **CameraX**: Παρέχει λειτουργίες προβολής, λήψης και ανάλυσης εικόνων.
  - **OpenCV:** Ενσωμάτωση του ως module για προηγμένη διαχείριση εικόνας.
- **Firebase Υπηρεσίες**:
  - **Firebase Database**: Διαχείριση και συγχρονισμός δεδομένων.
  - **Firebase Authentication**: Έλεγχος ταυτότητας χρηστών.
  - **Firebase Analytics**: Ανάλυση των δεδομένων χρήσης.
- **Μηχανική Μάθηση**:
  - **ML Kit Text Recognition**: Αναγνώριση κειμένου σε πολλαπλά συστήματα γραφής:
    - Λατινικά, Κινέζικα, Γιαπωνέζικα, Κορεάτικα, και Ντεβαναγκάρι.
- **Διαχείριση Εξουσιοδοτήσεων**:
  - **PermissionX**: Διαχείριση αδειών χρήστη.
- **Dependency Injection**:
  - **Hilt**: Απλοποιεί τη διαχείριση εξαρτήσεων (dependencies).
- **Δοκιμές**:
  - **JUnit**: Framework για unit tests (@Test).
  - **Espresso**: Framework για δοκιμές στις διεπαφές UI.

Οι βιβλιοθήκες αυτές μπορούν να παρατηρηθούν στην Εικόνα 4.12.

```
dependencies { this: DependencyHandlerScope

    implementation(fileTree(mapOf("dir" to "libs", "include" to listOf("*.jar", "*.aar"))))
    implementation ("com.github.chrisbanes:PhotoView:2.3.0")
    val navVersion = "2.6.0"
    implementation("androidx.navigation:navigation-fragment-ktx:$navVersion")
    implementation("androidx.navigation:navigation-ui-ktx:$navVersion")
    implementation ("com.squareup.retrofit2:converter-gson:2.9.0")
    implementation ("com.squareup.okhttp3:okhttp:4.9.0")
    implementation("androidx.core:core-ktx:1.12.0")
    implementation("androidx.appcompat:appcompat:1.6.1")
    implementation("com.google.android.material:material:1.11.0")
    implementation("androidx.constraintlayout:constraintlayout:2.1.4")
    implementation("androidx.navigation:navigation-ui-ktx:2.6.0")
    implementation("com.google.android.gms:play-services-location:21.2.0")
    implementation("com.google.firebase:firebase-database-ktx:21.0.0")
    implementation("com.google.firebase:firebase-auth-ktx:23.0.0")
    implementation(project(":opencv"))
    //implementation("com.google.ai.edge.litert:litert-gpu:1.0.1")
    testImplementation("junit:junit:4.13.2")
    androidTestImplementation("androidx.test.ext:junit:1.1.5")
    androidTestImplementation("androidx.test.espresso:espresso-core:3.5.1")
    implementation("androidx.viewpager2:viewpager2:1.0.0")
    implementation("me.relex:circleindicator:2.1.6")
    implementation("com.squareup.retrofit2:retrofit:2.9.0")
```

```
// CameraX
implementation("androidx.camera:camera-camera2:1.3.4")
implementation("androidx.camera:camera-lifecycle:1.3.4")
implementation("androidx.camera:camera-view:1.3.4")

// Firebase
implementation(platform("com.google.firebase:firebase-bom:33.1.1"))
implementation("com.google.firebase:firebase-analytics")
implementation ("com.google.dagger:hilt-android:2.44")
kapt ("com.google.dagger:hilt-android-compiler:2.44")
implementation ("com.google.code.gson:gson:2.10.1")
implementation ("com.mesibo.api:webrtc:1.0.5")
implementation ("com.guolindev.permissionx:permissionx:1.6.1")
implementation ("com.google.firebase:firebase-ml-vision:15.0.0")

// To recognize Latin script
implementation ("com.google.mlkit:text-recognition:16.0.1")
// To recognize Chinese script
implementation ("com.google.mlkit:text-recognition-chinese:16.0.1")
// To recognize Devanagari script
implementation ("com.google.mlkit:text-recognition-devanagari:16.0.1")
// To recognize Japanese script
implementation ("com.google.mlkit:text-recognition-japanese:16.0.1")
// To recognize Korean script
implementation ("com.google.mlkit:text-recognition-korean:16.0.1")
```

Εικόνα 4.12 Dependencies του Aather

4.5 Περιγραφή Δικαιωμάτων Εφαρμογής

Ακολουθεί η περιγραφή των κύριων δικαιωμάτων από την Εικόνα 4.13 που έχουν δηλωθεί στο αρχείο **AndroidManifest.xml**:

Δικαιώματα Πολυμέσων και Προβολής Μέσων:

- **android.permission.FOREGROUND_SERVICE_MEDIA_PROJECTION** & **android.permission.PROJECT_MEDIA**
  Επιτρέπουν την καταγραφή και προβολή μέσων, όπως καταγραφή οθόνης ή βίντεο σε πραγματικό χρόνο.

Επικοινωνία και Δικτύωση

- **android.permission.SEND_SMS**
  Δυνατότητα αποστολής SMS μηνυμάτων.
- **android.permission.INTERNET**
  Απαραίτητο για τη σύνδεση στο διαδίκτυο.
- **android.permission.ACCESS_NETWORK_STATE**
  Έλεγχος της κατάστασης σύνδεσης στο δίκτυο.

Πρόσβαση Ήχου και Κάμερας

- **android.permission.RECORD_AUDIO**
  Επιτρέπει την καταγραφή ήχου, απαραίτητη για λειτουργίες όπως η αναγνώριση ομιλίας.
- **android.permission.MODIFY_AUDIO_SETTINGS**
  Διαμόρφωση των ηχητικών ρυθμίσεων.
- **android.permission.CAMERA**
  Παρέχει πρόσβαση στην κάμερα για λειτουργίες όπως η ανίχνευση αντικειμένων και η λήψη φωτογραφιών.
- **android.permission.CAPTURE_VIDEO_OUTPUT**
  Καταγραφή βίντεο σε πραγματικό χρόνο.
- **android.hardware.camera.any**
  Δηλώνει την υποστήριξη της κάμερας από τη συσκευή.

Πρόσβαση Τοποθεσίας

- **android.permission.ACCESS_FINE_LOCATION**
  Πρόσβαση στην ακριβή ανίχνευση τοποθεσίας του χρίστη.
- **android.permission.ACCESS_COARSE_LOCATION**
  Γενική ανίχνευση τοποθεσίας για εφαρμογές που δεν χρειάζονται λεπτομερή δεδομένα.

Πρόσβαση σε Αρχεία και Πολυμέσα

- **android.permission.READ_MEDIA_IMAGES,**
  android.permission.READ_MEDIA_VIDEO,
  android.permission.READ_MEDIA_AUDIO
  Επιτρέπει την ανάγνωση αρχείων και πολυμέσων από τη συσκευή.
- **android.permission.READ_EXTERNAL_STORAGE &**
  android.permission.WRITE_EXTERNAL_STORAGE
  Απαραίτητα για την πρόσβαση και επεξεργασία εξωτερικών αρχείων.

Λειτουργίες Συστήματος

- **android.permission.FOREGROUND_SERVICE**
  Χρήση υπηρεσιών που εκτελούνται στο παρασκήνιο.
- **android.permission.SYSTEM_ALERT_WINDOW**
  Δημιουργία παραθύρων πάνω από άλλες εφαρμογές, όπως οι ειδοποιήσεις και τα widgets.

Άλλες Λειτουργίες

- **android.permission.WAKE_LOCK**
  Εμποδίζει τη συσκευή να τεθεί σε λειτουργία ύπνου (suspend state).
- **android.permission.VIBRATE**
  Ενεργοποίηση δόνησης για **haptic** λειτουργίες.

```
<uses-permission android:name="android.permission.FOREGROUND_SERVICE_MEDIA_PROJECTION" />
<uses-permission android:name="android.permission.PROJECT_MEDIA"/>
<uses-permission android:name="android.permission.SEND_SMS" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.READ_MEDIA_IMAGES"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.CAPTURE_VIDEO_OUTPUT"
    tools:ignore="ProtectedPermissions" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
<uses-permission android:name="android.permission.READ_MEDIA_IMAGES"/>
<uses-permission android:name="android.permission.READ_MEDIA_VIDEO"/>
<uses-permission android:name="android.permission.READ_MEDIA_AUDIO"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-feature android:name="android.hardware.camera.any" />
```

Εικόνα 4.13 Δικαιώματα του Aather

Κεφάλαιο 5ο: Ανάλυση Λειτουργιών, Activities και Κώδικα

Το κεφάλαιο αυτό εστιάζει στην αναλυτική παρουσίαση των λειτουργιών και δραστηριοτήτων (Activities) της εφαρμογής, αναδεικνύοντας τον ρόλο κάθε λειτουργίας στη συνολική εμπειρία του χρήστη. Η δομή του κεφαλαίου περιλαμβάνει την περιγραφή κάθε λειτουργίας, τη σύνδεσή της με το αντίστοιχο xml κώδικα και την τεχνολογική προσέγγιση που υιοθετήθηκε για την υλοποίηση τους.

5.1 Λειτουργίες Εγγραφής και Σύνδεσης Χρηστών

Η εφαρμογή όπως είχε προαναφερθεί σε προηγούμε κεφαλαιο, παρέχει δύο επιλογες σύνδεσης για τον χρήστη: την **εγγραφή νέου χρήστη** και τη **σύνδεση σε ήδη υπάρχον λογαριασμό**. Αυτές οι λειτουργίες είναι απαραίτητες για την προσωποποιημένη χρήση της εφαρμογής και την αποθήκευση δεδομένων χρήστη στο Firebase.

5.1.1 Εγγραφή Νέου Χρήστη

Η εγγραφή νέου χρήστη δίνει τη δυνατότητα στους χρήστες να δημιουργήσουν λογαριασμό εισάγοντας τα προσωπικά τους στοιχεία (αναφορά στην Εικόνα 5.1). Η εφαρμογή απαιτεί την παροχή:

- **Ονόματος χρήστη (username)**
- **Κωδικού πρόσβασης (password)**
- **Ύψους (height)**
- **Στοιχείου επαφής SOS (sosContact)**

Κατά τη διαδικασία εγγραφής, τα δεδομένα επαληθεύονται ως προς την πληρότητά τους. Εάν ο χρήστης παραλείψει κάποιο πεδίο, εμφανίζεται σχετικό μήνυμα λάθους. Τα δεδομένα αποθηκεύονται στη βάση δεδομένων Firebase, σε έναν ξεχωριστό κόμβο **users**, όπου κάθε χρήστης αποκτά ένα μοναδικό αναγνωριστικό αριθμό (user ID).

Η εφαρμογή παρέχει επίσης τη δυνατότητα δυναμικής ανάκτησης δεδομένων, όπως το όνομα χρήστη, ώστε να χρησιμοποιηθούν σε επόμενα στάδια της λειτουργίας, π.χ., για την προσαρμογή της διεπαφής. Μετά την ολοκλήρωση της εγγραφής, ο χρήστης οδηγείται σε ένα βήμα επιλογής χρήσης (usage), όπου μπορεί να επιλέξει τον ρόλο του στην εφαρμογή.

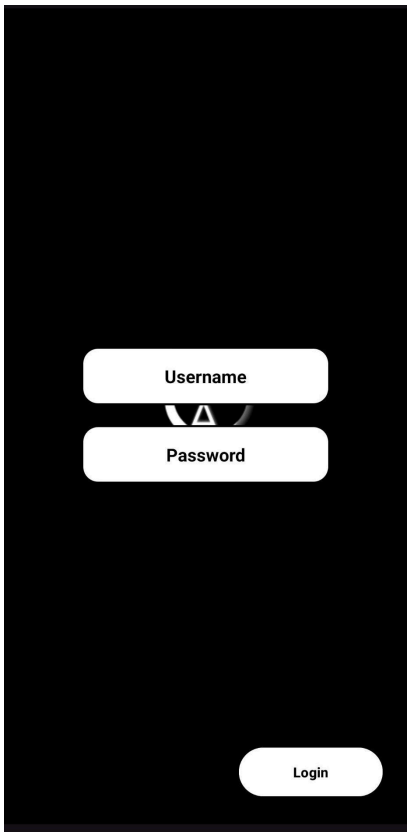
5.1.2 Σύνδεση Υπάρχοντος Χρήστη

Η λειτουργία σύνδεσης επιτρέπει στους χρήστες να εισέλθουν στον ήδη υπάρχον λογαριασμό τους, χρησιμοποιώντας το όνομα χρήστη και τον κωδικό πρόσβασής τους (αναφορά στην Εικόνα 5.2). Με την εισαγωγή των στοιχείων:

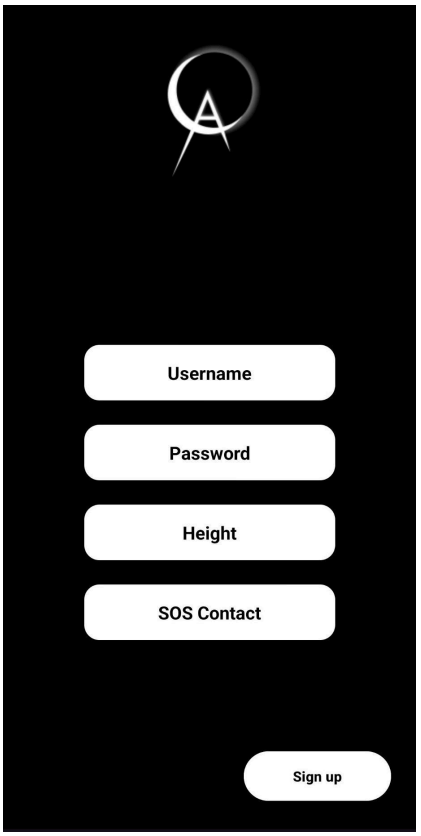
1. Η εφαρμογή ελέγχει αν υπάρχει χρήστης με το συγκεκριμένο όνομα στη βάση δεδομένων.
2. Επαληθεύει τον κωδικό πρόσβασης.

Σε περίπτωση επιτυχούς σύνδεσης, το σύστημα αναγνωρίζει τον χρήστη μέσω του μοναδικού user ID και τον κατευθύνει στην επόμενη λειτουργία, ανάλογα με τον προκαθορισμένο ρόλο του (π.χ., χρήστης με προβλήματα ακοής ή όρασης). Εάν τα δεδομένα είναι λανθασμένα ή ο χρήστης δεν

υπάρχει, εμφανίζεται κατάλληλο μήνυμα σφάλματος, π.χ., "Ο χρήστης δεν βρέθηκε" ή "Λανθασμένος κωδικός".



Εικόνα 5.1 Log in



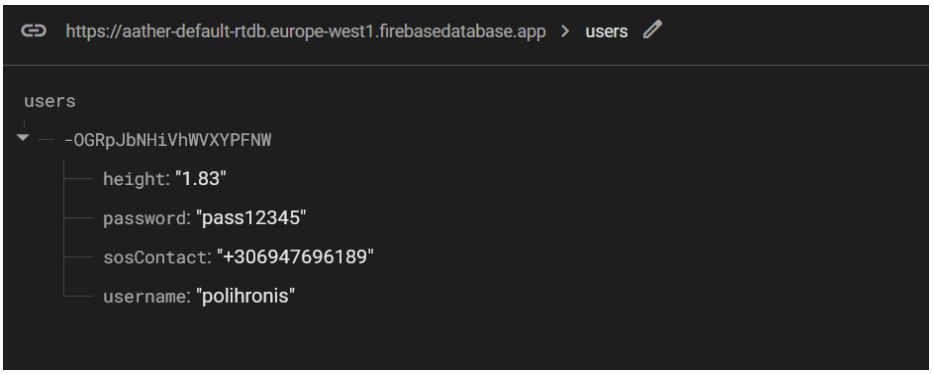
Εικόνα 5.2 Sign up

5.1.3 Τεχνικές Υλοποίησης και Αλληλεπίδραση με το Firebase

Για την υλοποίηση των λειτουργιών χρησιμοποιήθηκαν:

- Firebase Realtime Database: Όλα τα δεδομένα χρηστών αποθηκεύονται σε κόμβους με μοναδικά κλειδιά (Στην Εικόνα 5.3 παρατίθεται παράδειγμα).
- Android UI Components: Όπως EditText, Button και Toast, για την αλληλεπίδραση του χρήστη με την εφαρμογή.
- Animation Libraries: Χρησιμοποιήθηκαν για την αισθητική μετάβαση ανάμεσα στα διαφορετικά στάδια, όπως η εμφάνιση και απόκρυψη πεδίων ή κουμπιών.

```
Polihronis Varvaris *
private fun registerUser(username: String, password: String, height: String, sosContact: String) {
    // Create a user map to store the data
    val user = mapOf(
        "username" to username,
        "password" to password,
        "height" to height,
        "sosContact" to sosContact
    )

    // Push the user data to Firebase under the "users" node
    val newUserRef = database.child( pathString: "users").push()
    newUserRef.setValue(user)
        .addOnCompleteListener { task ->
            if (task.isSuccessful) {
                currentUserId = newUserRef.key
                retrieveUserName(currentUserId!!) { fetchedUsername ->
                    if (fetchedUsername != null) {
                        this.username = fetchedUsername
                        Toast.makeText( context: this,  text: "Username retrieved: $fetchedUsername", Toast.LENGTH_SHORT).show()
                    } else {
                        Toast.makeText( context: this,  text: "Failed to retrieve username", Toast.LENGTH_SHORT).show()
                    }
                }
                fadeOutEditTextsAndSubmitButton {
                    fadeInImageChoices()
                }
            } else {
                Toast.makeText( context: this,  text: "Registration failed: ${task.exception?.message}", Toast.LENGTH_SHORT).show()
            }
        }
}
```



```
private fun checkUserCredentials(username: String, password: String) {
    database.child( pathString: "users").orderByChild( path: "username").equalTo(username).get()
        .addOnSuccessListener { dataSnapshot ->
            if (dataSnapshot.exists()) {
                for (childSnapshot in dataSnapshot.children) {
                    val userPassword = childSnapshot.child( path: "password").getValue(String::class.java)
                    val usage = childSnapshot.child( path: "usage").getValue(String::class.java)

                    if (userPassword == password) {
                        currentUserId = childSnapshot.key
                        Toast.makeText( context: this,  text: "Login successful", Toast.LENGTH_SHORT).show()
                        navigateToNextScreen( usage: usage ?: "eyes")
                        return@addOnSuccessListener
                    }
                }
                Toast.makeText( context: this,  text: "Incorrect password", Toast.LENGTH_SHORT).show()
            } else {
                Toast.makeText( context: this,  text: "User not found", Toast.LENGTH_SHORT).show()
            }
        }
        .addOnFailureListener { exception ->
            Toast.makeText( context: this,  text: "Login failed: ${exception.message}", Toast.LENGTH_SHORT).show()
        }
}
```

Εικόνα 5.3 Υλοποίηση Sign Up και Log in

5.1.4 Επιλογή Ρόλου Νέου Χρήστη

Ο χρήστης αφου εχει κανει εισαγωγή των στοιχείων του και εχει πατήσει το LOG IN button, εμφανίζεται η νεα διεπαφή με τον χρήστη η οποία μπορεί να παρατηρηθεί στην Εικόνα 5.4. Εκεί ο χρήστης μπορεί να επιλέξει έναν από τους τρείς ρόλους του. Οι ρόλοι εχουν να κάνουν με τις λειτουργίες που θέλει ο user να του παρέχει η εφαρμογή και γινονται update στο firebase (Εικονα 5.5). Οι ρόλοι έχουν αναφερθεί και στην υποενότητα 3.4.6 και είναι οι παρακάτω:

- **Ears**: Άτομα με Δυσκολία στην Ακοή
- **Volunteer**: Εθελοντές
- **Eyes**: Άτομα με Δυσκολία στην Όραση

Εικόνα 5.4 Επιλογή Ρόλου

Εικόνα 5.5 Παράδειγμα επιλογής ρόλου ears

5.2 Εργαλείο Μεγέθυνσης και Σμίκρυνσης

Το συγκεκριμένο Activity προκύπτει απο την επιλογη του SCALE του χρήστη (Εικόνα 3.12). Το νέο αυτό Activity ονομάζεται **Size_image** και δίνει την δυνατότητα στον χρήστη να κάνει zoom in και zoom out σε φωτογραφίες της επιλογής του. Για να επιλέξει φωτογραφίες η εφαρμογή παρέχει δύο τρόπους. Ο ένας είναι με την χρήση κάμερας για την λήψη φωτογραφίας. Ενώ ο δεύτερος τρόπος είναι με την επιλογή φωτογραφίας από την συλλογή του.

Στο xml αρχείο του **Size_image** υπάρχουν τα παρακάτω στοιχεία:

- **backbutton**: Επιστρεφει τον χρήστη στο menu.
- **imageforzoombutton**: Ανοίγει την κάμερα του χρήστη για λήψη φωτογραφίας.
- **previesfromgallery**: Ανοίγει την την συλλογή του χρήστη για επιλογή φωτογραφίας.

- **photoViewContainer**: Είναι ένα FrameLaout στο οποιο εισαγεται η βιβλιοθηκη PhotoView. Λειτουργεί σαν κοντέινερ για την εικόνα.

5.2.1 PhotoView και Διαχείριση Εικόνας

Το PhotoView είναι η βασική λειτουργική μονάδα για την εμφάνιση και διαχείριση της εικόνας. Χάρη στη βιβλιοθήκη PhotoView , ο χρήστης μπορεί να μεγεθύνει και να μικρύνει την εικόνα με φυσικές κινήσεις (pinch-to-zoom). Η ενσωμάτωση της συγκεκριμένης βιβλιοθήκης υπάρχει στο **build.gradle.kts (:app)** αρχείο με τον παρακάτω κώδικα :

- implementation ("com.github.chrisbanes:PhotoView:2.3.0")

5.2.2 Επεξήγηση της Κλάσης Size_image

Λήψη Φωτογραφίας:

- Όταν ο χρήστης πατήσει το κουμπί **imageforzoombutton**, δημιουργείται ένα προσωρινό αρχείο εικόνας με τη μέθοδο **createImageFile**. Μέσω του **Intent** ανοίγει η κάμερα για λήψη φωτογραφίας. Η φωτογραφία αποθηκεύεται στο **URI** που παρέχεται από το FileProvider. Με την μέθοδο **onActivityResult** η εικόνα εμφανίζεται στο **PhotoView** μέσω του URI (**photoUri**).

Επιλογή από Συλλογή:

- Το κουμπί **btnSelectGallery** ανοίγει την επιλογή εικόνας από τη συλλογή της συσκευής με χρήση του Intent.**ACTION_PICK**. Η επιλεγμένη εικόνα φορτώνεται στο **PhotoView** από το **URI** της συλλογής (**selectedImageUri**).

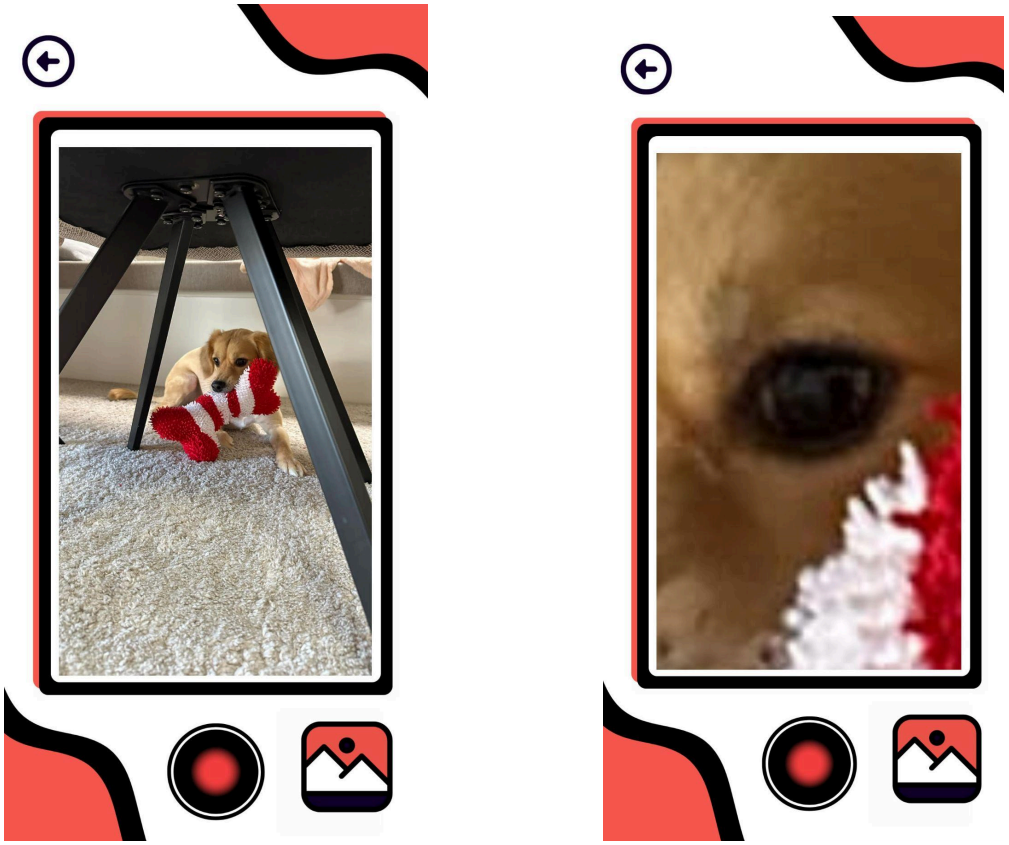
PhotoView: Ορίζεται μέγιστη και ελάχιστη κλίμακα ζουμ μέσω PhotoViewAttacher:

- **attacher.setMaximumScale(10f):** Επιτρέπει τη μεγέθυνση της εικόνας.
- **attacher.setMinimumScale(1f):** Παρουσιάζει την εικόνα στη φυσική της κλίμακα.
- Η αρχική κλίμακα ορίζεται με την ιδιότητα **scaleType** ως **CENTER_CROP**, για να διασφαλιστεί η εμφάνιση της εικόνας στο κεντρο της οθόνης.
- Με την χρήση του **double click** ο χρήστης έχει την δυνατότητα να πραγματοποιήσει zoom in και zoom out λειτουργίες , όπως επίσης με την χρήση pinch-to-zoom που αναφέρθηκε προηγουμένως. Αυτό λόγου της βιβλιοθήκης **PhotoView.**

URI στο Android: Στο Android, τα **URI** χρησιμοποιούνται συχνά για να αναφέρονται σε αρχεία, δεδομένα ή άλλους πόρους, όπως:

- **content://**: Δείχνει σε δεδομένα που διαχειρίζονται Content Providers.
- **file://**: Δείχνει σε τοπικά αρχεία στο σύστημα.
- **http:// ή https://**: Δείχνει σε διαδικτυακούς πόρους.

Στην Εικόνα 5.6 απεικονίζεται το Activity Size_image πριν και μετα το zoom in.



```
Polihronis Varvaris *
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_sizezoom)

    photoView = findViewById(R.id.photoView)
    btnCapturePhoto = findViewById(R.id.imageforzoombutton)
    btnSelectGallery = findViewById(R.id.previesfromgallery)
    btnBack = findViewById(R.id.backbutton)


    val attacher = PhotoViewAttacher(photoView)
    attacher.setMaximumScale(10f) // maximum zoom scale
    attacher.setMinimumScale(1f) // minimum scale to 1f

    photoView.scaleType = ImageView.ScaleType.CENTER_CROP

    btnCapturePhoto.setOnClickListener { it: View!
        val photoFile = createImageFile()
        photoUri = FileProvider.getUriForFile(
            context: this,
            authority: "${applicationContext.packageName}.fileprovider",
            photoFile
        )
```

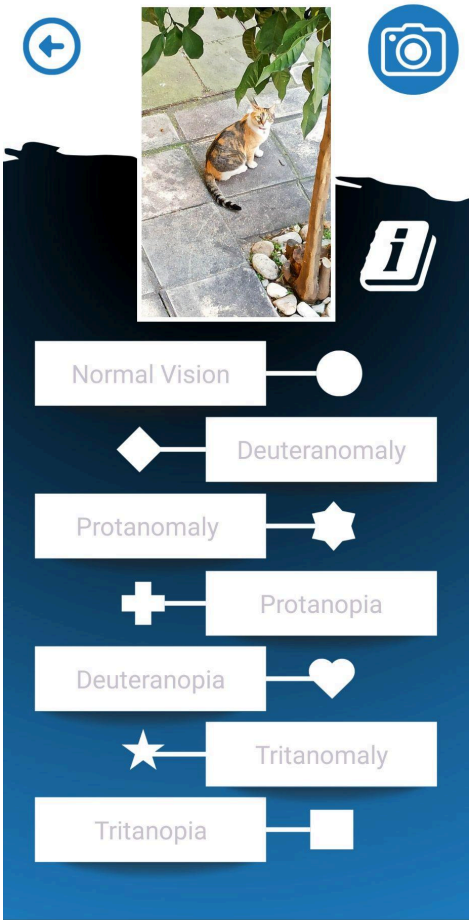
Εικόνα 5.6 Μερος Υλοποίησης και παράδειγμα Χρήσης Size_image

5.3 Εργαλείο Ανίχνευσης χρώματος

Το εργαλείο ανίχνευσης χρώματος βρίσκεται στην κλάση **PickColorFilter** και ο σκοπός του είναι η προσομοίωση διαφόρων διαταραχών αντίληψης χρώματος (αχρωματοψία). Η πρόσβαση στο Activity γίνεται μετά την επιλογή της Εικόνας 3.13 απο τον χρήστη, έχοντας σαν αποτέλεσμα την μεταφορά του χρήστη στην διεπαφή της Εικόνας 5.7.

Στο xml αρχείο του **PickColorFilter** υπάρχουν τα παρακάτω στοιχεία:

- **backbutton**: Επιστρεφει τον χρήστη στο menu.
- **camerabuttoncolor**: Ανοίγει την κάμερα του χρήστη για λήψη φωτογραφίας.
- **imageholdfilter**: Είναι ένα ImageView στο οποίο εμφανίζεται η τελευταία εικόνα της συλλογής, χρησιμοποιείται για την επιλογή σημείου στο οποίο θα τρέξει η ανίχνευση χρώματος.
- **menulistdropbutton**: Ανοίγει ένα καινούργιο πλαίσιο με οδηγίες για την απεικόνιση των παθήσεων.
- **TextViews**: Αποτυπώνει το επεξεργασμένο χρώμα και τον hex κώδικα του χρώματος.



Εικόνα 5.6 Διεπαφή του PickColorFilter

5.3.1 Ανάλυση στις Επιλεγμένες Κατηγορίες Αχρωματοψίας

Οι κατηγορίες αχρωματοψίας που επιλέχθηκαν είναι οι πιο συνηθισμένες παγκοσμίως. Για την μετατροπή τους και την οπτικοποίηση τους χρησιμοποιήθηκαν matrix βασισμένα σε ερευνες που δημοσιεύτηκαν σε επιστημονικά περιοδικά[15][16].

Στον παρακάτω πίνακα (Πίνακας 5.3) παρουσιάζονται οι κατηγορίες με τα βασικά χαρακτηριστικά τους:

Πάθηση	Matrix	Περιγραφή
Πρωτανωμαλία (Protanomaly)	[0.817 0.183 0.000] R [0.333 0.667 0.000] G [0.000 0.125 0.875] B	Ορισμένες αποχρώσεις του κόκκινου φαίνονται πιο πράσινες και λιγότερο φωτεινές.
Δευτερανωμαλία (Deuteranomaly)	[0.800 0.200 0.000] R [0.258 0.742 0.000] G [0.000 0.142 0.858] B	Κάνει ορισμένες αποχρώσεις του πράσινου να φαίνονται πιο κόκκινες
Τριτανωμαλία (Tritanomaly)	[0.967 0.033 0.000] R [0.000 0.733 0.267] G [0.000 0.183 0.817] B	Δύσκολη να διάκριση διαφοράς μεταξύ μπλε - πράσινου και μεταξύ κίτρινου - κόκκινου
Πρωτανοπία (Protanopia)	[0.567 0.433 0.000] R [0.558 0.442 0.000] G [0.000 0.242 0.758] B	Αδυναμία αντίληψης κόκκινου χρώματος.
Δευτερανοπία (Deuteranopia)	[0.625 0.375 0.000] R [0.700 0.300 0.000] G [0.000 0.300 0.700] B	Αδυναμία αντίληψης πράσινου χρώματος.
Τριτανοπία (Tritanopia)	[0.950 0.050 0.000] R [0.000 0.433 0.567] G [0.000 0.475 0.525] B	Αδυναμία αντίληψης μπλέ χρώματος.

Πίνακας 5.3 Matrix για Αχρωματοψία

5.3.2 Επεξήγηση της Κλάσης PickColorFilter

Φόρτωση Τελευταίας Εικόνας:

- **loadLastImage**() φορτώνει την πιο πρόσφατη εικόνα από τη συλλογή του χρήστη και την εμφανίζει στο **ImageView**.

Ανίχνευση Χρωμάτων:

- Μέσω της **OnTouchListener** στο **imageholdfilter**, ο χρήστης μπορεί να κάνει κλικ στην εικόνα και να δει τους κωδικούς RGB και Hex του σημείου.

Προσομοιώσεις Αχρωματοψίας:

- Με τις μεθόδους **simulateDeuteranomalyColor, simulateProtanomalyColor, simulateProtanopiaColor, simulateDeuteranopiaColor, simulateTritanomalyColor, simulateTritanopiaColor,** μετατρέπεται το σημείο επιλογής της φωτογραφίας σε νέο χρώμα προσαρμοσμένο στις παθησεις. Αυτό γίνεται με τον πολλαπλασιασμό των RGB της φωτογραφίας, με τους πίνακες matrix των παθήσεων.

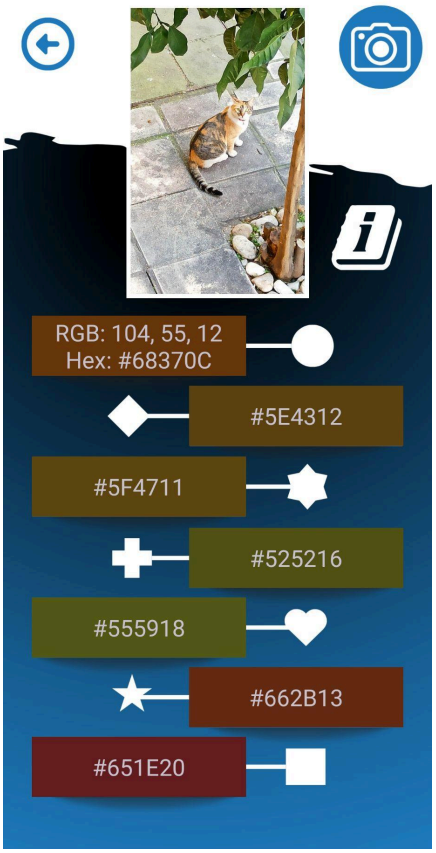
Ενημέρωση UI:

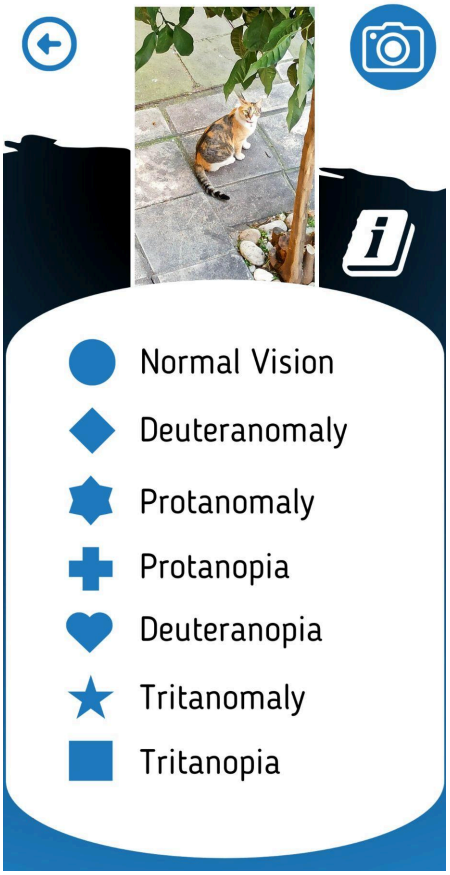
- Όταν ανιχνεύεται το νέο χρώμα ενημερώνονται τα **TextView** και **View**, για να εμφανιστούν τα αποτελέσματα Hex και η χρωματική προσομοίωση.

Επιπλέον Λειτουργία:

- Παρέχεται η δυνατότητα χρήσης προσαρμοσμένης κάμερας με το btnPhoto, Που μεταβιβάζει τον χρήστη στη δραστηριότητα **CameraPicker**.

Στις Εικόνες 5.7 και 5.8 παρουσιάζεται παράδειγμα χρήσης του εργαλείου ανίχνευσης.





Εικόνα 5.7 Προσομοίωση του PickColorFilter

Εικόνα 5.8 Information του PickColorFilter

5.4 Εργαλείο Ανίχνευσης κειμένου

Η λειτουργία ανίχνευσης κειμένου για να παρουσιαστεί στον χρήστη, πρέπει να πραγματοποιηθεί κλικ στην εικόνα 3.14. Το συγκεκριμένο Activity ονομάζεται **AiTxetImage** και σαν σκοπό έχει: την αναγνώριση κειμένου από φωτογραφία και την μετατροπή του σε ομιλία . Ο χρήστης αρχικά ερχεται σε επαφη με την διεπαφή της πρώτης Εικόνας 5.9. Όπως και και σε προηγούμενο εργαλείο, ο χρήστης έχει την επιλογή να τραβήξει μια νέα φωτογραφία για ανάλυση, ή να διαλέξει μια φωτογραφία απο την συλλογη του.

Στο xml αρχείο του **AiTxetImage** υπάρχουν τα παρακάτω στοιχεία:

- **backbutton**: Επιστρεφει τον χρήστη στο menu.
- **btnPhotoAi**: Ανοίγει την κάμερα του χρήστη για λήψη φωτογραφίας.
- **imagefortxtgeneretion**: Είναι ένα ImageView στο οποίο θα εμφανιστεί η επιλεγμένη εικόνα από την κάμερα ή την συλλογή.
- **btngalerry**: Ανοίγει την την συλλογή του χρήστη για επιλογή φωτογραφίας.
- **btndetect**: Κουμπί το οποίο την στιγμή που πατηθεί αναλύεται η εικόνα στο ImageView.
- **textgeneretedbyai**: Αποτυπώνει το κείμενο που βρέθηκε από την εικόνα.
- **soundbutton**: Μετατρέπει το κείμενο σε ομιλία.

5.4.1 Βασικές Τεχνολογίες

Οι δύο κύριες τεχνολογίες που χρησιμοποιήθηκαν είναι:

Google ML Kit (Text Recognition): Χρησιμοποιήθηκε για την αναγνώριση κειμένου από εικόνες μέσω της βιβλιοθήκης ML Kit της Google. Αντί για το Firebase Vision, η βιβλιοθήκη ML Kit είναι πιο εξειδικευμένη για αναγνώριση κειμένου και χρησιμοποιεί μοντέλα για διάφορες γραφές (Οι γραφές αναλύθηκαν στο υποκεφάλαιο των dependencies). Επίσης έχει υψηλή απόδοση με χαμηλή καθυστέρηση και λειτουργεί αυτόματα με bitmap (κατάλληλο για κινητές συσκευές).

Android Text-to-Speech (TTS): Είναι ενσωματωμένο στις android συσκευές και χρησιμοποιήθηκε για την μετατροπή του κειμένου σε φωνητική έξοδο.

5.4.2 Επεξήγηση της Κλάσης AiTxetImage

Φόρτωση Εικόνας:

- **btngal** ανοίγει την συλλογή και επιτρέπει στον χρήστη να επιλεξει μια φωτογραφία για ανάλυση.
- **btnPhotoAi** ανοίγει την κάμερα και επιτρέπει στον χρήστη τραβήξει μια φωτογραφία για ανάλυση.

Μετατροπή Εικόνας για Ανίχνευση:

- Το **MediaStore.Images.Media.getBitmap()** μετατρέπει την εικόνα σε αντικείμενο Bitmap, για μελλοντική ανάλυση με το **Google ML Kit** .

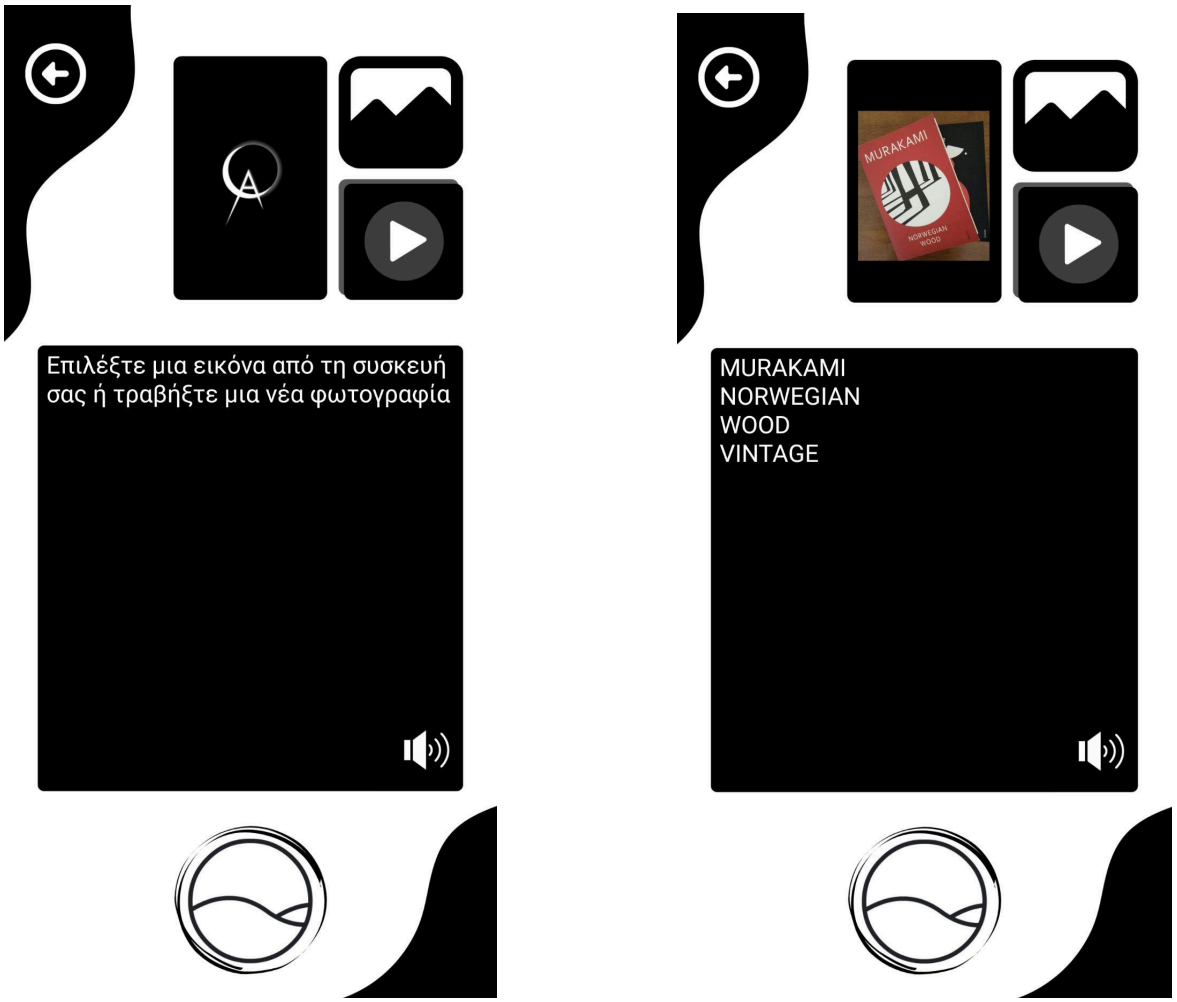
Ανίχνευση Κειμένου

- **Πατώντας το detectButton:**
  - Αν δεν έχει επιλεγεί εικόνα, εμφανίζεται μήνυμα σφάλματος.
  - Αν υπάρχει εικόνα, καλείται η detectTextFromImage().
- **detectTextFromImage(Bitmap)**
  - Χρησιμοποιείται το **Google ML Kit** για την αναγνώριση κειμένου:
    - Δημιουργείται ένα αντικείμενο InputImage από την εικόνα.
    - TextRecognition.getClient() δημιουργεί έναν αναγνωριστή.
    - Process() επεξεργάζεται την εικόνα και επιστρέφει τα αναγνωρισμένα δεδομένα.
- **Εμφάνιση Κειμένου:**
  - Σε περίπτωση που δεν ανιχνεύτηκε κείμενο, εμφανίζεται μήνυμα.
  - Αν ανιχνευτεί κείμενο, εμφανίζεται στο TextView.

Μετατροπή Κειμένου σε Ομιλία

- **Αρχικοποίηση TTS:** Χρησιμοποιείται το TextToSpeech για τη μετατροπή του κειμένου σε ομιλία

Στην Εικόνα 5.9 παρουσιάζεται παράδειγμα χρήσης του εργαλείου ανίχνευσης κειμένου.




```
Polihronis Varvaris *
private fun detectTextFromImage(bitmap: Bitmap) {
    val inputImage = InputImage.fromBitmap(bitmap, rotationDegrees: 0)
    val recognizer = TextRecognition.getClient(TextRecognizerOptions.DEFAULT_OPTIONS)

    recognizer.process(inputImage)
        .addOnSuccessListener { visionText ->
            displayText(visionText)
        }
        .addOnFailureListener { e ->
            Toast.makeText( context: this, text: "Failed to recognize text: ${e.message}", Toast.LENGTH_SHORT).show()
        }
}

new *
private fun displayText(visionText: Text) {
    if (visionText.text.isEmpty()) {
        Toast.makeText( context: this, text: "No text detected", Toast.LENGTH_SHORT).show()
    } else {
        txtDetector.text = visionText.text
    }
}
```

Εικόνα 5.9 Ροή λειτουργίας και μέρος υλοποίησης του AiTxetImage

5.5 Εργαλείο Ηχογράφησης Ομιλίας

Σαν στόχο, η κλάση **EduRecorder** έχει την διευκόλυνση των ακαδημαϊκών υποχρεώσεων του χρήστη. Αυτό επιτυγχάνεται με την παροχή του μηχανισμού ηχογράφησης ομιλίας. Το Activity εμφανίζεται στον χρήστη μετά την επιλογή της Εικόνας 3.10, έχοντας σαν αποτέλεσμα ο χρήστης να μεταφερθεί στην πρώτη διεπαφή της Εικόνας 5.10.

Στην παρούσα διεπαφή ο χρήστης μπορεί να ενεργοποιήσει το μικρόφωνο του, για την ηχογράφηση διαλέξεων ή συζητήσεων. Ακολουθώντας την ενεργοποίηση του μικροφώνου ο χρήστης μπορεί να διαβάζει την ηχογράφηση σε πραγματικό χρόνο, μετατρέποντας τον προφορικό λόγο σε γραπτό. Επίσης δίνεται η δυνατότητα στον χρήστη να αποθηκεύσει το αποτέλεσμα της διεργασίας στο κινητό του, για μελλοντική χρήση.

Στο xml αρχείο του **EduRecorder** υπάρχουν τα παρακάτω στοιχεία:

- **backbutton**: Επιστρεφει τον χρήστη στο menu.
- **btnrecorder**: Ανοίγει το μικρόφωνο του χρήστη για λήψη ηχογράφηση.
- **textrecorded**: Αποτυπώνει το κείμενο που βρέθηκε από την μετατροπή speech to text σε ενα EditText.
- **btndownload**: Αποθηκεύει το κείμενο στο κινητό.
- **btncopy**: Αντιγραφη τα περιεχόμενα του EditText στο πληκτρολόγιο (keyboard).
- **btntrash**: Διαγράφει το περιεχόμενο του EditText και τίθεται σε κατάσταση ετοιμότητας για την ηχογράφηση νεας ομιλίας.

5.5.1 Βασικές Τεχνολογίες

SpeechRecognizer API: Χρησιμοποιείται για την μετατροπή απο ομιλία σε κείμενο. Παρέχει δυνατότητες αναγνώρισης της χρονικής στιγμής που πραγματοποιείται γεγονός της ομιλίας. Για παράδειγμα αναγνωρίζει πότε ο χρήστης ξεκινά να μιλάει και πότε σταματάει. Η χρήση της συγκεκριμένης τεχνολογίας είναι απλή στην ενσωμάτωση της, καθώς είναι ενσωματωμένη στις Android συσκευές.

Android Intent: Χρησιμοποιείται για την κοινοποίηση του μετατροποιημένου κειμένου και για την αποθηκευση του. Δίνεται η επιλογή στον χρήστη να επιλέξει τον τρόπο αποθήκευσης και τον τρόπο κοινοποίησης του κειμένου. Για παράδειγμα μπορεί να αποθηκεύσω το κείμενο στις σημειώσεις του (notes).

ClipboardManager: Διαχειρίζεται την λειτουργία αντιγραφής του κειμένου που παρέχεται στον χρήστη.

5.5.2 Επεξήγηση της Κλάσης EduRecorder

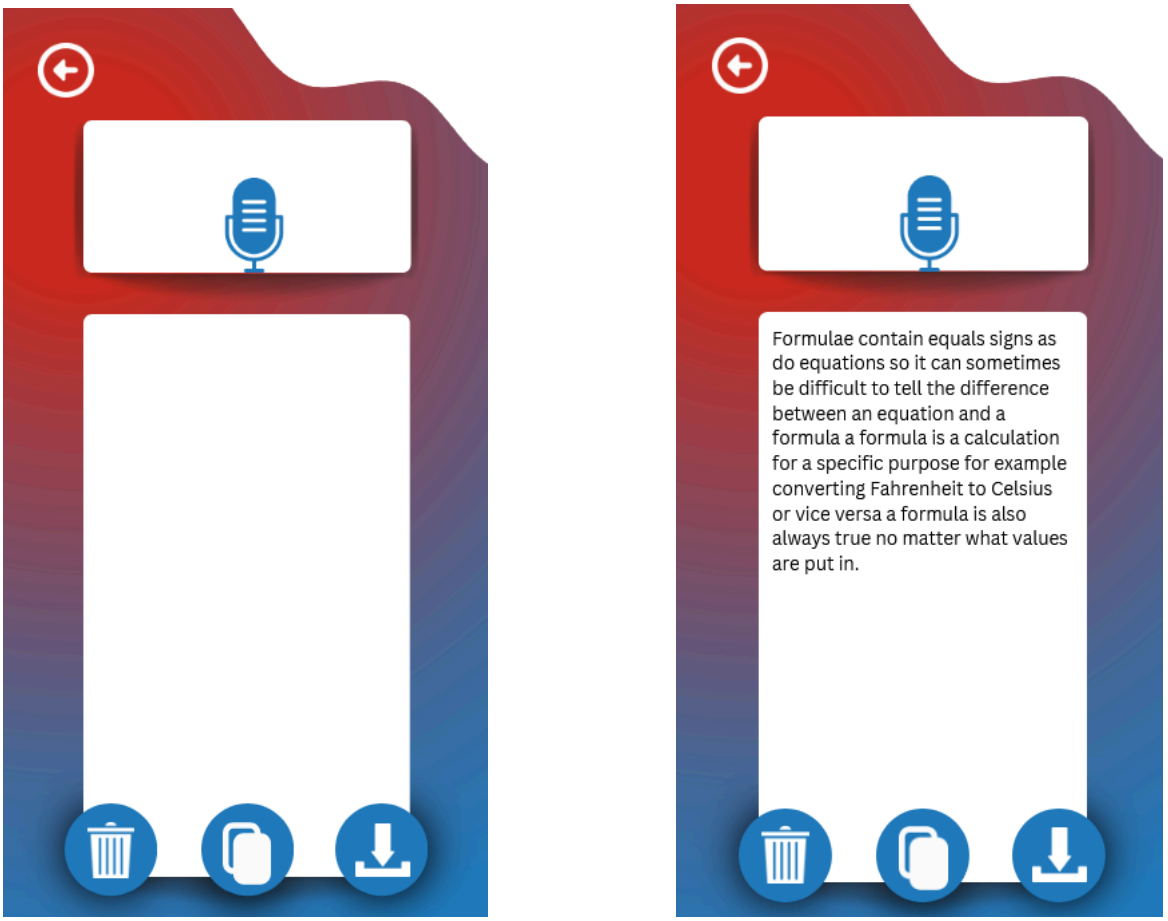
Καταγραφή Ομιλίας:

- **btnrecorder** ανοίγει το μικρόφωνο του χρήστη και ξεκινά την ηχογράφηση του περιβάλλοντός του.
- **SpeechRecognizer API** αναγνωρίζει και μετατρέπει την ομιλία σε κείμενο, το κείμενο αυτο καταγράφεται στο **EditText**.

Επεξεργασία και Χρήση Κειμένου

- Το **EditText** επιτρέπει στον χρήστη να διορθώσει τις σημειώσεις του πριν πριν προχωρήσει στα επόμενα βήματα.
- **copyTextToClipboard()** Διαβάζει το κείμενο από το **EditText** και γίνεται έλεγχος αν είναι άδειο με την χρήση **isBlank()**. Αν είναι τότε εμφανίζεται κατάλληλο μήνυμα. Αντίθετα αν έχει περιεχόμενο χρησιμοποιεί το **ClipboardManager** και αντιγράφη το κείμενο με την εντολή **setPrimaryClip().**
- **shareTextToNotes()** Διαβάζει το κείμενο από το **EditText** και ελέγχει όπως προηγουμένως αν είναι άδειο. Αν είναι γεμάτο δημιουργείται ένα καινούργιο **Intent** με το περιεχόμενο του κειμένου. Έπειτα με την χρήση του **Intent.createChooser()** δημιουργείται ένα popup, δίνοντας επιλογές κοινοποίησης και αποθήκευσης στον χρήστη.
- **clearEditText()** Επαναφέρει την μεταβλητή fullText στο αρχικό στάδιο (“ ”)

Στην Εικόνα 5.10 παρουσιάζεται παράδειγμα χρήσης του εργαλείου ηχογράφησης ομιλίας.



```
Polihronis Varvaris *
private fun createRecognitionListener(): RecognitionListener {
    return object : RecognitionListener {
        override fun onReadyForSpeech(params: Bundle?) {
            // Invoked when the recognizer is ready to start listening
        }

        override fun onBeginningOfSpeech() {
            // Invoked when the user starts speaking
        }

        override fun onRmsChanged(rmsdB: Float) {
            // Invoked to signal the sound level in the audio stream
        }

        override fun onBufferReceived(buffer: ByteArray?) {
            // Invoked when more sound has been received
        }

        override fun onEndOfSpeech() {
            // Invoked when the user stops speaking
        }

        override fun onError(error: Int) {
            Toast.makeText(applicationContext, text: "Error occurred: $error", Toast.LENGTH_SHORT).show()
        }

        override fun onResults(results: Bundle?) {
            val matches = results?.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION)
            if (matches != null && matches.isNotEmpty()) {
                fullText += matches[0] + " "
                editText.setText(fullText)
            }
        }

        override fun onPartialResults(partialResults: Bundle?) {
            val matches = partialResults?.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION)
            if (matches != null && matches.isNotEmpty()) {
            }
        }
```

Εικόνα 5.10 Μέρος υλοποίησης και ροή λειτουργίας του EduRecorder

5.6 Εργαλείο Μέτρησης Θορύβου

Η συγκεκριμένη λειτουργία βρίσκεται στην κλάση **NoiceTester,** με στόχος την μέτρηση θορύβου του χώρου και την προσομοίωση του με δόνηση για άτομα με ακουστικά προβλήματα. Επίσης με την διεπαφή **NoiceTester** η εφαρμογή παρέχει στον χρήστη λεπτομέρειες για τον θόρυβο στον χώρο τον οποίο βρίσκεται, όπως τα μεγιστα ντεσιμπέλ (decibel), τα ελάχιστα ντεσιμπέλ (decibel), τον μέσο όρο της έντασης και τα ντεσιμπέλ που αναλύονται σε πραγματικό χρόνο για την λειτουργία της προσομοίωσης. Η κλάση εμφανίζεται έπειτα από κλίκ στο Activity της Εικόνας 3.11 απο τον χρήστη, έχοντας σαν αποτέλεσμα την μεταφορά του χρήστη στην διεπαφή της Εικόνας 5.11.

Στο xml αρχείο του **NoiceTester** υπάρχουν τα παρακάτω στοιχεία:

- **backbutton**: Επιστρεφει τον χρήστη στο menu.
- **startButton**: Ανοίγει το μικρόφωνο και ξεκινάει τις μετρήσεις θορύβου.
- **volumeText**: Αποτυπώνει την μέτρηση θορύβου σε πραγματικό χρόνο χωρίς καθυστέρηση.
- **maxVolumeText**: Εμφανίζει το μέγιστο επίπεδο θορύβου που μετρήθηκε.
- **minVolumeText**: Εμφανίζει το ελάχιστο επίπεδο θορύβου που μετρήθηκε.
- **averageVolumeText**: Εμφανίζει τον μέσο όρο της μέτρησης θορύβου.
- **intensitySeekBar**: Ρυθμίζει την ένταση της δόνησης, χρησιμοποιείται για τη δοκιμή της δόνησης.

5.6.1 Βασικές Τεχνολογίες

MediaRecorder API : Χρησιμοποιήθηκε για την πρόσβαση στο μικρόφωνο του χρήστη και για την μέτρηση θορύβου. Η ύπαρξή του στηρίζεται άμεσα από τις Android συσκευές και αυτό το καθιστά κατάλληλο για την συγκεκριμένη εφαρμογή. Ακόμα, παρέχει δυνατότητα μέτρησης πραγματικού χρόνου και μικρό κατανάλωση πόρων (overhead).

Vibrator: Είναι ενσωματωμένο στις android συσκευές και χρησιμοποιήθηκε για την δόνηση της συσκευής.

VibrationEffect: Είναι η νέα και πιο ευέλικτη έκδοση του **Vibrator,** προσφέροντας περισσότερες επιλογές δόνησης στους προγραμματιστές.

Math: Είναι μαθηματική βιβλιοθήκη που βοηθάει την μετατροπή ήχου από amplitude σε db (decibel).

5.6.2 Επεξήγηση της Κλάσης NoiceTester

Εκκίνηση:

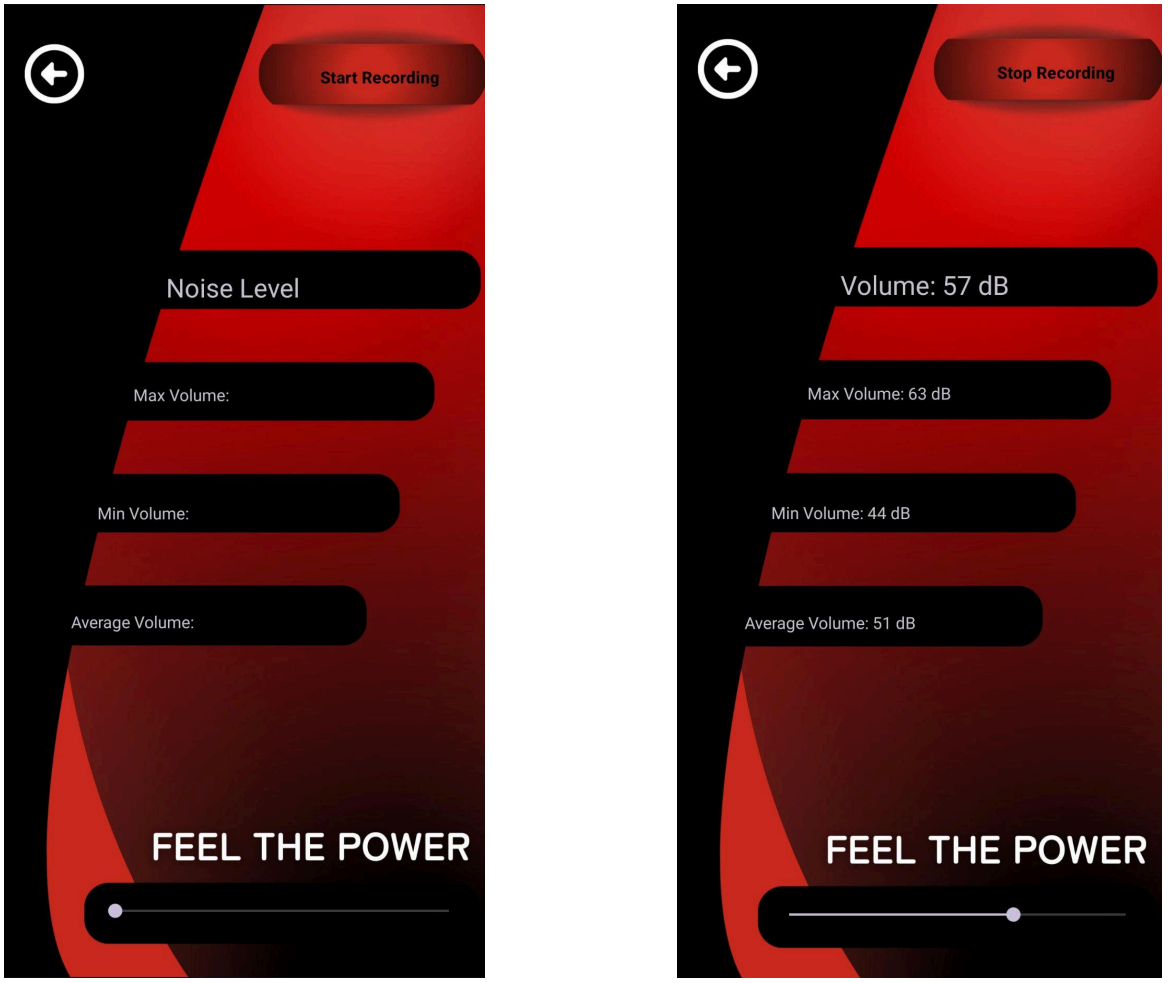
- Ο χρήστης για να ξεκινήσει τις λειτουργίες του συγκεκριμένου εργαλείου πρέπει να πατήσει το "Start Recording".
- Αρχίζουν οι λειτουργίες της διεπαφής βασισμένες στις ρυθμίσεις του **MediaRecorder.**
    - **setAudioSource(MediaRecorder.AudioSource.MIC)** Ορίζει ως πηγή ήχου να να είναι το μικρόφωνο **.**
    - **setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB)** Ορίζει η εγκραφή φωνής να γίνεται με βάση τον κωδικοποιητή AMR-N, ο οποίος θεωρείται ως ο κατάλληλος για τα μικρόφωνα που παρέχονται από τις κινητές συσκευές. Συγκεκριμένα είναι κατάλληλος για μικρόφωνα με ζώνη συχνοτήτων 200-3400 Hz.
    - **setOutputFile()** Αποθηκεύει το ηχογραφημένο αρχείο.
    - **startDecibelMeter()** Υπολογίζει την ένταση του αρχείου σε db.
    - **displayStatistics()** Εμφανίζει τις μετρούμενες εντάσεις.

Διαχείριση δόνησης:

- **mapVolumeToIntensity()** Είναι ένας χάρτης που περιέχει τις τιμές που θα πάρει η δόνηση με ανάλογα τον ηχογραφημένο θόρυβο. Παράδειγμα του κώδικα βρίσκεται στην Εικόνα 5.12

- Τα **startContinuousVibration()** και **updateVibrationPower()** ρυθμίζουν τη δόνηση με βάση την ένταση **mapVolumeToIntensity**.

Στην Εικόνα 5.13 παρουσιάζεται μέρος υλοποίησης του εργαλείου μέτρησης θορύβου.



Εικόνα 5.11 Ροή λειτουργίας του NoiceTester

```
private fun mapVolumeToIntensity(volumeInDb: Int): Int {
    return when {
        volumeInDb in 40 ≤ .. ≤ 55 -> {
            // Low intensity for volumes between 40 and 55 dB
            1
        }
```

Εικόνα 5.12 Παράδειγμα Χαμηλής Δόνησης

```
Polihronis Varvaris
private fun stopRecording() {
    mediaRecorder?.stop()
    mediaRecorder?.release()
    mediaRecorder = null
    isRecording = false
    startButton.text = "Start Recording"
    volumeText.text = "Stopped"
    displayStatistics()
    stopContinuousVibration()

    // Reset the average volume
    averageVolume = 0
}

Polihronis Varvaris *
private fun startDecibelMeter() {
    handler.post(object : Runnable {
        override fun run() {
            if (isRecording) {
                val amplitude = mediaRecorder?.maxAmplitude ?: 0
                val volumeInDb = (20 * log10(amplitude.toDouble())).roundToInt().coerceIn(0, 130)
                volumeText.text = "Volume: $volumeInDb dB"
                volumes.add(volumeInDb)

                val mappedIntensity = mapVolumeToIntensity(volumeInDb)

                updateVibrationPower(mappedIntensity)

                handler.postDelayed( r: this, delayMillis: 100)
            }
        }
    })
}
```

Εικόνα 5.13 Μέρος υλοποίησης

5.7 Εργαλείο Άμεσης Βοήθειας

Η λειτουργία άμεσης βοήθειας ενεργοποιείται όταν ο χρήστης κρατήσει πατημένο το power button για πέντε δευτερόλεπτα. Το συγκεκριμένο Activity ονομάζεται **SOSbutton** και σαν λειτουργίες έχει: την αποστολή της ακριβής τοποθεσίας του χρήστη και μήνυμα βοήθειας σε κοντινό του και έμπιστο πρόσωπο, την αυτόματη ηχογράφηση του χώρου και την αποθήκευση αυτου του αρχείου στο κινητό. Ο χρήστης με το που προκαλέσει το γεγονός για να τρέξει η **SOSbutton**, έρχεται σε επαφή με την διεπαφή της Εικόνας 5.14.

5.7.1 Βασικές Τεχνολογίες

Firebase: Χρησιμοποιήθηκε για την ανάκτηση του αριθμού τηλεφώνου που εισήγαγε ο χρήστης κατα την δημιουργία του λογαριασμού του.

MediaRecorder: Χρησιμοποιείται για την εγγραφή ήχου όπως εξηγήθηκε και προηγουμένως.

Google Play Services (FusedLocationProviderClient): Η χρήση του είναι απαραίτητη για την ακριβή τοποθεσία του χρήστη, την στιγμή της ενεργοποίησης του Activity. Χρησιμοποιεί συνδυαστικά τα δεδομένα από GPS, Wifi, δίκτυα κινητής τηλεφωνίας και αισθητήρες για ακριβέστερο εντοπισμό. Με αυτον τον τρόπο εξασφαλίζεται μεγαλύτερη ακρίβεια, ταχύτερη απόκριση και χαμηλότερη κατανάλωση πόρων σε σύγκριση με το απλό GPS.

SmsManager: Αποτελεί ένα απλό API που εντάσσεται στις εφαρμογές android και δίνεται η δυνατότητα αποστολής μηνύματος μέσα από την εφαρμογή.

5.7.2 Επεξήγηση της Κλάσης SOSbutton

Έναρξη δραστηριότητας (onCreate)

- **fetchSOSContactFromFirebase**: Ανάκτηση SOS επαφής από το Firebase
- Ορίζεται η πηγή ήχου (**MIC**), η μορφή εξόδου (**THREE_GPP**) και ο αποκωδικοποιητής του ήχου (**AMR_NB**).
- Η εγγραφή ξεκινά και σταματά αυτόματα μετα απο 30 δευτερόλεπτα με την χρήση **Handler**
- Το **UUID.randomUUID()** δημιουργεί μοναδικό όνομα αρχείου και το αποθηκεύει στην συσκευή.
- **getUserLocationAndSendSMS()**, ελέγχεται η άδεια τοποθεσίας. Αν η άδεια έχει δοθεί, ανακτάται η τοποθεσία και δημιουργείται το μήνυμα SOS.
- **sendSMS()** χειρίζεται την αποστολή του μηνύματος

```
private fun sendSMS(phoneNumber: String, message: String) {
    try {
        val smsManager = SmsManager.getDefault()
        smsManager.sendTextMessage(phoneNumber, scAddress: null, message, sentIntent: null, deliveryIntent: null)
        Toast.makeText(applicationContext, text: "SMS sent successfully", Toast.LENGTH_SHORT)
            .show()
    } catch (e: SecurityException) {
        Toast.makeText(
            applicationContext,
            text: "Permission denied to send SMS",
            Toast.LENGTH_SHORT
        ).show()
        e.printStackTrace()
    } catch (e: Exception) {
        Toast.makeText(
            applicationContext,
            text: "Failed to send SMS: ${e.message}",
            Toast.LENGTH_SHORT
        ).show()
        e.printStackTrace()
    }
}
new *
private fun startRecording() {
    val audioFilename = "Recording_${UUID.randomUUID()}.mp3"
    val audioFilePath =
        Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_MUSIC)?.absolutePath + "/" + audioFilename

    mediaRecorder = MediaRecorder().apply { this: MediaRecorder
        setAudioSource(MediaRecorder.AudioSource.MIC)
        setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP)
        setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB)
        setOutputFile(audioFilePath)
        try {
            prepare()
            start()
            isRecording = true
            Toast.makeText( context: this@SOSbutton, text: "Recording started", Toast.LENGTH_SHORT).show()

            Handler().postDelayed({
                stopRecording()
            }, delayMillis: 30000) // 30 seconds
        } catch (e: IOException) {
            Toast.makeText( context: this@SOSbutton, text: "Failed to start recording", Toast.LENGTH_SHORT)
                .show()
        }
```

Εικόνα 5.14 Μέρος υλοποίησης και ενεργοποίηση του SOSbutton

5.8 Εργαλείο Ρύθμισης Διεπαφών

Οι ρυθμίσεις της εφαρμογής βρισκονται στην κλάση **SplashActivity**. Εκεί ο χρήστης έχει την δυνατότητα να επεξεργαστεί διάφορα χαρακτηριστικά της εφαρμογής. Συγκεκριμένα μπορεί να ενεργοποιήσει τις φωνητικές ανακοινώσεις για άτομα με προβλήματα όρασεις, τις λειτουργίες περιστροφής της συσκευή για πλοήγηση της συσκευής από άτομα με προβλήματα όρασης, την

λειτουργία haptic του κινητού για δονήσεις που βοηθούν άτομα με προβλήματα ακοής και την ενεργοποίηση της ανίχνευσης κίνησης του τηλεφώνου (shake phone), για άτομα με οπτικές αναπηρίες. Επίσης, η δημιουργία της κλάσης καθιστά μια σημαντική βοήθεια στους προγραμματιστές, για την επέκταση της εφαρμογής ή την δημιουργία της δικής τους, καθώς παρέχει τα πιο βασικά εργαλεία πλοήγησης για ατομα με αισθητικές διαταραχές. Η κλάση εμφανίζεται έπειτα από κλίκ στο Activity της Εικόνας 3.16 απο τον χρήστη, έχοντας σαν αποτέλεσμα την μεταφορά του χρήστη στην διεπαφή της Εικόνας 5.15.

Στο xml αρχείο του **SplashActivity** υπάρχουν τα παρακάτω στοιχεία:

- **backbutton**: Επιστρεφει τον χρήστη στο menu.
- **shakebutton**: Ενεργοποιεί τη λειτουργία ανίχνευσης κίνησης της συσκευής (shake).
- **vibrateButton**: Ενεργοποιεί τη λειτουργία δόνηση.
- **rotatebutton**: Ενεργοποιεί τη λειτουργία περιστροφής της συσκευής.
- **announcebutton**: Ενεργοποιεί την λειτουργία ανακοινώσεων στις διεπαφές.

5.8.1 Βασικές Τεχνολογίες

TextToSpeech API : Χρησιμοποιείται για την προσομοίωση της ενεργοποίησης φωνητικών ανακοινώσεων, έχει αναλυθεί προηγουμένως.

VibrationEffect API: Χρησιμοποιείται για την προσομοίωση της ενεργοποίησης δονήσεων στην εφαρμογή, έχει αναλυθεί προηγουμένως.

onConfigurationChanged API: Όταν συμβαίνει αλλαγή στην περιστροφή της συσκευης (Portrait ↔ Landscape), το Android καταστρέφει και δημιουργεί ξανα την διεπαφή με τον νέο του προσανατολισμό (orientation), διαγράφοντας τα τρέχον δεδομένα της διεπαφής. Το συγκεκριμένο API καλείτε για να μην καταστρέφετε καθε φορα το Activity, με αποτέλεσμα χρήστης να διατηρεί την προοδο του στην διεπαφή.

Android Sensors (Accelerometer): Τεχνολογία που ανιχνεύει την επιτάχυνση της κίνησης στους άξονες x,y και z. Το **Accelerometer** είναι το πιο διαδεδομένο αισθητήριο εργαλείο σε κινητές συσκευές και αυτό το καθιστά κατάλληλο για την εφαρμογή.

5.8.2 Επεξήγηση της Κλάσης SplashActivity

Εκκίνηση:

- Ο χρήστης για να ξεκινήσει τις λειτουργίες του συστήματος πρέπει να πατήσει το αντίστοιχο κουμπί του κάθε εργαλείου κουμπί. Με την ενεργοποίηση κάθε εργαλείου δημιουργείται και ένα **checkmark**, υποδεικνύοντας στον χρήστη την ενεργοποίηση του εργαλείου.

Δόνηση (Vibration):

- Η λειτουργία γίνεται μέσω του **API VibrationEffect** για εκδόσεις Android 8.0 και νεότερες. Δημιουργεί δύο διαφορετικές δονησης:
    - **vibrateStrong()** ενεργοποιεί μια ισχυρή δόνηση (500ms, ένταση 220), για την κατανόηση από τον χρήστη της ενεργοποίησης της δόνησης.
    - **vibrateLow()** ενεργοποιεί μια αδύναμη δόνηση (500ms, ένταση 2), για την κατανόηση από τον χρήστη της απενεργοποίηση της δόνησης.

Ανακοινώσεις (TextToSpeech):

- Αρχικά ελέγχεται αν έχει οριστή σωστά το **TextToSpeech**.
- Αν έχει οριστεί σωστά και έχει πατηθεί το συγκεκριμένο κουμπί, ανακοινώνεται ένα μήνυμα ενεργοποίησης με το πέρασμα του μηνύματος “Announce Button Activate” στην μεταβλητή **TextToSpeech.** Αντίθετα περνάει στην μεταβλητή το string "Announce Button Deactivate".
    - **Στο TextToSpeech.QUEUE_FLUSH** το **QUEUE_FLUSH** διαγράφει οποιαδήποτε προηγούμενη εργασία στην ουρά και εκφωνεί μόνο το νέο μήνυμα. Εναλλακτικά, υπάρχει το **QUEUE_ADD**, το οποίο προσθέτει το νέο μήνυμα στην ουρά, συνεχίζοντας απο το προηγούμενο.
    - **Bundle** μπορεί να περιέχει πρόσθετες παραμέτρους για την εκφώνηση, όπως ο τόνος ή γλώσσα. Στην περίπτωση αυτή έχει την τιμή **NULL,** χρισημοποιόντας έτσι την **default** ομιλία της συσκευής.

Ανίχνευση Κίνησης (Shake Detection):

- Ο αλγόριθμος ανίχνευσης υπολογίζει την επιτάχυνση (**shake acceleration**) με βάση τις τιμές x, y, και z. Ελέγχει αν η επιτάχυνση ξεπερνά ένα όριο (15 m/s²) και αν έχει περάσει αρκετός χρόνος (800ms) από την τελευταία ανίχνευση.
    - **Sensor.TYPE_ACCELEROMETER** επιστρέφει σε πραγματικό χρόνο τις συντεταγμένες της συσκευης:
        - **event.values[0]**: Άξονα x (οριζόντια διάσταση).
        - **event.values[1]**: Άξονα y (κατακόρυφη διάσταση).
        - **event.values[2]**: Άξονα z (κάθετη διάσταση ή "βάθος").

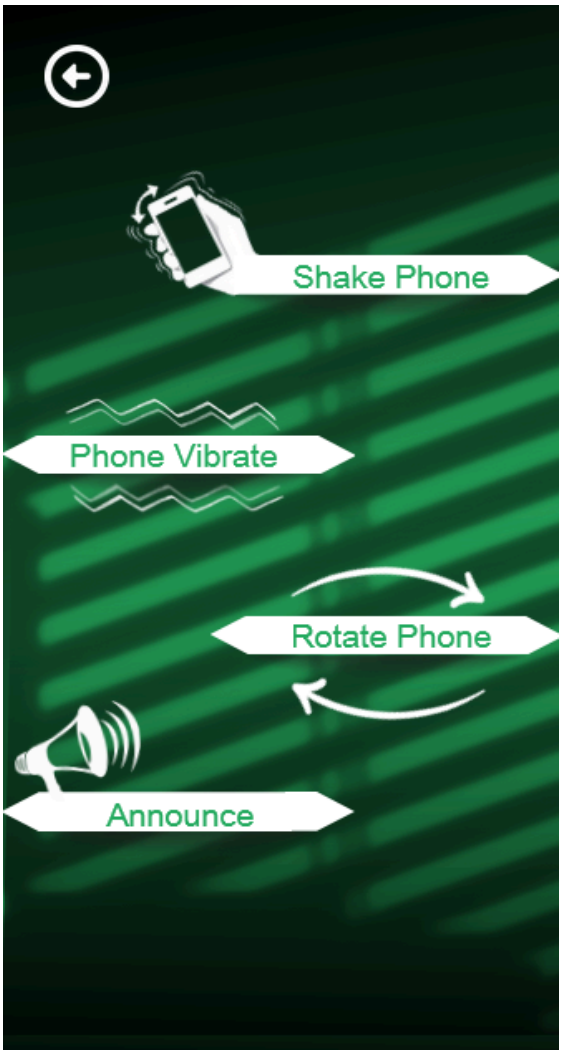
    - Υπολογίζεται η συνολική επιτάχυνση από τον τύπο :

$$\Sigma υνολική\ Επιτάχυνση(shakeAcceleration) = \sqrt{x2 + y2 + z2}$$

      με την βοήθεια της βιβλιοθηκης Math.

Ανίχνευση Περιστροφής (Rotation):

- **newConfig**: Ανιχνεύει την αλλαγή της περιστροφής του κινητού.
    - **ORIENTATION_LANDSCAPE**: Οθόνη σε οριζόντια διάταξη (landscape mode).
    - **ORIENTATION_PORTRAIT**: Οθόνη σε κάθετη διάταξη (portrait mode).
- **super.onConfigurationChanged**: Αναφέρεται στην υπερκλάση (superclass) της δραστηριότητας, δηλαδή την **AppCompatActivity** και χειρίζεται την διατήρηση της προόδου του χρήστη στην εφαρμογή.



```
private fun announceText(message: String) {
    textToSpeech.speak(message, TextToSpeech.QUEUE_FLUSH,  params: null,  utteranceId: null)
}

new *
override fun onSensorChanged(event: SensorEvent?) {
    if (event != null && event.sensor.type == Sensor.TYPE_ACCELEROMETER) {
        val x = event.values[0]
        val y = event.values[1]
        val z = event.values[2]

        val shakeAcceleration = Math.sqrt((x * x + y * y + z * z).toDouble())

        if (shakeAcceleration > 15) {
            val currentTime = System.currentTimeMillis()
            if (currentTime - lastShakeTimestamp > shakeThreshold) {
                showToast("Phone shaken!")

                val shakeButton = findViewById<Button>(R.id.shakebutton)

                if (shakeButton.text.contains( other: "✓")) {
                    removeCheckmark(shakeButton)
                } else {
                    addCheckmark(shakeButton)
                }

                lastShakeTimestamp = currentTime
            }
        }
    }
}
```

Εικόνα 5.15 Μέρος υλοποίησης και η διεπαφή SplashActivity

Κεφάλαιο 6ο: Εργαλείο Εκμάθησης Νοηματικής

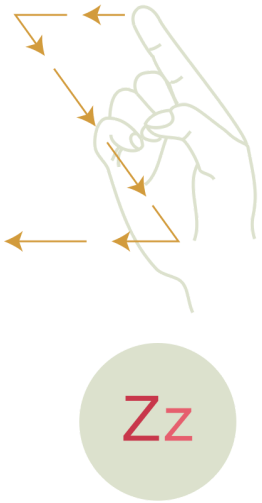
Η εκμάθησης νοηματικής αποτελεί το σημαντικότερο και το πιο πολυδιάστατο εργαλείο της εφαρμογης. Αφενός η χρήση του είναι εύκολη, αφετέρου ο τρόπος υλοποιησης του ήταν δυσνόητος και απαιτητικός. Γι αυτό λοιπόν το συγκεκριμένο κεφάλαιο θα αναλύσει εξονυχιστικά το τρόπο δημιουργίας του εργαλείου και την εξέλιξή του. Η εξήγηση του συγκεκριμένου εργαλείου και η κατανόηση του θα προμηθεύσει τους προγραμματιστές με γνώσεις για τα παρακάτω:

- Την επιλογή κατάλληλων τεχνολογιών και εργαλείων ανάλογα το πρότζεκτ .
- Την δημιουργία κατάλληλου συνόλου δεδομένων (dataset).
- Την δημιουργία μοντέλου τεχνητής νοημοσύνης για κινητές συσκευές.
- Την ενσωμάτωση του μοντέλου στην εφαρμογή και τις λειτουργίες του.

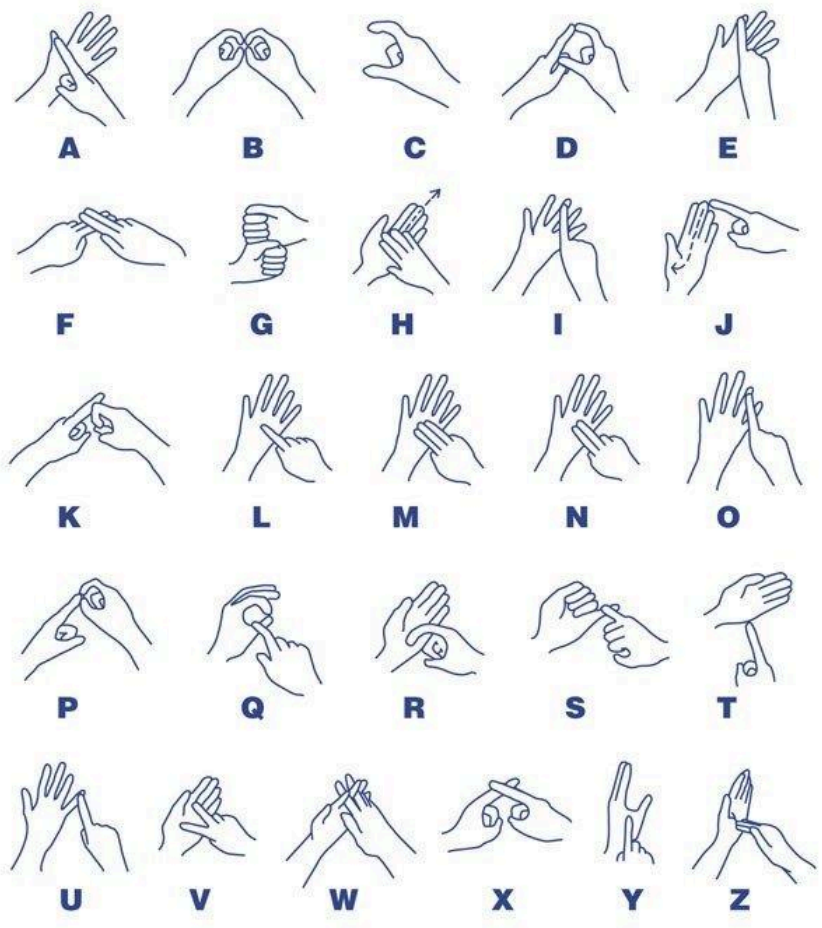
6.1 Πρώτα Βήματα

Για την επιλογή των κατάλληλων εργαλείων πρέπει πρώτα να αναλυθούν ποιοι είναι οι στόχοι του εργαλείου. Ο κύριος στόχος του εργαλείου είναι η εκμάθηση νοηματικής γλώσσας μέσω κάμερας σε πραγματικό χρόνο, χωρίς την ανάγκη πρόσβασης στο διαδίκτυο. Αναλυτικότερα ο χρήστης θα έχει την δυνατότητα να μάθει το αλφάβητο της γλώσσας χρισημοποιόντας τα χέρια του μπροστά από την κάμερα, σχηματίζοντας γράμματα του αλφάβητου και βλέποντας στην οθόνη του κινητού, σε πραγματικό χρόνο, την αναγνώρισή του γράμματος που σχημάτισε. Η επίτευξη του πρότζεκτ συνδέεται σε μεγάλο βαθμό την την επιλογή της γλώσσας.

Η επιλογή γλώσσας έχει μεγάλη σημασία στην αναπτυξη του εργαλείου, καθώς ανάλογα την γλώσσα πρέπει να ληφθούν υπόψιν οι αντίστοιχες δυσκολίες. Κάποια χαρακτηριστικά που πρέπει να αναφερθούν για την επιλογή γλώσσας είναι: αν χρειάζεται να δημιουργηθεί καινούργιο dataset, αν χρειάζεται να αναγνωρίζεται και η κίνηση του χεριού ως ξεχωριστη λειτουργια και αν χρειάζεται η χρήση και των δύο χεριών για την απεικόνιση του αλφάβητου. Με άλλα λόγια στη γλώσσα ASL (American Sign Language) για να σχηματίσει ο χρήστης το γράμμα “Z” πρέπει να κινήσει τον δείκτη του σε συγκεκριμένα σημεία (αναφορά στην Εικόνα 6.1), αντίστοιχα ο χρήστης για να σχηματίσει το BSL (British Sign Language) αλφάβητο επιβάλλεται να χρησιμοποιήσει και τα δύο του χέρια(αναφορά στην Εικόνα 6.2). Η δυνατότητα απεικόνισης του αλφάβητου της Ελληνικής Νοηματικής Γλώσσας (Greek Sign Language - GSL), χωρίς να είναι απαραίτητη η χρήση κινήσεων και η δυνατότητα να απεικονιστεί με ενα μονο χερι, την καθιστά κατάλληλη για την ανάπτυξη του πρότζεκτ. Επίσης υπάρχουν έτοιμα dataset για την GSL, παρόλα αυτα δημιουργήθηκε καινούργιο σύνολο δεδομένων για τις ειδικές ανάγκες του πρότζεκτ. Αυτό γιατί η το πρότζεκτ θα πρέπει τα τρέχει αυτόνομα σε κινητές συσκευές, χωρίς να χρησιμοποιεί μεγάλο αριθμό πόρω.



Εικόνα 6.1 Παράδειγμα γράμματος “Z” στην ASL



Εικόνα 6.2 Παράδειγμα γραμμάτων της BSL

6.2 Επιλογή Εργαλείων

Αρχικά είναι σημαντικό να αναφερθεί ότι το εργαλείο εκμάθησης νοηματικής γλώσσας φιλογένήτε σε μια νέα εφαρμογή με όνομα “Nyx”. Ήταν αναγκαία η δημιουργία μιας ακόμα εφαρμογής καθώς το εργαλείο πρέπει να τρέχει και σε πιο παλιες εκδοσεις Android, χρισημοποιόντας πολύ λιγότερους πόρους. Στην νεα εφαρμογή χρησιμοποιήθηκαν πιο παλιές εκδόσεις γλωσσών προγραμματισμού και βιβλιοθηκων για την μέγιστη συμπερίληψη κινητών συσκευων Android.

Τα κύρια εργαλεία που χρησιμοποιήθηκαν κατα την διάρκεια της ανάπτυξης της εφαρμογής είναι τα παρακάτω:

1) **Εργαλεία για Δημιουργία Dataset:**

Υπάρχουν διάφοροι τρόποι για την δημιουργία dataset. Κάποιοι από αυτούς είναι είναι με συλλογή φωτογραφιών από έτοιμα dataset, με χρήση Web Scraping και η δημιουργία dataset με χειροκίνητη εισαγωγή δεδομένων. Στην εφαρμογή χρησιμοποιήθηκε ο τρίτος τρόπος, αυτό έγινε δυνατό με το εργαλείο **Create hand dataset apk** [17]. Έτσι δημιουργήθηκε ένα καινούργιο dataset για να χρησιμοποιηθεί στην ανάπτυξη του μοντέλου τεχνητής νοημοσύνης.

2) **Εργαλεία για Ανάπτυξη μοντέλου τεχνητής νοημοσύνης:**

Python 3.9: Χρησιμοποιήθηκε για την δημιουργία μοντέλου τεχνητής νοημοσύνης. Παρέχει έναν μεγάλο αριθμό από βιβλιοθήκες που βοηθούν σημαντικά στην ανάπτυξη του μοντέλου:

a) **Numpy:** Υποστηρίζει τις μαθηματικές πράξεις σε πολυδιάστατους πίνακες.
b) **Pandas:** Βοηθάει την επεξεργασία και την ανάλυση δεδομένων. Ειδικότερα, προσφέρει εργαλεία για χειρισμό δεδομένων σε μορφή πινάκων (dataframes).
c) **Tensorflow:** Το TensorFlow είναι μια ειδικευμένη βιβλιοθήκη για μηχανική μάθηση. Χρησιμοποιείται κυρίως για εκπαίδευση και εξαγωγή συμπερασμάτων νευρωνικών δικτύων. Είναι ένα από τα πιο δημοφιλή πλαίσια βαθιάς μάθησης και θα αναλυθεί περισσότερο στην συνεχεια.
d) **Os:** Χρησιμοποιείται για το διάβασμα και την επεξεργασία αρχείων του συστήματος.
e) **Cv2:** Το **OpenCV** είναι βιβλιοθήκη για επεξεργασία και απόκτηση δεδομένων από εικόνες και βίντεο.
f) **Matplotlib.pyplot:** Δημιουργεί γραφήματα και χρησιμοποιείται για την οπτικοποίηση του μοντέλου.
g) **Tqdm:** Δημιουργεί progress bar και χρησιμοποιείται για την οπτικοποίηση της προόδου του μοντέλου.
h) **Scikit-learn:** Χρησιμοποιείται για την διαχείριση του dataset.
i) **Keras:** Υποστηρίζει την δημιουργία και την εκπαίδευση νευρωνικών δικτύων. Θα αναλυθεί παρακάτω ο λογος επιλογής του.
j) **Gc:** Επιτρέπει την αποδέσμευση μνήμης που δεν χρησιμοποιείται πλέον, βελτιώνοντας την απόδοση του μοντέλου.

3) **Εργαλεία για Ανάπτυξη Android εφαρμογή:**

   a) **Android.content.res.AssetFileDescriptor** και **android.content.res.AssetManager**: Χρησιμοποιούνται για τη φόρτωση αρχείων (όπως τα μοντέλα τεχνητής νοημοσύνης) από φακέλους στην εφαρμογή.
   b) **Android.graphics.Bitmap**: Για τη διαχείριση και την επεξεργασία εικόνων.
   c) **OpenCV version 3413:** Επεξεργάζεται και αντλεί δεδομένα από εικόνες και βίντεο.
   d) **TensorFlow Lite:** Χρησιμοποιείται για την εκτέλεση μοντέλων TensorFlow Lite, όπως το μοντέλο μηχανικής μάθησης σε συσκευές Android.
   e) **Java 8:** Σε αντίθεση με την εφαρμογή Aather που χρησιμοποιούσε kotlin, η Nyx χρησιμοποιεί **Java 8** για συμβατότητα με πιο παλιές συσκευές.

6.2.1 Kerras και Tensorflow

Το **TensorFlow** είναι μια ανοιχτού κώδικα βιβλιοθήκη για μηχανική μάθηση. Μπορεί να χρησιμοποιηθεί για πολύπλοκα μοντέλα όπως τα νευρωνικά δίκτυα και υποστηρίζει διάφορες γλώσσες προγραμματισμού, όπως Python, C++, και Java. Επίσης έχει την δυνατότητα να εκτελεστεί σε CPU, GPU, TPU (Tensor Processing Units), καθώς και σε κινητές συσκευές και browser. Ακόμη, παρέχει αυτόματη υπολογιστική διαφοροποίηση για την εκπαίδευση μοντέλων. Το σημαντικότερο του όμως χαρακτηριστικό είναι η η δυνατότητα μετατροπής μοντέλου σε **Tensorflow lite (TFLite)**. Το **TFLite** είναι μια ελαφριά έκδοση του **TensorFlow**, σχεδιασμένη ειδικά για τη χρήση μοντέλων μηχανικής μάθησης σε κινητές συσκευές με περιορισμένους πόρους.

Το **Keras** είναι μια υψηλού επιπέδου βιβλιοθήκη (high-level API) που αρχικά δημιουργήθηκε ως ανεξάρτητο εργαλείο, αλλά πλέον είναι ενσωματωμένο στο TensorFlow. Είναι σχεδιασμένο για να είναι φιλικό προς τον χρήστη και παρέχει εύχρηστες διεπαφές για τη δημιουργία, την εκπαίδευση, και τη δοκιμή νευρωνικών δικτύων. Επίσης υποστηρίζει προκατασκευασμένα μοντέλα, όπως το **EfficientNet**, για εξειδικευμένες εφαρμογές.

Οι βασικές διαφορές μεταξύ του **Kerras** και του **Tensorflow** αναφέρονται στον Πίνακα 6.2:

Χαρακτηριστικό	TensorFlow	Keras
Επίπεδο	Χαμηλού επιπέδου API	Υψηλού επιπέδου API
Αποδοτικότητα	Πλήρης λειτουργικότητα	Βελτιστοποιημένο για ταχύτητα και μέγεθος
Μέγεθος μοντέλου	Μεγαλύτερο μέγεθος	Μικρότερο μέσω κβαντοποίησης
Ευελιξία	Υψηλή	Περιορισμένη
Χρήση	Πιο σύνθετο, απαιτεί ρύθμιση για βασικές λειτουργίες.	Απλούστερο και φιλικό για αρχάριους.

Πίνακας 6.2 Διαφορές Kerras και Tensorflow

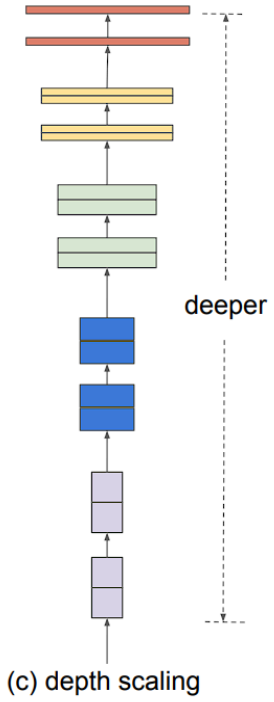
6.2.2 Επιλογή EfficientNet

Τα **CNN** (Συνελικτικά Νευρωνικά Δίκτυα) χρησιμοποιούνται για ανίχνευση αντικειμένων και ταξινόμηση εικόνων. Η ικανότητά τους να εκπαιδεύονται από εικόνες έχει οδηγήσει σε τεχνολογικές ανακαλύψεις στα αυτόνομα οχήματα, την ιατρική διάγνωση και την αναγνώριση προσώπου. Ωστόσο, καθώς αυξάνεται το μέγεθος και η πολυπλοκότητα των συνόλων δεδομένων, τα CNN γίνονται πιο σύνθετα για να διατηρούν υψηλή ακρίβεια. Η αύξηση της πολυπλοκότητας απαιτεί και περισσότερους υπολογιστικούς πόρους. Η απαίτηση της υψηλής υπολογιστικής δύναμης καθιστά τα CNN μη πρακτικά για εφαρμογές σε πραγματικό χρόνο και χρήση σε συσκευές με περιορισμένες δυνατότητες και πόρους, όπως η εφαρμογή Aather. Η κλιμάκωση του μοντέλου μπορεί να επιτευχθεί με τρεις τρόπους: αυξάνοντας το βάθος, το πλάτος ή την ανάλυση εικόνας του μοντέλου.

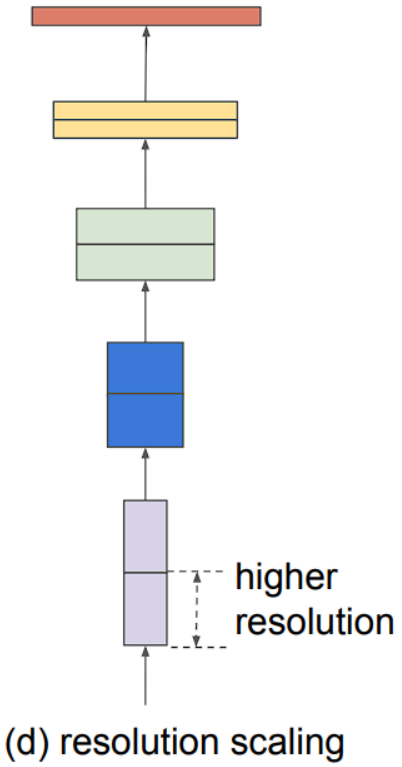
- Βάθος (c): Η κλιμάκωση του βάθους δικτύου είναι η πιο συχνά χρησιμοποιούμενη μέθοδος. Η ιδέα είναι απλή, το βαθύτερο **CNN** καταγράφει πιο σύνθετα χαρακτηριστικά και επίσης ομαδοποιεί καλύτερα (οπτικοποίηση στην Εικόνα 6.3).
- Ανάλυση εικόνας (d): Οι εικόνες με υψηλότερη ανάλυση επιτρέπουν στο μοντέλο να καταγράφει πιο λεπτομερείς μοτίβα. Τα νεότερα μοντέλα τείνουν να χρησιμοποιούν υψηλότερη ανάλυση. Ωστόσο, η υψηλότερη ανάλυση οδηγεί επίσης σε αυξημένες υπολογιστικές απαιτήσεις (οπτικοποίηση στην Εικόνα 6.4).
- Πλάτος (b): Χρησιμοποιείται σε μικρότερα μοντέλα. Η διεύρυνση ενός μοντέλου του επιτρέπει να καταγράφει πιο λεπτομερή χαρακτηριστικά (οπτικοποίηση στην Εικόνα 6.5).

Το γενικότερο πρόβλημα της προκύπτει από τους πολλούς παραμέτρους. Η αύξηση του βάθους ή του πλάτους οδηγεί σε σημαντική αύξηση του αριθμού των παραμέτρων στο δίκτυο. Κάθε παράμετρος απαιτεί υπολογισμό κατά τη διάρκεια της εκπαίδευσης και της πρόβλεψης. Περισσότεροι παράμετροι μεταφράζονται σε περισσότερους υπολογισμούς, αυξάνοντας τη συνολική υπολογιστική επιβάρυνση. Το EfficientNet στοχεύει στην επίλυση αυτού του προβλήματος, παρέχοντας έναν αποτελεσματικό και βιώσιμο τρόπο για την κλιμάκωση των CNN.[20]

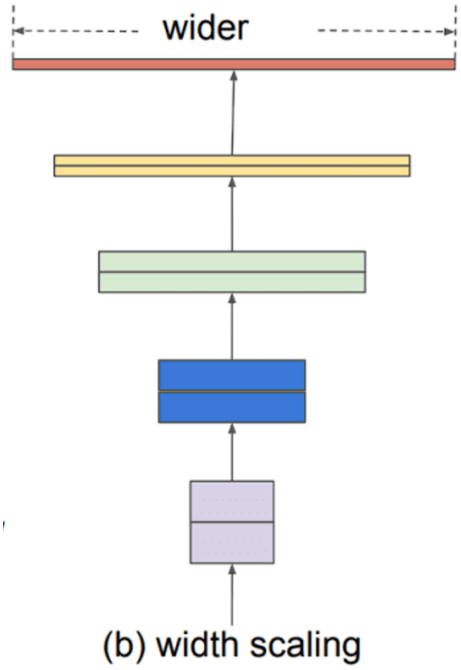
EfficientNet είναι μια οικογένεια μοντέλων νευρωνικών δικτύων βαθιάς μάθησης για ταξινόμηση εικόνων. Είναι γνωστό για την υψηλή του απόδοση στην ακρίβεια ταξινόμησης εικόνων και την αποδοτική χρήση υπολογιστικών πόρων. Χρησιμοποιεί μικρότερο αριθμό παραμέτρων και λιγότερη υπολογιστική δύναμη σε σχέση με άλλα μοντέλα (π.χ. ResNet) και έχει μια ενιαία στρατηγική για την κλιμάκωση των τριών κύριων διαστάσεων ενός νευρωνικού δικτύου: Πλάτος (Width), Βάθος (Depth) και Ανάλυση εικόνας (Image Resolution). Ο δημιουργός του **EfficientNet** παρατήρησε ότι οι διαφορετικές διαστάσεις κλιμάκωσης (βάθος, πλάτος, μέγεθος εικόνας) δεν είναι ανεξάρτητες. Οι εικόνες υψηλής ανάλυσης απαιτούν βαθύτερα δίκτυα για να καταγράψουν χαρακτηριστικά μεγάλης κλίμακας με περισσότερα pixel. Επιπλέον, απαιτούνται ευρύτερα δίκτυα για την καταγραφή των λεπτομερειών που υπάρχουν σε αυτές τις εικόνες υψηλής ανάλυσης. Σύμφωνα με την στρατηγική μέθοδο, για την επιδίωξη καλύτερης ακρίβειας και αποτελεσματικότητας, είναι κρίσιμο να εξισορροπηθούν όλες οι διαστάσεις του πλάτους, του βάθους και της ανάλυσης δικτύου. Η μέθοδος του συντελεστή σύνθετης κλίμακας κλιμακώνει ομοιόμορφα και τις τρεις διαστάσεις, με αναλογικό τρόπο χρησιμοποιώντας έναν προκαθορισμένο συντελεστή ένωσης ϕ.[21]



Εικόνα 6.3 Η κλιμάκωση του βάθους



Εικόνα 6.4 Η κλιμάκωση με Ανάλυση



Εικόνα 6.5 Η κλιμάκωση του Πλάτος

Με την αναλογική κλιμάκωση και των τριών διαστάσεων, το **EfficientNet** αποφεύγει τους περιορισμούς της κλιμάκωσης ενός άξονα. Οι προκαθορισμένοι συντελεστές επιτρέπουν τη δημιουργία μιας οικογένειας μοντέλων **EfficientNet** (**B0, B1, B2,** κ.λπ.) με ποικίλες χωρητικότητες. Κάθε μοντέλο προσφέρει μια διαφορετική αντιστάθμιση ακρίβειας-αποτελεσματικότητας, καθιστώντας τα κατάλληλα για διαφορετικές εφαρμογές. Σε σύγκριση με την παραδοσιακή κλιμάκωση, η σύνθετη κλιμάκωση επιτυγχάνει παρόμοια ή καλύτερη ακρίβεια με σημαντικά λιγότερα FLOPs και παραμέτρους (Λειτουργίες FLOating-point Per Second), καθιστώντας τις ιδανικές για συσκευές περιορισμένων πόρων.

Η ανάλυση κάθε **EfficientNet** μοντέλου μπορεί να παρατηρηθεί στον Πίνακα 6.3 και έχει δημιουργηθεί με με πληροφορίες από το keras documentation[18]:

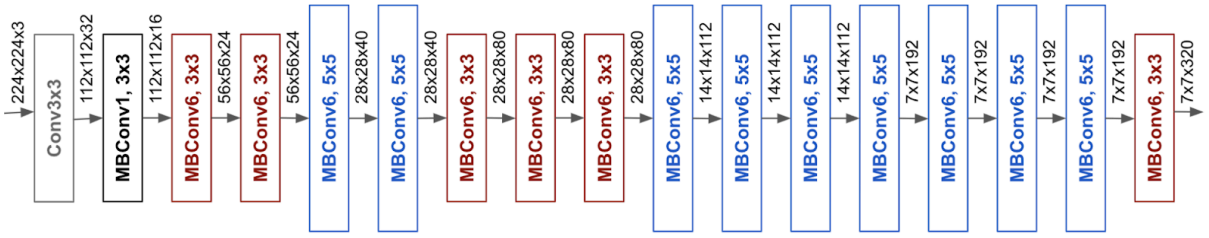
Μοντέλο	**Παράμετροι (M)**	**FLOPs (B)**	**Top-1 Ακρίβεια (%)**	**Μέγεθος Μοντέλου (MB)**	**Κατάλληλο για**
EfficientNetB0	5.3	0.39	77.1	16	Κινητές συσκευές
EfficientNetB1	7.8	0.7	79.1	26	Ελαφριές εφαρμογές σε cloud
EfficientNetB2	9.2	1.0	80.1	30	Mid-range συσκευές

EfficientNetB3	12.0	1.8	81.6	48	Mid-range/ισχυρά smartphones
EfficientNetB4	19.0	4.2	82.9	75	Υψηλών επιδόσεων συσκευές
EfficientNetB5	30.0	9.9	83.6	121	Ισχυρά laptops/servers
EfficientNetB6	43.0	19.0	84.0	208	High-end servers
EfficientNetB7	66.0	37.0	84.4	256	Εξειδικευμένα data centers

Πίνακας 6.3 EfficientNetBX Analysis

Απο τον πίνακα προκύπτει ότι το κατάλληλο μοντέλο για την δημιουργία της εφαρμογής είναι το **EfficientNetB0** και η αρχιτεκτονική του μπορεί να παρατηρηθεί στην Εικόνα 6.6**.**

Η ελαφριά αρχιτεκτονική του **B0**, το καθιστά ιδανικό για ανάπτυξη σε κινητές συσκευές και πλατφόρμες υπολογιστών αιχμής με περιορισμένους πόρους. Αυτό επιτρέπει στο **EfficientNet** να χρησιμοποιείται σε εφαρμογές που πραγματοποιούνται σε πραγματικό χρόνο, όπως η επαυξημένη πραγματικότητα,η αναγνώριση εικόνας και η εκτέλεση ανάλυσης βίντεο σε πραγματικό χρόνο.



Εικόνα 6.6 Η Αρχιτεκτονική του EfficientNetB0

Παρακάτω είναι ένας πίνακας που συγκρίνει τα βασικά χαρακτηριστικά του **EfficientNet**, **YOLO**, **MobileNet** και **SSD** (Single Shot Detector) για χρήση σε εφαρμογές τεχνητής νοημοσύνης, ο οποίος δικαιολογεί περεταίρω την επιλογή του **EfficientNet** (Πίνακα 6.4).

Χαρακτηριστικό	EfficientNet	YOLO	MobileNet	SSD
Ακρίβεια	Υψηλή ακρίβεια για ταξινόμηση και ανίχνευση	Υψηλή ακρίβεια για real-time ανίχνευση	Καλή ακρίβεια, αλλά χαμηλότερη από EfficientNetB0	Καλή ακρίβεια, αλλά εξαρτάται από την έκδοση

Μέγεθος Μοντέλου	Μικρό (4-5 MB), κατάλληλο για κινητές συσκευές	Μεγαλύτερο μέγεθος (>50 MB)	Πολύ μικρό (4-6 MB), ιδανικό για κινητές συσκευές	Μεσαίο (20-30 MB)
Απαιτήσεις Υπολογιστικής Ισχύος	Μέτριες, καλό scaling για περιορισμένους πόρους	Υψηλές, απαιτεί ισχυρές κάρτες γραφικών για ανίχνευση σε πραγματικό χρόνο	Πολύ χαμηλές, σχεδιασμένο για περιορισμένα περιβάλλοντα	Μέτριες, λιγότερες από YOLO αλλά περισσότερες από MobileNet
Ανίχνευση Νοηματικής Γλώσσας	Ιδανικό για ταξινόμηση και κατηγοριοποίηση χειρονομιών	Επικεντρώνεται στη real-time ανίχνευση αντικειμένων	Καλή απόδοση, αλλά περιορισμένη ακρίβεια σε περίπλοκα gestures (όπως αυτά της GSL)	Καλή ακρίβεια, αλλά ταχύτητα μικρότερη από YOLO
Κατανάλωση Ενέργειας	Χαμηλή κατανάλωση, βελτιστοποιημένο για κινητές συσκευές	Υψηλή κατανάλωση	Πολύ χαμηλή, κατάλληλο για χαμηλής ισχύος συσκευές	Μέτρια κατανάλωση
Επεκτασιμότητα	Υψηλή λόγω του compound scaling	Χαμηλή επεκτασιμότητα	Καλή επεκτασιμότητα	Καλή επεκτασιμότητα
Υποστήριξη για Τεχνολογίες Κινητών	Βελτιστοποιημένο για TensorFlow Lite	Δεν είναι βελτιστοποιημένο για κινητές συσκευές	Εξαιρετική ενσωμάτωση με TensorFlow Lite	Υποστηρίζεται από TensorFlow Lite, αλλά όχι τόσο ελαφρύ όσο το MobileNet
Ταχύτητα	Μέτρια ταχύτητα (ικανοποιητική για κινητές)	Πολύ γρήγορο	Πολύ γρήγορο	Γρήγορο αλλά συνήθως αργότερο από YOLO

Πίνακας 6.4 Χαρακτηριστικά EfficientNetB, YOLO, MobileNet και SSD

Το EfficientNetB0 είναι η καλύτερη επιλογή για την ανίχνευση νοηματικής γλώσσας σε κινητές συσκευές, λόγω της υψηλής της ακρίβειας, χαμηλής κατανάλωσης πόρων και εύκολης ενσωμάτωσης σε εφαρμογές κινητών συσκευών.

6.3 Υλοποίηση στον Υπολογιστή

Αρχικά, πριν γίνει η υλοποίηση του εργαλείου στο Android studio, υλοποιήθηκε στο Visual Studio Code με την γλώσσα Python. Αυτό γιατί πρέπει είναι ένας ωραίος τρόπος για εξοικείωση του με τα εργαλεία και τις τεχνολογίες που θα χρησιμοποιηθούν στην συνέχεια για την δημιουργία του εργαλείου σε κινητές συσκευές. Πιο συγκεκριμένα, εξασκήθηκαν τεχνικές μηχανικής μάθησης, δημιουργίας συνόλου δεδομένων, εξοικείωση με εργαλεία όπως το OpenCV, το mediapipe και άλλες.

Για να κατασκευαστεί το εργαλείο χρειάστηκαν τα παρακάτω Python αρχεία:

- collect_imgs: Συλλογή Εικόνων (PC)
- create_dataset: Δημιουργία Dataset (PC)
- train_classifier: Εκπαίδευση (PC)
- inference_classifier: Ομαδοποίηση αποτελεσμάτων (PC)

6.3.1 Συλλογή Εικόνων (PC)

Για την δημιουργία του Dataset, εκμεταλλεύτηκε η τεχνική μετατροπής frames (στιγμιότυπα) σε εικόνες (jpg). Οι διαστάσεις τους είναι 640px πλάτους και 480px ύψους. Περιέχονται περίπου 500 φωτογραφίες για κάθε ένα γράμμα της GSL, ο αριθμός αυτος θεωρείται ένας αξιόλογος αριθμός για την ομαδοποίηση των εικόνων. Πιο συγκεκριμένα :

- Δημιουργεί τον κύριο φάκελο αποθήκευσης **./data**, αν αυτός δεν υπάρχει ήδη.
- **number_of_classes = 24** → Καθορίζει πόσες θα είναι οι κλάσεις. Είναι 24 όσα και τα γράμματα της ελληνικής γλώσσας.
- **dataset_size = 500** → Ορίζει το πόσες φωτογραφίες να τραβήξει για το κάθε γράμμα.
- Ελέγχεται η προσβασιμότητα στην κάμερα και εμφανίζει μήνυμα Error αν δεν πάει κάτι σωστα.
- Γίνεται προετοιμασία φακέλων για κάθε κατηγορία, δημιουργώντας έναν υποφάκελο μέσα στο **DATA_DIR** για κάθε κατηγορία (π.χ., ./data/0, ./data/1, ./data/2)
- Ενεργοποιείται η κάμερα του υπολογιστή, και ο αριθμός 2 δηλώνει τον δείκτη της κάμερας. Αν δεν δουλέυει μπορούν να δοκιμαστούν οι αριθμοί 0 και 1.
- Έπειτα για κάθε κατηγορία ο χρήστης βλέπει ένα μήνυμα που τον ενημερώνει ότι τα δεδομένα για την κατηγορία θα ξεκινήσουν μόλις πατήσει το πλήκτρο **'q'**. Έτσι ο χρήστης έχει τον έλεγχο του χρονικού διαστήματος μεταξύ των καταγραφών.
- Οι εικόνες συλλέγονται από την κάμερα και αποθηκεύονται στον αντίστοιχο φάκελο της κατηγορίας με ονόματα αρχείων σαν και αυτά: 0.jpg, 1.jpg, ..., 499.jpg.
- Μετά την ολοκλήρωση όλων των γραμμάτων η καταγραφή ολοκληρώνεται και το παράθυρο κλείνει.

Ο κωδικας μπορεί να παρατηρηθεί στην Εικόνα 6.7

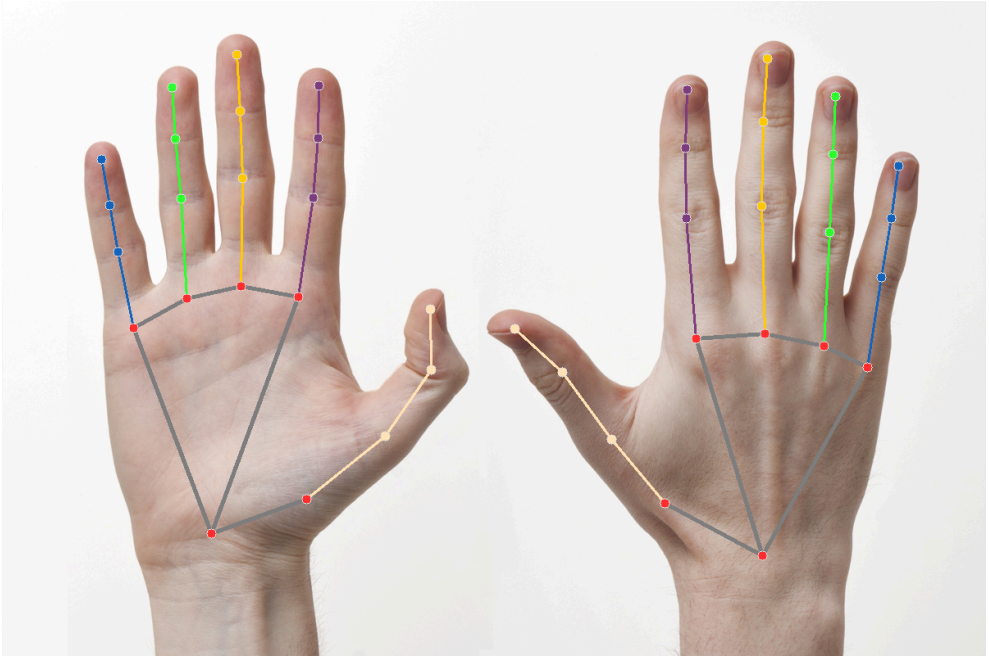
```python
import os
import cv2

DATA_DIR = r'D:\pythaitry\code\imagetraining'
if not os.path.exists(DATA_DIR):
    os.makedirs(DATA_DIR)
number_of_classes = 24
dataset_size = 500
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Error: Cannot open the camera.")
    exit()

for j in range(number_of_classes):
    class_dir = os.path.join(DATA_DIR, str(j))
    if not os.path.exists(class_dir):
        os.makedirs(class_dir)

    print(f'Collecting data for class {j}')

    while True:
        ret, frame = cap.read()
        if not ret:
            print("Error: Failed to read frame.")
            break
        cv2.putText(frame, 'Ready? Press "Q" to start :)', (100, 50),
                    cv2.FONT_HERSHEY_SIMPLEX, 1.3, (0, 255, 0), 3, cv2.LINE_AA)
        cv2.imshow('frame', frame)
        if cv2.waitKey(25) == ord('q'):
            break
    counter = 0
    while counter < dataset_size:
        ret, frame = cap.read()
        if not ret:
            print("Error: Failed to read frame.")
            continue  # Skip this iteration if frame read fails
        cv2.imshow('frame', frame)
        cv2.waitKey(25)
        cv2.imwrite(os.path.join(class_dir, f'{counter}.jpg'), frame)
        counter += 1

cap.release()
cv2.destroyAllWindows()
```

Εικόνα 6.7 Υλοποίηση του collect_imgs

6.3.2 Δημιουργία Dataset (PC)

Η δημιουργία του dataset έχει να κάνει με την μετατροπή των εικόνων που συλλέχθηκαν σε πληροφορία. Η κλάση ανιχνεύει την φιγούρα του χεριού και στην συνέχεια χαράζει άξονες για την ανάλυση των πληροφοριών τοποθεσίας των άκρων. Αυτο το βήμα είναι κρίσιμο, καθώς προετοιμάζει τα δεδομένα για την εκπαίδευση του μοντέλου μηχανικής μάθησης, εξάγοντας τα χαρακτηριστικά των εικόνων χρησιμοποιώντας τη βιβλιοθήκη **Mediapipe.** Στη συνέχεια, αποθηκεύει τα δεδομένα και τις αντίστοιχες ετικέτες σε ένα αρχείο **pickle**. Αναλυτικά:

- Αρχικά ορίζονται τα χαρακτηριστικά του **Mediapipe**:
    - **mp.solutions.hands**: Φορτώνει το μοντέλο Mediapipe για την ανίχνευση του χεριού.
    - **mp_drawing**: Εργαλείο για να σχεδιάζει τα ανιχνευμένα σημεία και τον τρόπο σύνδεσής τους πάνω στην εικόνα (οπτικοποίηση της λειτουργίας στην Εικόνα 6.8).



Εικόνα 6.8 Landmarks για την αναφορά σημείων στην μηχανική μάθηση

    - **static_image_mode=True**: Καθορίζει ότι η είσοδος είναι στατική εικόνα και όχι βίντεο, καθώς έχει ήδη μετατραπεί από βίντεο σε εικόνες.
    - **min_detection_confidence=0.3**: Δηλώνει την ελαχιστη εμπιστοσύνη για την ανίχνευση χεριών. Άρα κατα πάσα πιθανότητα η φωτογραφία που υπάρχει στο dataset περιέχει χέρι.
- **DATA_DIR**: Ο φάκελος που περιέχει το dataset.
- **data**: Λίστα στην οποία αποθηκεύονται τα χαρακτηριστικά κάθε εικόνας (όπως τα landmarks).
- **labels**: Λίστα όπου αποθηκεύονται οι ετικέτες ομάδας κάθε εικόνας.
- Για κάθε φάκελο **dir_** (που αντιπροσωπεύει μια κατηγορία) εξετάζει κάθε εικόνα **img_path** μέσα στον αντίστοιχο φάκελο.
    - **cv2.imread**: Διαβάζει την εικόνα.
    - **cv2.cvtColor**: Μετατρέπει την εικόνα από BGR (που είναι η προεπιλεγμένη μορφή του OpenCV) σε RGB.

- **hands.process**: Επεξεργάζεται την εικόνα για να εντοπίσει το χέρι.
- Αν ανιχνευτεί παραπάνω από ένα χέρι, τότε εξετάζει κάθε ανιχνεύσιμο χέρι ξεχωριστά.
  - Αρχικά, συλλέγει τις **x** και **y** συντεταγμένες κάθε σημείου (landmark) και τις αποθηκεύει στις λίστες **x_** και **y_**.
  - Στη συνέχεια, υπολογίζει τις σχετικές συντεταγμένες κάθε σημείου με βάση το ελάχιστο **x** και **y**. Αυτό βοηθά στη γενίκευση των δεδομένων και στην ανεξαρτησία από τη θέση του χεριού.
  - Εάν έχουν συλλεχθεί 42 χαρακτηριστικά (21 σημεία x και 21 y), τα χαρακτηριστικά προστίθενται στη λίστα **data** και η κατηγορία στη λίστα **labels**.

- Με το **pickle** αποθηκεύονται τα δεδομένα που καταγράφηκαν στους προηγούμενους πίνακες στο αρχείο **data.pickle**.

Ο κωδικας μπορεί να παρατηρηθεί στην Εικόνα 6.9

```
import os
import pickle
import mediapipe as mp
import cv2
import matplotlib.pyplot as plt

mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
mp_drawing_styles = mp.solutions.drawing_styles
hands = mp_hands.Hands(static_image_mode=True, min_detection_confidence=0.3)
DATA_DIR = 'D:/pythaitry/code/imagetraining'
data = []
labels = []
for dir_ in os.listdir(DATA_DIR):
    for img_path in os.listdir(os.path.join(DATA_DIR, dir_)):
        data_aux = []
        x_ = []
        y_ = []
        img = cv2.imread(os.path.join(DATA_DIR, dir_, img_path))
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        results = hands.process(img_rgb)
        if results.multi_hand_landmarks:
            for hand_landmarks in results.multi_hand_landmarks:
                for i in range(len(hand_landmarks.landmark)):
                    x = hand_landmarks.landmark[i].x
                    y = hand_landmarks.landmark[i].y
                    x_.append(x)
                    y_.append(y)
                for i in range(len(hand_landmarks.landmark)):
                    x = hand_landmarks.landmark[i].x
                    y = hand_landmarks.landmark[i].y
                    data_aux.append(x - min(x_))
                    data_aux.append(y - min(y_))
            if (len(data_aux) == 42):
                print (f"Data class: {dir_}")
                data.append(data_aux)
                labels.append(dir_)
f = open('data.pickle', 'wb')
pickle.dump({'data': data, 'labels': labels}, f)
f.close()
```

Εικόνα 6.9 Υλοποίηση του create_dataset

6.3.3 Εκπαίδευση (PC)

Η παραπάνω κλάση υλοποιεί την εκπαίδευση ενός μοντέλου μηχανικής μάθησης, συγκεκριμένα του **Random Forest Classifier**, για την ταξινόμηση δεδομένων που προέρχονται από χειρονομίες χεριών.

Η επιλογή του **Random Forest Classifier** βασίστηκε στην ταχύτητα εκπαίδευσης και στην αρκετά απλή ενσωμάτωσης του στον κώδικα. Επειδή όπως αναλύθηκε προηγουμένως το **Mediapipe** θα αντικατασταθεί από το εξειδικευμένο μοντέλο **EfficientNetB0**, ήταν περιττή ανάπτυξης λογικής μοντέλου για μηχανικής μάθησης. Η **Random Forest** παρέχει μια γρηγορη και εύκολη λύση για την δοκιμή του dataset και την εξάσκηση στα εργαλεία του **OpenCV**. Παρακάτω αναλύονται τα βήματα του κώδικα:

- Αρχικά η **pickle.load** φορτώνει τα δεδομένα από το αρχείο **data.pickle**. Το αρχείο αυτό περιέχει τα χαρακτηριστικά (data) και τις αντίστοιχες ετικέτες (labels) από το προηγούμενο στάδιο.
- **np.asarray**: Μετατρέπει τις λίστες σε πίνακες **NumPy** για καλύτερη απόδοση του μοντέλου.
- **train_test_split**: Διαχωρίζει τα δεδομένα σε σύνολα εκπαίδευσης (**x_train, y_train**) και σύνολα δοκιμής (**x_test, y_test**). Με το **test_size=0.2** καθορίζεται ότι το **20%** των δεδομένων χρησιμοποιείται για δοκιμή (**testing**) ενώ το υπόλοιπο **80%** για την εκπαίδευση. Ο διαχωρισμός αυτός γίνεται για την σωστή αξιολόγηση του μοντέλου, καθώς αν η δοκιμή του μοντέλου έτρεχε σε σε εικόνες που έχει ήδη εκπαιδευτεί, τα αποτελέσματα δεν θα ήταν έγκυρα. Το **shuffle=True** ανακατεύει τα δεδομένα πριν από τον διαχωρισμό και το **stratify=labels** διασφαλίζει ότι η κατανομή των κλάσεων παραμένει ίδια στα σύνολα εκπαίδευσης και δοκιμής.
- Έπειτα δημιουργείται ένα μοντέλο **Random Forest** με τις προεπιλεγμένες παραμέτρους. Τα σημαντικότερα χαρακτηριστικά του **Random Forest** είναι ότι δημιουργεί πολλά δέντρα αποφάσεων και συνδυάζει τις προβλέψεις των δέντρων για να δώσει την τελική πρόβλεψη.
- Το μοντέλο εκπαιδεύεται χρησιμοποιώντας τα δεδομένα **x_train** και **y_train**
- **model.predict(x_test):** Το μοντέλο προβλέπει τις ετικέτες για τα δεδομένα δοκιμής (x_test).
- **accuracy_score(y_predict, y_test)**: Υπολογίζει την ακρίβεια των προβλέψεων συγκρίνοντας τις προβλεπόμενες ετικέτες (**y_predict**) με τις πραγματικές (**y_test**).
- Τέλος, εμφανίζει το ποσότητα επιτυχίας σε ποσοστό.

Ο κωδικας μπορεί να παρατηρηθεί στην Εικόνα 6.10

```python
import pickle

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import numpy as np


data_dict = pickle.load(open('D:/pythaitry/code/data.pickle','rb'))

data = np.asarray(data_dict['data'])
labels = np.asarray(data_dict['labels'])

x_train, x_test, y_train, y_test = train_test_split(
    data, labels, test_size=0.2, shuffle=True, stratify=labels
    )

model = RandomForestClassifier()

model.fit(x_train, y_train)

y_predict = model.predict(x_test)

score = accuracy_score(y_predict, y_test)

print('{}% of samples were classified correctly!'.format(score * 100))

f = open('model.p', 'wb')
pickle.dump({'model': model}, f)
f.close()
```

Εικόνα 6.10 Υλοποίηση του train_classifier

6.3.4 Ομαδοποίηση αποτελεσμάτων (PC)

Η κλάση αυτή αφορά τη ζωντανή αναγνώριση χειρονομιών μέσω κάμερας, χρησιμοποιώντας το εκπαιδευμένο μοντέλο μηχανικής μάθησης που δημιουργήθηκε, για την πρόβλεψη της ελληνικής νοηματικής. Τα βήματα:

- Αρχικά, το εκπαιδευμένο μοντέλο φορτώνεται από το αρχείο **model.p**.
- Στην συνέχεια ανοίγει η κάμερα με την παράμετρο 0.
- Ορίζονται τα χαρακτηριστικά του MediaPipe όπως έγινε και στην δημιουργία του dataset:
    - Ελάχιστη εμπιστοσύνη 0.3.

    - **mp_drawing**: Χρησιμοποιείται για την απεικόνιση των landmarks (οπτικοποίηση της λειτουργίας στην Εικόνα 6.7).
- **labels_dict**: Αντιστοιχεί τους αριθμούς εξόδου που προκύπτουν από το μοντέλο σε ελληνικούς χαρακτήρες.
- **ImageFont.truetype**: Ρυθμίζει τη γραμματοσειρά για εμφάνιση ελληνικών χαρακτήρων.
- Ξεκινά η επανάληψη η οποία εκτελείται συνεχώς έως ότου ο χρήστης πατήσει το πλήκτρο **q**.
    - Λαμβάνει ένα καρέ από την κάμερα και ελέγχει αν αυτό διαβάστηκε επιτυχώς.
    - Μετατρέπει το καρέ σε μορφή RGB για το MediaPipe και ανιχνεύει τα landmarks των χεριών.
    - Εξάγει τις **x** και **y** συντεταγμένες των 42 συνολικά σημείων (**landmarks**) και κανονικοποιεί τις τιμές τους.
    - Το μοντέλο προβλέπει την τάξη και ο χαρακτήρας αντιστοιχίζεται μέσω του **labels_dict** για την σωστή απεικόνιση του μοντέλου.
    - Σχεδιάζετε με το **OpenCV** ένα ορθογώνιο γύρω από το χέρι και εμφανίζει τον προβλεπόμενο χαρακτήρα.
- Όταν πατηθεί το **q** σταματάει η κάμερα και κλείνουν όλα τα παράθυρα.

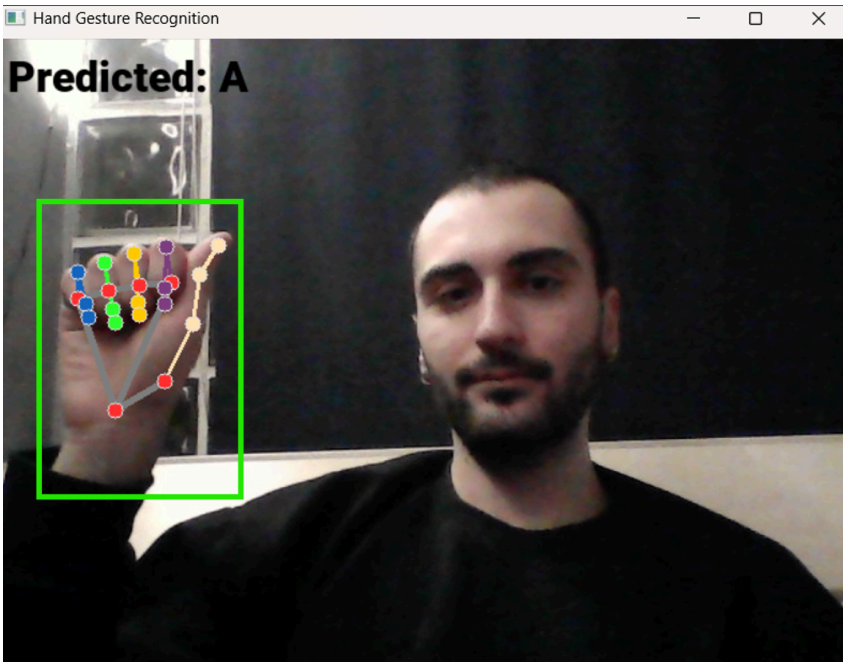
Ο κωδικας μπορεί να παρατηρηθεί στην Εικόνα 6.11

```
import pickle
import cv2
import mediapipe as mp
import numpy as np
from PIL import ImageFont, ImageDraw, Image

model_dict = pickle.load(open('D:/pythaitry/code/model.p', 'rb'))
model = model_dict['model']
cap = cv2.VideoCapture(0)
mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
mp_drawing_styles = mp.solutions.drawing_styles
hands = mp_hands.Hands(static_image_mode=True, min_detection_confidence=0.3)
labels_dict = {0: 'Α', 1: 'Β', 2: 'Γ', 3: 'Δ', 4: 'Ε', 5: 'Ζ',6: 'Η', 7: 'Θ',
               8: 'Ι',9: 'Κ', 10: 'Λ', 11: 'Μ',12: 'Ν', 13: 'Ξ', 14: 'Ο',15: 'Π',
               16: 'Ρ', 17: 'Σ',18: 'Τ', 19: 'Υ', 20: 'Φ',21: 'Χ', 22: 'Ψ', 23: 'Ω'}
font_path = 'D:\pythaitry\code\Roboto-Black.ttf'
font_size = 32
font = ImageFont.truetype(font_path, font_size)
while True:
    data_aux = []
    x_ = []
    y_ = []
    ret, frame = cap.read()
    if not ret:
        break
    H, W, _ = frame.shape
    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    results = hands.process(frame_rgb)
    if results.multi_hand_landmarks:
        hand_landmarks = results.multi_hand_landmarks[0]
        mp_drawing.draw_landmarks(
            frame,
            hand_landmarks,
            mp_hands.HAND_CONNECTIONS,
            mp_drawing_styles.get_default_hand_landmarks_style(),
            mp_drawing_styles.get_default_hand_connections_style())
        for i in range(len(hand_landmarks.landmark)):
            x = hand_landmarks.landmark[i].x
            y = hand_landmarks.landmark[i].y
            x_.append(x)
            y_.append(y)
```

```
        for i in range(len(hand_landmarks.landmark)):
            x = hand_landmarks.landmark[i].x
            y = hand_landmarks.landmark[i].y
            data_aux.append(x - min(x_))
            data_aux.append(y - min(y_))
        x1 = int(min(x_) * W) - 10
        y1 = int(min(y_) * H) - 10
        x2 = int(max(x_) * W) - 10
        y2 = int(max(y_) * H) - 10
        if len(data_aux) == 42:
            prediction = model.predict([np.asarray(data_aux)])
            predicted_character = labels_dict[int(prediction[0])]
            cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 0, 0), 4)
            image_pil = Image.fromarray(frame)
            draw = ImageDraw.Draw(image_pil)
            draw.text((x1, y1 - 10), predicted_character, font=font, fill=(0, 0, 0))
            frame = np.array(image_pil)
    cv2.imshow('frame', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

Εικόνα 6.11 Υλοποίηση του inference_classifier

Ένα παράδειγμα αναγνώρισης της ελληνικής νοηματικής εμφανίζεται στην Εικόνα 6.12. Η συγκεκριμένη προσπάθεια συνέλαβε στην ανάπτυξη της λογικής, η οποία θα χρησιμοποιηθεί για την κατασκευή του εργαλείου σε κινητές συσκευές.



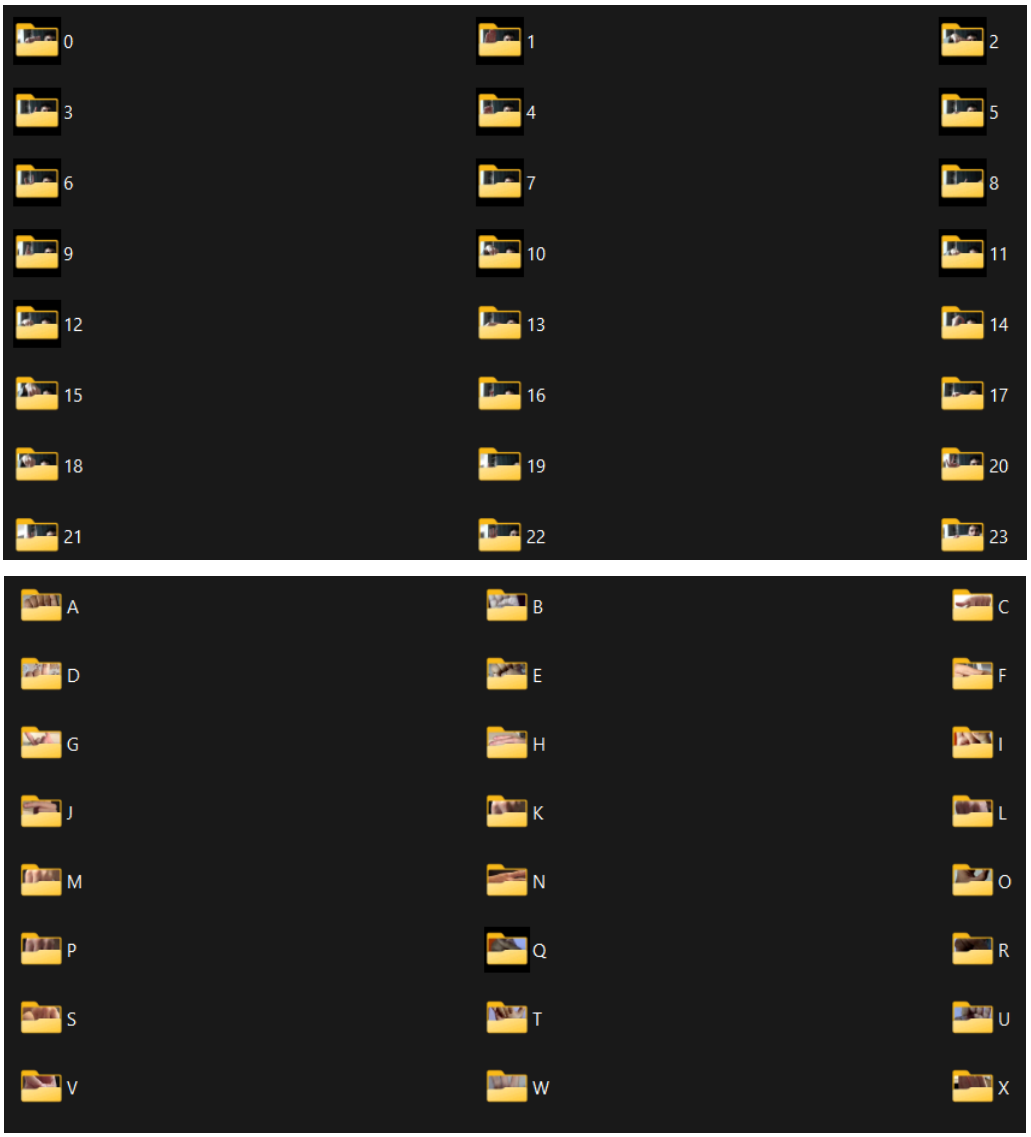
Εικόνα 6.12 Παράδειγμα υλοποίησης σε υπολογιστικό περιβάλλον

6.4 Δημιουργία Dataset

Για την δημιουργία ενός αξιόπιστο μοντέλου Ai χρειάζεται ένα μεγάλο dataset. Όσο μεγαλύτερο τόσο πιο αποδοτικό θα είναι το μοντέλο. Η δημιουργία dataset απο την αρχή είναι αρκετά χρονοβόρα, καθώς χρειάζεται να τραβηχτουν ή να συλλεχθούν δεκάδες χιλιάδες φωτογραφίες.Αυτός είναι και ο λόγος της χρήσης του **Create hand dataset apk** που αναφέρθηκε και προηγουμένως. Η συγκεκριμένη Android εφαρμογή κάνει αρκετά πράγματα που είναι αξιοσημείωτο. Αρχικά χρησιμοποιεί την κάμερα για λήψη βίντεο, ώστε να χωριστούν τα στιγμιότυπα (frames) και να αποθηκευτούν ως αρχεια εικόνων (jpg).

Επίσης, με το μοντέλο **hand_model** και την βιβλιοθήκη **OpenCV** αναγνωρίζει το χέρι και διαμορφώνει τις διαστάσεις, με σκοπό να αποτυπώνεται στην φωτογραφία μόνο το χέρι. Αυτό γίνεται για δύο κύριους λόγους. Ο πρώτος είναι ότι με το **crop** της φωτογραφίας, περιοριζει τις περιττές πληροφορίες που θα περιέχει η εικόνα, όπως πρόσωπα και αντικείμενα, έτσι το τελικό μοντέλο θα είναι αρκετά πιο αξιόπιστο. Ο δεύτερος κυριος λόγος είναι η δημιουργία μικρότερου σε μέγεθος μοντέλου. Κατά την ανάπτυξη του πρότζεκτ χρησιμοποιήθηκε δύο τρόποι δημιουργίας dataset, με περικοπή και χωρίς περικοπή. Το μοντέλο με περικοπή ήταν σημαντικά μικρότερο από το μοντέλο χωρίς περικοπή. Στην Εικόνα 6.13, οι φάκελοι με αριθμητικά ονόματα αποτελούν το dataset χωρίς περικοπή και ο κατάλογος έχει μέγεθος **814** megabytes για **11.997** φωτογραφίες. Αντίστηχα οι φάκελοι με τα Αγγλικά γράμματα είναι το dataset με περικοπή και έχει μέγεθος **342** megabytes για **12.495** φωτογραφίες.

Τέλος η εφαρμογή προσφέρει την δυνατότητα ονομασίας φακέλου σε κάθε νέα εγγραφή βίντεο. Ετσι ο δημιουργός δεν χρειάζεται να χωρίζει συνέχεια τις φωτογραφίες, καθώς εντάσσονται στον προκαθορισμένο φάκελο. Συνολικά συλλέχθηκαν 24.492 φωτογραφίες, από τις οποίες χρησιμοποιήθηκαν οι **12.495.**



Type:	File folder
Location:	D:\pythaitry\code
Size:	814 MB (854,063,808 bytes)
Size on disk:	837 MB (877,936,640 bytes)
Contains:	11,977 Files, 24 Folders

Type:	File folder
Location:	D:\pythaitry\code
Size:	342 MB (359,511,882 bytes)
Size on disk:	367 MB (385,060,864 bytes)
Contains:	12,495 Files, 24 Folders

Εικόνα 6.13 Dataset με περικοπή και χωρίς

6.5 Δημιουργία Μοντέλου Ai

Για την εκπαίδευση του μοντέλου κατασκευάστηκε το αρχείο **sign_language_recognition.py**. Για την εφαρμογή δημιουργήθηκαν 5 διαφορετικά μοντέλα και χρησιμοποιήθηκε αυτό με τις καλύτερες αποδόσεις. Η λογική ροή που ακολουθήθηκε είναι: Προετοιμασία δεδομένων → Διαχωρισμός Συνόλου Δεδομένων → Κατασκευή Μοντέλου → Εκπαίδευση Μοντέλου → Μετατροπή Μοντέλου. Παρακάτω περιγράφεται μερικές επιπλέον λεπτομέρειες για κάθε σημαντικό σημείο του κώδικα:

- Εισαγωγή Βιβλιοθηκών **numpy, pandas, tensorflow, os, cv2, matplotlib.pyplot, tqdm, keras και sklearn.model_selection.**
    - **NumPy: 1.26.4**
    - **Pandas: 2.2.3**
    - **TensorFlow: 2.18.0**
    - **OpenCV: 4.10.0**
    - **Matplotlib: 3.9.3**
    - **Tqdm: 4.67.1**
    - **Scikit-learn: 1.5.2**
    - **Scikit-learn: 1.5.2**
    - **Keras: 3.7.0**
- Προσθήκη στο **path** η διαδρομή στο οποίο βρίσκεται το dataset που δημιουργήθηκε προηγουμένως.
- Χρήση του **os.listdir()** για να την εισαγωγή όλων των αρχείων και η **sort()** για να την ταξινομηση τους.
- Δημιουργία κενών λιστών για την αποθήκευση των εικόνων και των ετικετών τους.
- Με το **OpenCV** και την εντολή **cv2.imread()** διαβάζεται κάθε εικόνα και γίνεται μετατροπή της ποιότητας σε 96x96 pixels, καθώς και μετατροπή από **BGR** σε **RGB**. Τα αποτελέσματα Αποθηκεύονται στους στις κενές λίστες που δημιουργήθηκαν.
- Μετατροπή των λιστών σε **NumPy arrays** για καλύτερη απόδοση στην εκπαίδευση του μοντέλου (Εικόνα 6.14).
- Χρησιμοποιείται η **train_test_split()** από τη βιβλιοθήκη **sklearn,** για να διαχωριστούν τα δεδομένα σε αυτά που θα χρησιμοποιηθούν για την εκπαίδευση του μοντέλου και αυτά που θα χρησιμοποιηθούν για την δοκιμή του μοντέλου (85% και 15% αντίστοιχα).
- Με το **Sequential()** δημιουργείται το μοντέλο, προσθέτοντας το **pretrained** μοντέλο **EfficientNetB0**.
- Προστίθεται ένα **GlobalAveragePooling2D** για να μειώσει τις διαστάσεις της εξόδου του προεκπαιδευμένου μοντέλου και για να γενικευθεί το μοντέλο.
- Χρήση του **Data Augmentation** για παραγωγή περισσότερων δεδομένων εκπαίδευσης μέσω μετασχηματισμών στις υπάρχουσες εικόνες (π.χ., περιστροφή, αναστροφή, αλλαγή φωτεινότητας, ζουμ). Η συγκεκριμένη μέθοδος προστέθηκε αργότερα ως λύση στο πρόβλημα του overfitting.
- Στη συνέχεια, προστίθεται ένα **Dropout(0.3)** για την αποφυγή **overfitting** και τέλος ορίζει έναν νευρώνα με το **Dense(1)** για την έξοδο.
- ο **Adam** προσαρμόζει τα βάρη “W” και τα bias “b” στο δίκτυο για να μειώσει το σφάλμα της πρόβλεψης (**MAE**).

$$y = activation(W \cdot x + b)$$

- Η **MAE** χρησιμεύει ως μέτρο για να αξιολογηθεί πόσο καλά το μοντέλο ταιριάζει στα δεδομένα.
- Το μοντέλο εκπαιδεύεται για 100 **εποχές** με μέγεθος **batch_size** 32. Ο έλεγχος γίνεται με το σύνολο δοκιμής και χρησιμοποιούνται **callbacks**, για να αποθηκευτεί το καλύτερο μοντέλο και να μειωθεί ο ρυθμός εκμάθησης όταν η ακρίβεια δεν βελτιώνεται.
- Τέλος, μετατρέπεται το ολοκληρωμένο μοντέλο σε **TensorFlow Lite** μορφή για κινητές συσκευές και αποθηκεύεται(Εικόνα 6.15).

```
import numpy as np
import pandas as pd
import tensorflow as tf
import os
import cv2
import matplotlib.pyplot as plt
from tqdm import tqdm


path=r"D:\pythaitry\code\data"
files=os.listdir(path)
files.sort()
print(files)
image_array=[]
label_array=[]

for i in tqdm(range(len(files))):
    sub_file=os.listdir(path+"/"+files[i])
    for j in range(len(sub_file)):

        file_path=path+"/"+files[i]+"/"+sub_file[j]
        image=cv2.imread(file_path)
        image=cv2.resize(image,(96,96))
        image=cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
        image_array.append(image)
        label_array.append(i)


image_array=np.array(image_array)
label_array=np.array(label_array,dtype="float")
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(image_array,label_array,test_size=0.15)
del image_array,label_array
import gc
gc.collect()
```

Εικόνα 6.14 Ορισμοί και βιβλιοθήκες για την δημιουργία του μοντέλου

```
model=Sequential()
pretrained_model=tf.keras.applications.EfficientNetB0(input_shape=(96,96,3),include_top=False)
model.add(pretrained_model)
model.add(layers.GlobalAveragePooling2D())
model.add(layers.Dropout(0.3))
model.add(layers.Dense(1))
model.build(input_shape=(None,96,96,3))

model.summary()
model.compile(optimizer="adam",loss="mae",metrics=["mae"])
ckp_path="trained_model/modelkostakis.weights.h5"
model_checkpoint=tf.keras.callbacks.ModelCheckpoint(
                                    filepath=ckp_path,
                                    monitor="val_mae",
                                    mode="auto",
                                    save_best_only=True,
                                    save_weights_only=True
                                    )

reduce_lr=tf.keras.callbacks.ReduceLROnPlateau(
                                    factor=0.9,
                                    monitor="val_mae",
                                    mode="auto",
                                    cooldown=0,
                                    patience=5,
                                    verbose=1,
                                    min_lr=1e-6)


Epochs=100
Batch_Size=32
history=model.fit(
                X_train,
                Y_train,
                validation_data=(X_test,Y_test),
                batch_size=Batch_Size,
                epochs=Epochs,
                callbacks=[model_checkpoint,reduce_lr]
                )

model.load_weights(ckp_path)
converter=tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model=converter.convert()

with open("model.tflite","wb") as f:

    f.write(tflite_model)

prediction_val=model.predict(X_test,batch_size=32)

print(prediction_val[:10])
print(Y_test[:10])
```

Εικόνα 6.15 Δημιουργία του μοντέλου και υλοποίηση του testing

Η εκπαίδευση κάθε μοντέλου διαρκεί περίπου 3 ώρες, αλλά αυτό ισχύει μόνο για τη συγκεκριμένη συσκευή και τις συγκεκριμένες ρυθμίσεις που χρησιμοποιούνται. Υπάρχουν πολλοί παράγοντες που επηρεάζουν τον χρόνο εκπαίδευσης, όπως ο αριθμός των εποχών (epochs), η κάρτα γραφικών και άλλοι υπολογιστικοί πόροι. Κατά τη διάρκεια της εκπαίδευσης, είναι απαραίτητο ο υπολογιστής να παραμείνει ανοιχτός και να μην τεθεί σε λειτουργία ύπνου (sleep mode), καθώς αυτό θα διακόψει τη διαδικασία εκπαίδευσης του μοντέλου. Με την εκτέλεση του **sign_language_recognition.py** στο τερματικό (terminal), εμφανίζεται στον χρήστη ένας πίνακας που παρουσιάζει τη δομή του μοντέλου και πληροφορίες για τα διάφορα επίπεδά του (layers).

Ο πίνακας στην Εικόνα 6.16 περιλαμβάνει τις εξής πληροφορίες:

- **layer (type)**: Το όνομα και ο τύπος του επιπέδου **(EfficientNetB0, GlobalAveragePooling2D, Dropout, Dense**).
    - **GlobalAveragePooling2D:** Μετατρέπει το τρισδιάστατο σχήμα του **EfficientNetB0** σε μονοδιάστατο, για συμπύκνωση του μοντέλου (3, 3, 1280 → 1280)
- **output shape**: Το σχήμα εξόδου κάθε επιπέδου.
- **param**: Ο αριθμός των παραμέτρων του επιπέδου, δηλαδή των τιμών που προσαρμόζονται κατά την εκπαίδευση.

Έπειτα παρουσιάζεται το συνολικό πλήθος παραμέτρων του μοντέλου, οι παράμετροι που εμφανίζονται κατά την εκπαίδευση και οι παράμετροι που παραμένουν σταθερές (π.χ. σε προεκπαιδευμένα μοντέλα). Επίσης ο χρήστης βλέπει λεπτομέρειες για κάθε εποχή (epoch), όπως:

- **Epoch**: Ο αριθμός της εποχής που βρίσκεται σε εξέλιξη (π.χ 1/10).
- **Loss**: Το σφάλμα του μοντέλου κατά την εκπαίδευση.
- **MAE (Mean Absolute Error)**: Η μέση απόλυτη απόκλιση μεταξύ των προβλέψεων του μοντέλου και των πραγματικών τιμών.
- **Val_loss / Val_mae**: Οι αντίστοιχες μετρήσεις για τα δεδομένα επικύρωσης (validation).
- **Learning rate**: Ο ρυθμός εκμάθησης που χρησιμοποιεί ο αλγόριθμος (π.χ 0.0010).

```
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X']
100%|██████████| 24/24 [02:33<00:00,  6.41s/it]
2025-01-24 05:37:23.603278: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to
use available CPU instructions in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler fl
ags.
Model: "sequential"
```

Layer (type)	Output Shape	Param #
efficientnetb0 (Functional)	(None, 3, 3, 1280)	4,049,571
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dropout (Dropout)	(None, 1280)	0
dense (Dense)	(None, 1)	1,281

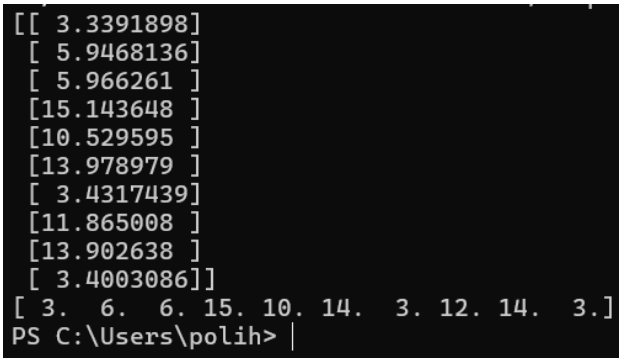
```
Total params: 4,050,852 (15.45 MB)
Trainable params: 4,008,829 (15.29 MB)
Non-trainable params: 42,023 (164.16 KB)
Epoch 1/10
332/332 ━━━━━━━━━━━━━━━━━━━━ 148s 352ms/step - loss: 4.3428 - mae: 4.3428 - val_loss: 3.2480 - val_mae: 3.2480 - learnin
g_rate: 0.0010
Epoch 2/10
332/332 ━━━━━━━━━━━━━━━━━━━━ 114s 344ms/step - loss: 1.1783 - mae: 1.1783 - val_loss: 1.5629 - val_mae: 1.5629 - learnin
g_rate: 0.0010
Epoch 3/10
```

Εικόνα 6.16 Δημιουργία του μοντέλου και υλοποίηση του testing

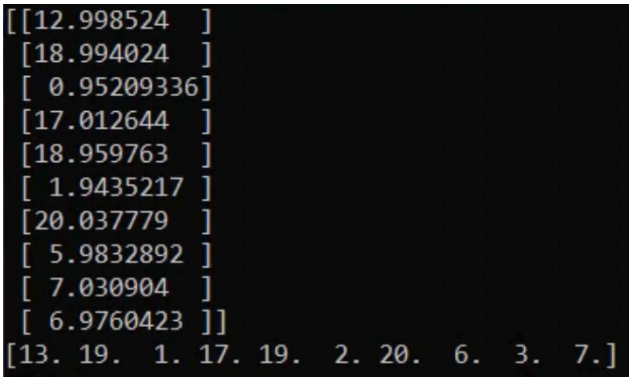
Για την κατανόηση των αποτελεσμάτων του μοντέλου, καθώς και για την καλύτερη αξιολόγηση των μοντέλον, δημιουργήθηκε ακόμα ένα έξτρα μοντελο (έκτο μοντέλο). Το συγκεκριμένο μοντέλο σε αντίθεση με τα υπόλοιπα, εκπαιδεύεται για 10 εποχές και όχι 100. Αυτό έχει σαν αποτέλεσμα το μοντέλο να μην είναι τόσο ακριβές είναι τα υπόλοιπα. Στην Εικόνα 6.17 περιέχονται τα αποτελέσματα του μοντέλου που εκπαιδεύτηκε 10 εποχές, ενώ στην Εικόνα 6.18 το μοντέλο που εκπαιδεύτηκε για 100 εποχές. Παρατηρείτε ότι οι προβλέψειμες τιμές της Εικόνας 6.18 είναι σημαντικά πιο κοντά από τις αντίστοιχες του έκτου μοντέλου.

Οι περισσότερες προβλέψεις, για το μοντέλο που τρέχει 10 εποχές, είναι πολύ κοντά στις πραγματικές τιμές, με αποκλίσεις κυρίως κάτω από 0.5. Η μεγαλύτερη απόκλιση είναι **0.53** στη θέση 5 (10.529 αντί για 10). Στο μοντέλο με που έτρεξε 100 εποχές, οι προβλέψεις είναι εξαιρετικά κοντά στις πραγματικές τιμές στις περισσότερες θέσεις. Ωστόσο, υπάρχει μία μεγάλη απόκλιση στη θέση 9 (7.03 αντί για 3), η οποία "ξεφεύγει" σημαντικά σε σχέση με τις υπόλοιπες τιμές.

Οι προβλέψεις του μοντέλου με 100 epochs είναι γενικά πιο ακριβείς. Οι αποκλίσεις είναι πολύ μικρές, κάτι που δείχνει ότι το μοντέλο είναι πιο αποτελεσματικό, ακόμα κι αν δεν πετυχαίνει όλες τις προβλέψεις ακριβώς. Ωστόσο, η μεγαλύτερη απόκλιση στη θέση 9 στο μοντέλο με 100 epochs είναι αποτέλεσμα **overfitting**, που διορθώθηκε με την προσθήκη του **Data Augmentation** (Εικόνα 6.19).



Εικόνα 6.17 Αποτέλεσμα μοντέλου 10 εποχών



Εικόνα 6.18 Αποτέλεσμα μοντέλου 100 εποχών

```
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

Εικόνα 6.19 Βελτίωση μοντέλου με Data Augmentation

6.6 Ενσωμάτωση Μοντέλου στην Εφαρμογή

Η εφαρμογή περιέχει δύο κύριες κλάσης. την **CameraActivity** και την **objectDetectorClass.** Η **CameraActivity** είναι αυτή που χειρίζεται την κάμερα, ενώ η **objectDetectorClass** χειρίζεται την αναγνώριση της νοηματικής.

CameraActivity

- **Εισαγωγή βιβλιοθηκών :**
  - **OpenCV module 3413:** Για τη διαχείριση εικόνων και βίντεο.
    - ■
  - **AndroidX:** Για τη διαχείριση αδειών της κάμερας.
  - **TensorFlow Lite:** Για την υλοποίηση ανίχνευσης αντικειμένων σε πραγματικό χρόνο.
- **Βασικές λειτουργίες :**
  - **setCameraIndex(CameraBridgeViewBase.CAMERA_ID_FRONT)** ορισμός επιλογής κάμερα.
  - **hand_model.tflite** φόρτωση του pretrained μοντέλου, για ανίχνευση χεριών**.**
  - Φορτώνει τις ετικέτες από το αρχείο **custom_label.txt.**
  - Ορίζεται το μέγεθος εισόδου του μοντέλου σε 300x300 pixels.
  - Κάλεσμα και διαβασμα απο την **objectDetectorClass**, η οποία υλοποιεί την ανίχνευση αντικειμένων μέσω TensorFlow Lite (Εικόνα 6.20).
  - **onCameraFrame()** επεξεργάζεται κάθε καρέ (frame) που λαμβάνεται από την κάμερα με **inputFrame.rgba()** για έγχρωμη εικόνα και **inputFrame.gray()** για γκρι κλίμακα.
  - Με το **core.flip** διορθώνεται ο προσανατολισμός της εικόνας (κατακόρυφη και οριζόντια αναστροφή).
  - Από την **objectDetectorClass** καλείται η **recognizeImage()** για την ανίχνευση χειρονομίας.

```
    setContentView(R.layout.activity_camera);

    mOpenCvCameraView=(CameraBridgeViewBase) findViewById(R.id.frame_Surface);
    mOpenCvCameraView.setVisibility(SurfaceView.VISIBLE);

    // Set the camera index to front camera
    mOpenCvCameraView.setCameraIndex(CameraBridgeViewBase.CAMERA_ID_FRONT);

    mOpenCvCameraView.setCvCameraViewListener(this);
    try{
        objectDetectorClass=new objectDetectorClass(getAssets(), modelPath: "hand_model.tflite",
                labelPath: "custom_label.txt", inputSize: 300, classification_model: "Sign_l_m.tflite", classification_input_size: 96);
        Log.d( tag: "MainActivity", msg: "Model is successfully loaded");
    }
    catch (IOException e){
        Log.d( tag: "MainActivity", msg: "Getting some error");
        e.printStackTrace();
    }
}
```

Εικόνα 6.20 Κάλεσμα και παράμετροι του objectDetectorClass

objectDetectorClass

- **Αρχικά βήματα**
    - **Interpreter:** Υπεύθυνο για την εκτέλεση των TFLite μοντέλων.Ο interpreter2 είναι για κλασικοποίηση.
    - **GpuDelegate:** Ενεργοποιεί την υποστήριξη GPU για γρηγορότερη επεξεργασία του μοντέλου.
    - **labelList:** Αποθηκεύει τις ετικέτες (labels) που αντιστοιχούν στις κλάσεις του μοντέλου.
    - Διαστάσεις εικόνας (**INPUT_SIZE, PIXEL_SIZE, IMAGE_MEAN, IMAGE_STD**): Χρησιμοποιούνται για την κλιμάκωση των εικόνων που λαμβάνονται σε πραγματικό χρόνο, ώστε να είναι συμβατές με τα μοντέλα.
- Δημιουργία **constructor** για την κλάση **objectDetectorClass.**
- **loadModelFile** φορτώνει τα αρχεία μοντέλου από τα assets και τις ταμπέλες τους.
- **Μέθοδος recognizeImage:** Αυτή είναι η κύρια μέθοδος για την αναγνώριση αντικειμένων σε μια εικόνα τύπου Mat (από OpenCV).
    - Αναστροφή εικόνας όπως αναφέρθηκε προηγουμένως.
    - Η **Mat** εικόνα μετατρέπεται σε **Bitmap** για περαιτέρω επεξεργασία και για να ταιριάζει με τις διαστάσεις του μοντέλου(Εικόνα 6.21).

```
Mat rotated_mat_image=new Mat();

Mat a=mat_image.t();
Core.flip(a,rotated_mat_image, flipCode: 1);
a.release();
Bitmap bitmap=null;
bitmap=Bitmap.createBitmap(rotated_mat_image.cols(),rotated_mat_image.rows(),Bitmap.Config.ARGB_8888);
Utils.matToBitmap(rotated_mat_image,bitmap);
height=bitmap.getHeight();
width=bitmap.getWidth();

Bitmap scaledBitmap=Bitmap.createScaledBitmap(bitmap,INPUT_SIZE,INPUT_SIZE, filter: false);

ByteBuffer byteBuffer=convertBitmapToByteBuffer(scaledBitmap);

Object[] input=new Object[1];
input[0]=byteBuffer;

Map<Integer,Object> output_map=new TreeMap<>();
```

Εικόνα 6.21 Αρχικοποίηση των λειτουργών recognizeImage

- Τα αποτελέσματα της πρόβλεψης αντικειμένων περιλαμβάνουν:
  - **boxes**: Συντεταγμένες των αντικειμένων.
  - **scores**: Πιθανότητες πρόβλεψης.
  - **classes**: Κατηγορίες των αντικειμένων.
- Καλείτε η **getAlphabets** για τα αποτελέσματα της πρόβλεψης αντικειμένων περιλαμβάνουν:
  - Οι συντεταγμένες των αντικειμένων επεξεργάζονται ώστε να προσαρμοστούν στις διαστάσεις της αρχικής εικόνας.
  - Μετατρέπει την πρόβλεψη σε συγκεκριμένο γράμμα ανάλογα με την αριθμητική τιμή (**sign_v**). Εάν το σκορ πιθανότητας είναι πάνω από 0.5, θεωρείται ότι το αντικείμενο έχει ανιχνευθεί. (Εικόνα 6.22).

```
private String get_alphabets(float sign_v) {
    String val = "";

    if (sign_v >= -0.5 & sign_v < 0.5) {
        val = "Alpha";
    } else if (sign_v >= 0.5 & sign_v < 1.5) {
        val = "Beta";
    } else if (sign_v >= 1.5 & sign_v < 2.5) {
        val = "Gamma";
    } else if (sign_v >= 2.5 & sign_v < 3.5) {
        val = "Delta";
    } else if (sign_v >= 3.5 & sign_v < 4.5) {
        val = "Epsilon";
    } else if (sign_v >= 4.5 & sign_v < 5.5) {
        val = "Zeta";
    } else if (sign_v >= 5.5 & sign_v < 6.5) {
        val = "Eta";
    } else if (sign_v >= 6.5 & sign_v < 7.5) {
        val = "Theta";
    } else if (sign_v >= 7.5 & sign_v < 8.5) {
        val = "Iota";
    } else if (sign_v >= 8.5 & sign_v < 9.5) {
        val = "Kappa";
    } else if (sign_v >= 9.5 & sign_v < 10.5) {
        val = "Lambda";
    } else if (sign_v >= 10.5 & sign_v < 11.5) {
        val = "Mu";
    } else if (sign_v >= 11.5 & sign_v < 12.5) {
        val = "Nu";
    } else if (sign_v >= 12.5 & sign_v < 13.5) {
        val = "Xi";
    } else if (sign_v >= 13.5 & sign_v < 14.5) {
        val = "Omicron";
    } else if (sign_v >= 14.5 & sign_v < 15.5) {
        val = "Pi";
    } else if (sign_v >= 15.5 & sign_v < 16.5) {
        val = "Rho";
```

Εικόνα 6.22 Παράδειγμα μέρους αντιστοιχίας πρόβλεψης

- **convertBitmapToByteBuffer** και **convertBitmapToByteBuffer1** (Εικόνα 6.23):
    - Μετατρέπουν μια εικόνα **Bitmap** σε **ByteBuffer**, ώστε να μπορεί να χρησιμοποιηθεί ως είσοδος στο TensorFlow Lite μοντέλο που έχει φορτοθεί από τον φάκελο assets.
    - Τα pixels της εικόνας κανονικοποιούνται στη μορφή 0-1 (με διαίρεση στο 255.0).

```
private ByteBuffer convertBitmapToByteBuffer(Bitmap bitmap) {
    ByteBuffer byteBuffer;
    int quant=1;
    int size_images=INPUT_SIZE;
    if(quant==0){
        byteBuffer=ByteBuffer.allocateDirect( capacity: 1*size_images*size_images*3);
    }
    else {
        byteBuffer=ByteBuffer.allocateDirect( capacity: 4*1*size_images*size_images*3);
    }
    byteBuffer.order(ByteOrder.nativeOrder());
    int[] intValues=new int[size_images*size_images];
    bitmap.getPixels(intValues, offset: 0,bitmap.getWidth(), x: 0, y: 0,bitmap.getWidth(),bitmap.getHeight());
    int pixel=0;
    for (int i=0;i<size_images;++i){
        for (int j=0;j<size_images;++j){
            final  int val=intValues[pixel++];
            if(quant==0){
                byteBuffer.put((byte) ((val>>16)&0xFF));
                byteBuffer.put((byte) ((val>>8)&0xFF));
                byteBuffer.put((byte) (val&0xFF));
            }
            else {
                byteBuffer.putFloat( v: (((val >> 16) & 0xFF))/255.0f);
                byteBuffer.putFloat( v: (((val >> 8) & 0xFF))/255.0f);
                byteBuffer.putFloat( v: (((val) & 0xFF))/255.0f);
            }
        }
    }
    return byteBuffer;
}
```

Εικόνα 6.23 Παράδειγμα μέρους κατηγοριοποίησης αντικειμένων

- Απεικόνιση αποτελεσμάτων με χρήση του module **OpenCV,** δημιουργείται περίβλημα γύρω από το ανυχνευμένο χέρι με σημείωση το αποτέλεσμα του γράμματος που ανιχνεύτηκε (Εικόνα 6.24):
    - Αν η βαθμολογία του αντικειμένου είναι μεγαλύτερη από 0.5, τότε το αντικείμενο θεωρείται ως έγκυρο και συνεχίζεται η διαδικασία απεικόνισης.
    - Ανακτούνται οι συντεταγμένες για την σωστή τοποθέτηση του box (περίβλημα)
    - **y1, x1, y2, x2** είναι οι τέσσερις τιμές που καθορίζουν το ορθογώνιο box του αντικειμένου στην εικόνα .
    - Οι συντεταγμένες αυτές είναι τιμές μεταξύ 0 και 1, οπότε πολλαπλασιάζονται με το ύψος και το πλάτος για να προσαρμοστούν στις πραγματικές διαστάσεις της εικόνας.
    - **Rect** ορίζει την περιοχή της εικόνας που περιέχει το αντικείμενο, και στη συνέχεια το απομονώνει.

```
for (int i=0;i<10;i++){
    float class_value=(float) Array.get(Array.get(Object_class, index: 0),i);
    float score_value=(float) Array.get(Array.get(score, index: 0),i);
    if(score_value>0.5){
        Object box1=Array.get(Array.get(value, index: 0),i);
        float y1=(float) Array.get(box1, index: 0)*height;
        float x1=(float) Array.get(box1, index: 1)*width;
        float y2=(float) Array.get(box1, index: 2)*height;
        float x2=(float) Array.get(box1, index: 3)*width;
        if(y1<0){
            y1=0;
        }
        if (x1<0){
            x1=0;
        }
        if (x2>width){
            x2=width;
        }
        if (y2>height){
            y2=height;
        }
```

```
float w1=x2-x1;
float h1=y2-y1;
Rect cropped_roi=new Rect((int)x1,(int)y1,(int)w1,(int)h1);
Mat cropped=new Mat(rotated_mat_image,cropped_roi).clone();
Bitmap bitmap1=null;
bitmap1=Bitmap.createBitmap(cropped.cols(),cropped.rows(),Bitmap.Config.ARGB_8888);
Utils.matToBitmap(cropped,bitmap1);
Bitmap scaledBitmap1=Bitmap.createScaledBitmap(bitmap1,Classification_Input_Size,Classification_Input_Size, filter: false);
ByteBuffer byteBuffer1=convertBitmapToByteBuffer1(scaledBitmap1);
float[][] output_class_value=new float[1][1];
interpreter2.run(byteBuffer1,output_class_value);
Log.d( tag: "objectDetectionClass", msg: "output_class_value: "+output_class_value[0][0]);
String sign_val=get_alphabets(output_class_value[0][0]);
Imgproc.putText(rotated_mat_image, text: ""+sign_val,new Point( x: x1+10, y: y2+40), fontFace: 2, fontScale: 1.5,new Scalar(255, 255, 255, 255), thickness: 2);
Imgproc.rectangle(rotated_mat_image,new Point(x1,y1),new Point(x2,y2),new Scalar(0, 255, 0, 255), thickness: 2);
```

Εικόνα 6.24 Παράδειγμα μέρους σχεδίασης το box

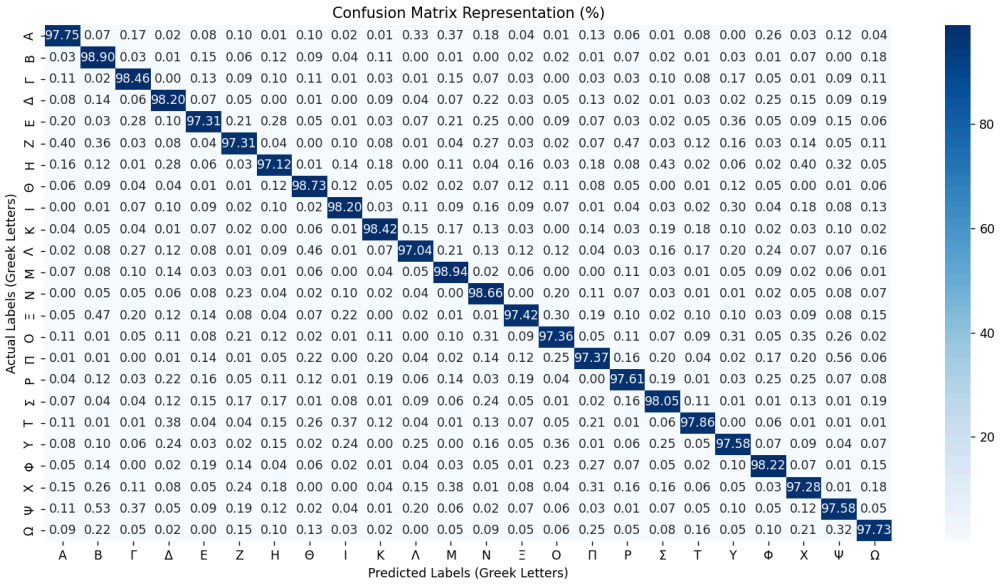
Στο xml αρχείο του **MainActivity (Nyx)** υπάρχουν τα παρακάτω στοιχεία:

- **ImageView**: Φωτογραφία με τα γράμματα του Ελληνικού αλφάβητου, για την εκπαίδευση του χρήστη.
- **camera_button**: Μεταφέρει τον χρήστη στην διεπαφή **CameraActivity** που περιέχει την κάμερα και τις λειτουργίες αναγνώρισης νοηματικής.

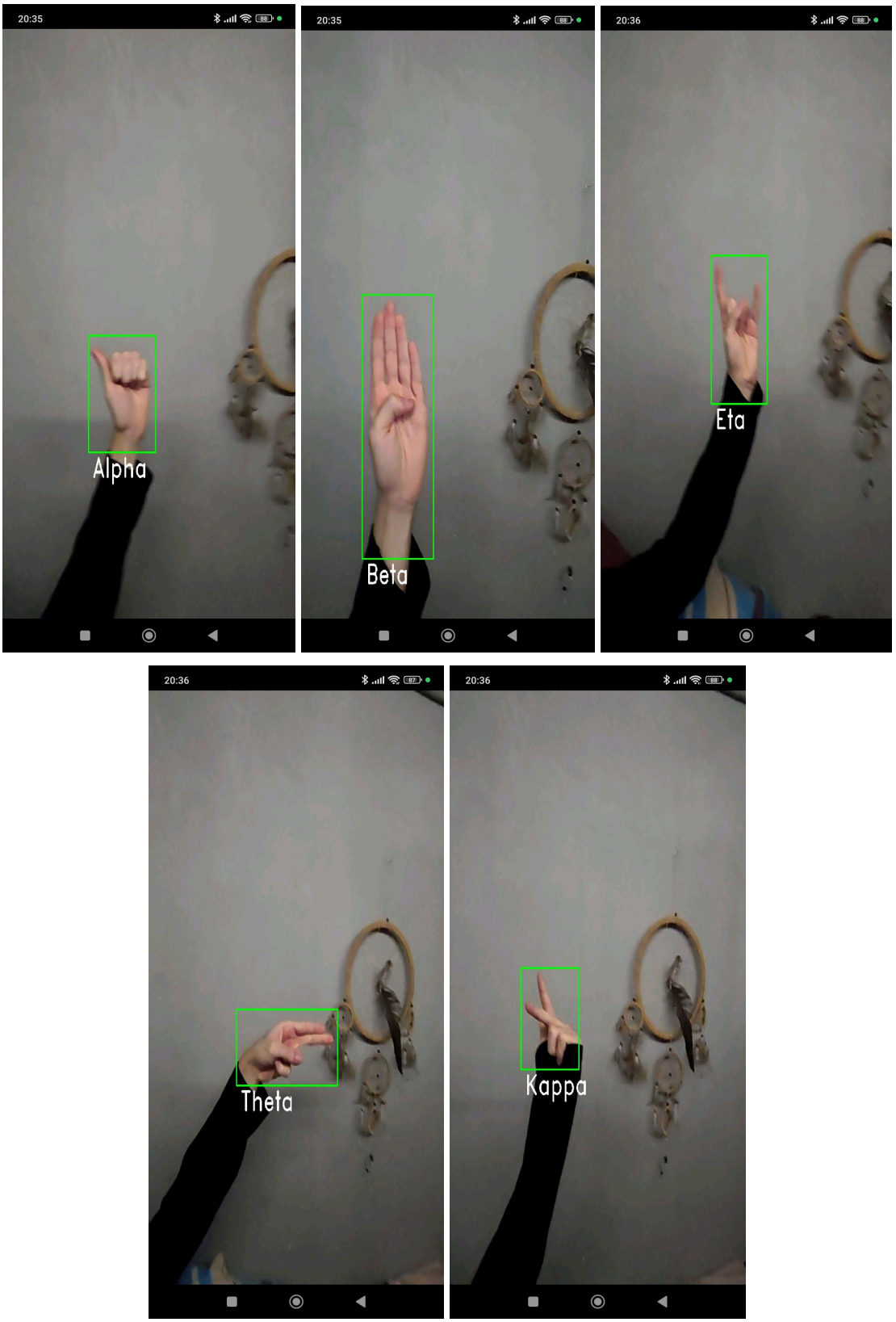
Η ροή πλοήγησης αρχίζει από την εφαρμογή Aather και συγκεκριμένα από την επιλογή του χρήστη να κάνει κλικ στην Εικόνα 3.9. Στην συνέχεια, καλείτε το namespace της εφαρμογής Nyx και ο χρήστης μεταφέρεται αυτόματα στην εφαρμογή. Η πρώτη διεπαφή που θα το εμφανιστεί είναι η διεπαφή της Εικόνας 6.25, η οποία απεικονίζει γράμματα το GSL. Αν πατήσει ο χρήστης στο κουμπί της διεπαφής, θα μεταφερθεί σε ένα καινούργιου περιβάλλον, που έχει φτιαχτεί συγκεκριμένα για να έχει την δυνατότητα κάμερας και αναγνώρισης νοηματικής. Για παραδείγματα αυτού του περιβάλλοντος αλλά και για παραδείγματα χρήσης της αναγνώρισης νοηματικής, μπορεί να παρατηρηθεί η εικόνα 6.27.

Εικόνα 6.25 MainMenu της εφαρμογής Nyx και απεικόνιση GSL

Ο πίνακας που απεικονίζεται στην Εικόνα 6.26 χρησιμεύει ως βάση για τον υπολογισμό της αξιολόγησης της λειτουργικότητας. Αναλύει τις προβλέψεις σε τέσσερις κατηγορίες: σωστές προβλέψεις και για τις δύο κατηγορίες (αληθινά θετικά και αληθινά αρνητικά) και λανθασμένες προβλέψεις (ψευδώς θετικά και ψευδώς αρνητικά). Αυτό βοηθά να κατανοηθεί πού κάνει λάθη το μοντέλο, ώστε να είναι πιο εύκολη η ανάλυση του προβλήματος και η βελτίωση της εφαρμογής.



Εικόνα 6.26 Βελτίωση μοντέλου με Data Augmentation



Εικόνα 6.27 Παράδειγμα Χρήσης Αναγνώρισης Νοηματικής

Κεφάλαιο 7ο: Συμπεράσματα και προτάσεις βελτίωσης

Σε αυτήν την ενότητα θα αναλυθούν τα προβλήματα που παραμένουν στην εφαρμογή και οι δυσκολίες που αντιμετωπίστηκαν, τα συμπεράσματα από την υλοποίηση της εφαρμογής, καθώς και οι τρόποι βελτίωσης κάποιον συγκεκριμένων λειτουργιών.

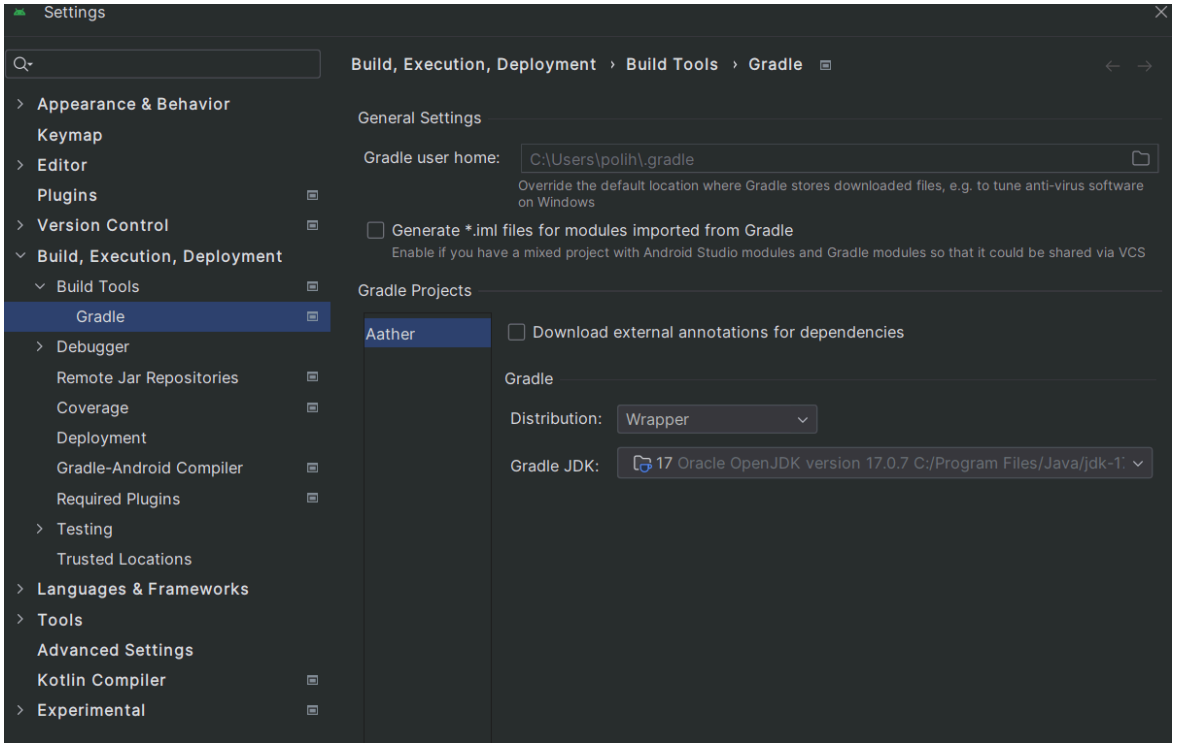
7.1 Δυσκολίες

Η πολυπλοκότητα της εφαρμογής επηρέασε πολύ την αναπτυξη της και δημιούργησε αρκετές δυσκολίες σε αρκετές διεπαφές. Παρακάτω παρουσιάζονται οι δυσκολίες που κατάφεραν να αντιμετοπιστούν στο πρότζεκτ:

- **Εκκίνηση διεπαφής άμεσης βοήθειας.** Πολλες Android συσκευές όπως η Xiaomi δεν επιτρέπουν την πρόσβαση και τον έλεγχο του κουμπιού “power button”, από του προγραμματιστές. Αυτό γίνεται λόγω ασφάλειας, διότι το συγκεκριμένο κουμπί θεωρείται αντικείμενο **χαμηλού επιπέδου συστήματος** (low-level system component). Η αρχική ιδέα ήταν να γίνεται προσβαση στην άμεση βοήθεια μέσω του συγκεκριμένου κουμπιού, παρόλα αυτά επειδή δεν υπάρχει πρόσβαση στο root της συσκευής, έπρεπε να βρεθεί ένας παραπλήσιος τρόπος ενεργοποίηση της. Η καλύτερη επιλογή ήταν η χρήση των ηχητικών κουμπιών (volume up, volume down), καθώς το Android παρέχει **APIs** για την παρακολούθηση των κουμπιών έντασης.
- Ανίχνευση Ομιλίας. Στην διεπαφή του EduRecorder, η ανίχνευση δεν λογικό είναι να μην δουλεύει πάντα σωστα, με αποτέλεσμα να απεικόνοντα λάθος λέξεις και εκφράσεις στις σημειώσεις του χρήστη. Γι'αυτό απο TextView το πλαίσιο κειμένου άλλαξε σε Edittext. Έτσι ο χρήστης αν παρατηρήσει λάθος μπορεί να το διορθώσει πριν το αποθήκευση στο κινητό του.
- Η δημιουργία μιας πιο σύγχρονης εφαρμογης (Aather), απαιτούσε και την αλλαγή της έκδοση java στο περιβάλλον του υπολογιστή. Για να γίνει αυτό πρέπει ο προγραμματιστής να κατεβάσει την έκδοση που επιθυμεί απο την επίσημη ιστοσελίδα της **Oracle** (σε αυτήν την περίπτωση χρησιμοποιήθηκε η java 17). Στην συνέχεια επιβάλλεται να ρυθμίσει το **JAVA_HOME** και να ενημερώσει το **Path**, οι ρυθμίσεις αυτές μπορούν να βρεθούν κάνοντας δεξί κλικ στο "Ο Υπολογιστής μου" → Ιδιότητες → Προηγμένες Ρυθμίσεις Συστήματος → Μεταβλητές Περιβάλλοντος.
- Οπως αναφέρθηκε η Nyx χρησιμοποιεί την έκδοση java 8. Στην ανάπτυξη του πρότζεκτ αρκετός χρόνος σπαταλίστικε στην προσπάθεια δημιουργίας-τρέξιμο (**build**) της εφαρμογής μέσω του Android Studio. Είναι απαραίτητο να επισημανθεί ότι όταν κατασκευάζεται ένα πρότζεκτ μέσα στο Android studio, το πρόγραμμα λαμβάνει αυτόματα κάποιες από τις default ρυθμίσεις που έχουν οριστεί από τον χρήστη τη πρώτη φορά που άνοιξε το **IDE** (Integrated Development Environment). Στην περίπτωση αυτή η αλλαγή της έκδοσης java δημιούργησε θέματα συμβατότητα με τις **default** ρυθμίσεις. Για να ρυθμιστούν σωστα αυτες οι ρυθμίσεις στις σύγχρονες εκδόσεις του Android studio, ο χρήστης πρέπει να πλοηγηθεί στο menu→ Settings→ Build, Execution, Deployment → Build Tools → Gradle και να επιλέξει στην

συνέχεια ο χρήστης στο πεδίο **Gradle JDK** την έκδοση κατασκευής που επιθυμεί (Στην Εικόνα 7.1 αναφέρετε η κατασκευή του Aather) .

- Υπήρξαν δυσκολίες στην απεικόνιση ελληνικών χαρακτήρων με την χρήση του OpenCV. Δόθηκε μια προσωρινη λυση, χρησιμοποιώντας λατινικούς χαρακτήρες για την απεικόνιση των ονομάτων του ελληνικού αλφάβητου.



Εικόνα 7.1 Παράδειγμα JDK Aather

7.2 Προβλήματα

Παρακάτω γίνεται μια γρήγορη αναφορά σε αδυναμίες, που ακόμα δεν έχουν βρεθεί λύσεις και θα τεθούν προς επεξεργασία για το μέλλον.

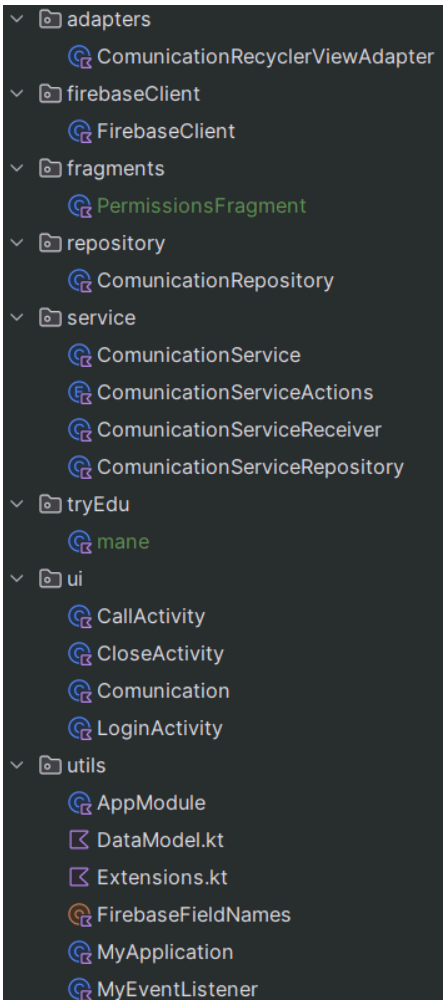
- Στο εργαλείο μέτρησης θορύβου δεν είναι αρκετά ακριβής ο αλγόριθμος μετατροπής θορύβου σε δόνηση. Αναλυτικότερα, ενώ υπάρχουν διαμορφώσεις στην δόνηση, η συνεχής δόνηση της συσκευής δυσκολεύει την αναγνώριση αισθητής διαφοράς έντασης από τον χρήστη.
- Στο εργαλείο ηχογράφησης ομιλίας πολλές φορές δεν αναγνωρίζεται η ελληνική γλώσσα. Επίσης η ηχογράφηση σταματάει όταν σταματήσει η ανίχνευση ομιλίας από το μικρόφωνο, αντίθετα θα έπρεπε να σταματάει μετά από ένα διάστημα 50 δευτερολέπτων παύσης της ομιλίας,
- Στο εργαλείο μέτρησης θορύβου δεν είναι αρκετά ακριβής ο αλγόριθμος μετατροπής θορύβου σε δόνηση. Αναλυτικότερα, ενώ υπάρχουν διαμορφώσεις στην δόνηση, η συνεχής δόνηση της συσκευής δυσκολεύει την αναγνώριση αισθητής διαφοράς έντασης από τον χρήστη.

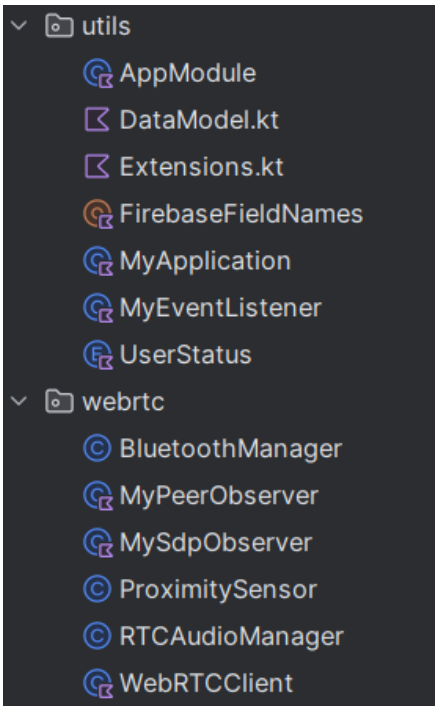
- Θεωρείται αδυναμία του συστήματος η ύπαρξη δύο διαφορετικών εφαρμογών με διαφορετικές εκδόσεις. Προς το παρόν δεν έχει βρεθεί τρόπος ενσωματωσης της εφαρμογής Nyx στο Aather. Επίσης, η ανίχνευση ορισμένων νοηματικών γραμμάτων, που είναι όμοια μεταξύ τους, δυσκολεύονται να αναγνωριστούν. Παράλληλα χρειάζεται καλός φωτισμός για την λειτουργία αναγνώρισης. Το συγκεκριμένο είναι μια έξτρα αδυναμία της μικρής πολυμορφικότητας του dataset, καθώς αυτό περιέχει χειρονομίες μόνο από δύο άτομα.

7.3 Μελλοντικές Βελτίωσης

Αρχικά η πιο σημαντική ενημέρωση που αναπτύσσεται τον τελευταίο καιρο στην εφαρμογή είναι η δημιουργία επικοινωνιακής πλατφόρμας, οι οποία θα γίνει προσβάσιμη στους χρήστες όταν τελειοποιηθεί. Η πλατφόρμα θα ενεργοποιείται μετά από το πάτημα του χρήστη στην αντίστοιχη διεπαφή **Connected** (Εικόνες 3.6, 3.7 και 3.8). Εκεί ο χρήστης θα έχει τον λογαριασμό του μέσω **firebase** και θα του δίνετε η δυνατότητα επικοινωνίας με άλλους χρήστες της εφαρμογής με δύο τρόπους: μέσω μηνυμάτων και μέσω βιντεοκλήση εκμεταλλεύοντας την τεχνολογία **WebRTC**.Πρέπει να αναφερθεί ότι ο χρήστης στις διεπαφές Manual (χρησιμοποιείται για επιλογή βοήθειας από του τους εθελοντες) και Connected εμφανίζουν την Εικόνα 7.2 και ενημερώνουν την παρούσα κατάσταση του εργαλείου. Η δομή του εργαλείου και ένα μεγάλο μέρος του έχει ήδη πραγματοποιηθεί (οπτικοποίησης της δομής στην Εικόνα 7.3), παρόλα αυτα δεν χρειάζεται να αναλυθεί παραπάνω το συγκεκριμένο εργαλείο καθως βρίσκεται προς επεξεργασίας.

Εικόνα 7.2 Ενημέρωση Status στον User





Εικόνα 7.3 Δομή της Επικοινωνιακής πλατφόρμας

Η παρακάτω λίστα περιγράφει κάποιες ιδέες και κάποιους τους τρόπους με τους οποίους θα βελτιωθούν στο μέλλον τα παρόν εργαλεία, που παρέχονται για δοκιμή στον χρήστη:

- **Διαμόρφωση Ui:** Στο μέλλον με την χρήση του firebase και ενός ερωτηματολογίου, ο χρήστης θα ενημερώνει το σύστημα με πληροφορίες για την πάθησή του. Ετσι θα αλλάζει το γραφικό περιβάλλον, τα χρώματα του συστήματος, καθώς και τις διεπαφές για να προσαρμόζονται στα χαρακτηριστικά του χρήστη.
- **Εργαλείο Μέτρησης Θορύβου:** Βελτίωση ακρίβειας μετρήσεων με αλλαγή του κωδικοποιητή AMR-NB σε AAC (Advanced Audio Codec), που μπορεί να καλύπτει υψηλότερες και χαμηλότερες συχνότητες.
- Επίσης καλό θα ήταν να αναπτυχθεί το εργαλείο **Μεγέθυνσης και Σμίκρυνσης** για να μετατρέπει όχι μονο φωτογραφίες αλλα και βίντεο σε πραγματικό χρόνο.
- Με αφετηρία την εφαρμογή **Nyx**, μπορούν να αναπτυχθούν ενδιαφέρουσες τεχνολογίες. Συγκεκριμένα, για την ανίχνευση και μετάφραση της νοηματικής γλώσσας, θα μπορούσε να δημιουργηθεί ένα νέο μοντέλο τεχνητής νοημοσύνης, το οποίο θα εκπαιδεύεται σε ένα αρκετά μεγαλύτερο dataset. Μετά από έρευνα, διαπιστώθηκε ότι το dataset που παρέχει το Ινστιτούτο Τεχνολογιών Πληροφορικής και Επικοινωνιών[19] θα ήταν μια καλή επιλογή για το πρότζεκτ. Η δημιουργία ενός ai μοντέλου από ένα τόσο μεγάλο dataset θα επιβαρύνει πολύ την εφαρμογή, οπότε η αλλαγη της εκπαίδευσης μοντέλου EfficienNetB0 είναι απαραίτητη. Το EfficienNetB5 για ενσωμάτωση του μοντέλου σε server είναι μια αξιοσημείωτη λύση που θα βοηθούσε στην βελτίωση της εφαρμογής, διότη η εφαρμογή θα επικοινωνεί με τον σερβερ και δεν θα επιβαρύνεται η ίδια.

7.4 Επίλογος

Υπάρχει μια λανθασμένη αντίληψη ότι τα smartphones είναι λιγότερα πιθανόν να μην χρησιμοποιηθούν από άτομα με προβλήματα όρασης, καθώς βασίζονται σε μεγάλο βαθμό στην οπτική λειτουργικότητα του χρίστη. Τα τελευταία χρόνια, η κινητή τεχνολογία έχει ενσωματώσει αξιόλογες προηγμένες τεχνολογίες που χρησιμοποιούν ήχους, απτικές διεπαφές και χειρονομίες για να αλληλεπιδράσουν οι χρήστες με τα smartphone, υποκαθιστώντας έτσι την οπτική και φωνητική αλληλεπίδραση. Τέτοια εργαλεία ανέπτυξε και διαθέτει η συγκεκριμένη διπλωματική εργασια, για να βοηθήσει στην κάλυψη της μεγάλης ανάγκης δημιουργίας βοηθητικών εφαρμογών σε κινητές συσκευές, που αφορούν τις κοινότητες στις οποιες εντάσσονται άτομα με προβλήματα όρασης και ακοής.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Βιβλία

[5]World Health Organization, *World Report on Hearing*. Sensory Functions, Disability and Rehabilitation (SDR), Geneva, Switzerland: World Health Organization, 202*1*. [Online]. Available: https://www.who.int/publications/i/item/9789240020481

Internet Site

[1] World Health Organization, "Blindness and visual impairment" World Health Organization, 2019. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment.

[2] World Health Organization, *World Report on Vision*, World Health Organization, 2019. [Online]. Available: https://www.who.int/publications/i/item/world-report-on-vision.

[3]Εθνική Στατιστική Υπηρεσία, *Greece in Figures*, Εθνική Στατιστική Υπηρεσία, 2020. [Online]. Available: https://www.statistics.gr/greece-in-figures.

[4] World Health Organization, 2021. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss.

[6] Εθνική Στατιστική Υπηρεσία, *Greece in Figures*. Athens, Greece: Εθνική Στατιστική Υπηρεσία, 2020. [Online]. Available: https://www.statistics.gr/greece-in-figures.

[12] Vision Australia, "*Typography in Inclusive Design Part 1: 8 key tips for accessible typography*" Vision Australia Digital Access Blog, [Online]. Available: https://www.visionaustralia.org/business-consulting/digital-access/blog/typography-in-inclusive-design-part-1.

[13] Amardeep Rawat, "*How to Design Accessibility App for Visually Impaired?*" AppInventiv Blog, Aug. 22, 2022. [Online]. Available: https://appinventiv.com/blog/design-accessibility-app-for-visually-impaired/.

[14] Android Developers, "*Kotlin Overview*" [Online]. Available: https://developer.android.com/kotlin/overview.

[17] Pramod Kumar, "*Create hand dataset apk*", Kaggle, [Online]. Available: https://www.kaggle.com/datasets/pramod722445/create-hand-dataset-apk.

[18] Keras Team, "*EfficientNet B0 to B7*" Keras 3 API documentation, 2025. [Online]. Available: https://keras.io/api/applications/efficientnet/.

[19] Information Technologies Institute, "The Greek Sign Language (GSL) Dataset," [Online]. Available: https://vcl.iti.gr/dataset/gsl/.

[20] M. Tan and Q. V. Le, "*EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling*" Google AI Blog, May 29, 2019. [Online]. Available:

https://research.google/blog/efficientnet-improving-accuracy-and-efficiency-through-automl-and-model-scaling/.

[21] Viso.ai, "*EfficientNet: Optimizing Deep Learning Efficiency*" [Online]. Available: https://viso.ai/deep-learning/efficientnet/.

Journal Articles

[7] Suraj Singh Senjam, Souvik Manna & Covadonga Bascaran (2021) "*Smartphones-Based Assistive Technology: Accessibility Features and Apps for People with Visual Impairment, and its Usage, Challenges, and Usability Testing, Clinical Optometry"*, 311-322, DOI: 10.2147/OPTO.S336361. [Online]. Available: https://doi.org/10.2147/OPTO.S336361.

[8] E. Biçek and M. N. Almalı, "*A Mobile Application That Allows People Who Do Not Know Sign Language to Teach Hearing-Impaired People by Using Speech-to-Text Procedures*" *Int. J. Appl. Math. Electron. Comput.*, vol. 8, no. 1, pp. 027-033, Mar. 2020, doi: 10.18100/ijamec.682806. [Online]. Available: https://www.dergipark.org.tr/ijamec.

[9] H. M. Nasir, N. M. A. Brahin, M. M. M. Aminuddin, M. S. Mispan, and M. F. Zulkifli, "*Android based application for visually impaired using deep learning approach*" *IAES Int. J. Artif. Intell. (IJ-AI)*, vol. 10, no. 4, pp. 879–888, Dec. 2021, doi: 10.11591/ijai.v10.i4.pp879-888. [Online]. Available: http://ijai.iaescore.com.

[10] Mohd Nadhir Ab Wahab et al., "*Text reader for visually impaired person*" J. Phys.: Conf. Ser., vol. 1755, 012055, 2021. [Online]. Available: https://doi.org/10.1088/1742-6596/1755/1/012055.

[11] A. Danoji, A. Dhage, R. Kamat, P. Puranik, and S. Sengupta, "MonVoix - An Android application for hearing impaired people," *J. Commun. Technol. Electronics Comput. Sci.*, vol. 8, 2016. [Online]. Available: https://ojs.jctecs.com/index.php/com/article/view/123.

[15] G. M. Machado, M. M. Oliveira, and L. A. F. Fernandes, "*A Physiologically-based Model for Simulation of Color Vision Deficiency*" IEEE Transactions on Visualization and Computer Graphics, vol. 15, no. 6, pp. 1291–1298, 2009. [Online]. Available: https://www.inf.ufrgs.br/~oliveira/pubs_files/CVD_Simulation/CVD_Simulation.html.

[16] F. Viénot, H. Brettel, and J. D. Mollon, "*Digital Video Colourmaps for Checking the Legibility of Displays by Dichromats*" Color Research & Application, vol. 24, no. 4, pp. 243–252, 1999. [Online]. Available: https://vision.psychol.cam.ac.uk/jdmollon/papers/colourmaps.pdf.