ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ
ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη Αυτόνομης Εφαρμογής Μηχανικής Μάθησης για πλοήγηση σε εσωτερικούς και εξωτερικούς χώρους, με χρήση τεχνολογίων AR.

Διπλωματική Εργασία

Καλαθάς Σ. Δημήτριος

Επιβλέπων : Παναγιώτης Τσανάκας Καθηγητής ΕΜΠ

Αθήνα, Οκτώβριος 2023

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ
ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη Αυτόνομης Εφαρμογής Μηχανικής Μάθησης για πλοήγηση σε εσωτερικούς και εξωτερικούς χώρους, με χρήση τεχνολογίων AR.

Διπλωματική Εργασία

Καλαθάς Σ. Δημήτριος

Επιβλέπων : Παναγιώτης Τσανάκας Καθηγητής ΕΜΠ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 6η Οκτώβριου του 2023.

.....	.....	.....
Π.Τσανάκας	Δ.Σούντρης	Σωτ.Ξύδης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Επ. Καθηγητής Ε.Μ.Π

Αθήνα, Οκτώβριος 2023

.....
Καλαθάς Σ. Δημήτριος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Στον 21ο αιώνα και με αφορμή την ραγδαία εξέλιξη της τεχνολογίας, ο άνθρωπος προσπαθεί να απλοποιήσει την καθημερινότητα του, να εξοικονομήσει χρόνο και να αναπτύξει συνεχώς τον τρόπο ζωής και διαβίωσης του, για να καταφέρει να ανταπεξέλθει στους ραγδαίους και συνεχόμενους ρυθμούς ανάπτυξης. Μια περίπτωση που εντάσσεται στην εξοικονόμηση χρόνου και στην απλοποίηση της καθημερινότητας, είναι η χαρτογράφηση πολύπλοκων εσωτερικών και εξωτερικών χώρων με σκοπό την αλληλεπίδρασή τους με τον άνθρωπο. Καθημερινά οι άνθρωποι θέλοντας να καλύψουν κάποιες ανάγκες του, όπως εκπαίδευση, ιατρική φροντίδα, διασκέδαση κτλ. επισκέπτονται τους αντίστοιχους χώρους που προσφέρουν αυτές τις δυνατότητες. Όμως, πολλές φορές έρχονται αντιμέτωποι με ένα χαοτικό περιβάλλον από πολύπλοκες διαδρομές και κατευθύνσεις, αναγκάζοντάς τους να ξοδέψουν πολύτιμο χρόνο κατα τη διάρκεια της πλοήγησής τους, και έχοντας ως πιθανό αποτέλεσμα να χάσουν την ουσία της επίσκεψης τους.

Εδώ επιστρατεύονται οι τεχνολογίες αιχμής, οι οποίες με τον συνδυασμό έξυπνων λειτουργιών και εργαλείων, μπορούν να δημιουργήσουν αποτελεσματικές εφαρμογές για να καλύψουν τις προαναφερθείσες ανάγκες. Αρχικά, στην σημερινή εποχή, το smartphone είναι ίσως η μοναδική συσκευή που όλοι κατέχουν, άρα και το κατάλληλο μέσο στο οποίο θα προτείνεται να υλοποιηθεί μια εφαρμογή πλοήγησης. Επιπρόσθετα θα ήταν ιδανικό να χρησιμοποιηθούν τεχνικές που θα έκαναν την χρήση της εύκολη και ταυτόχρονα αποτελεσματική για κάθε άτομο που θα είχε πρόσβαση. Αυτές έρχονται με την Επαυξημένη Πραγματικότητα που μπήκε στην ζωή της τεχνολογίας και έχει αλλάξει σε μεγάλο βαθμό τον τρόπο κατασκευής εφαρμογών. Μπορεί να συνεισφέρει στην δημιουργία μιας εύχρηστης και διαδραστικής εφαρμογής που θα καθοδηγεί τον επισκέπτη σε πραγματικό χρόνο και με μεγάλη ακρίβεια. Τέλος, έρχεται να προστεθεί η Μηχανική Μάθηση, η οποία πλέον είναι συνυφασμένη με σχεδόν κάθε νέα τεχνολογική κατασκευή. Έχει τη δυνατότητα να συνεισφέρει στην βελτιστοποίηση των εφαρμογών και στην προκειμένη περίπτωση να υποβοηθήσει ένα σύστημα πλοήγησης να προσαρμόζεται σε αλλαγές του περιβάλλοντα χώρου, τις οποίες η Επαυξημένη Πραγματικότητα χρησιμοποιεί για την καθοδήγηση και την δημιουργία των αντίστοιχων διαδρομών.

Για αυτόν το λόγο, σκοπός της παρούσας διπλωματικής είναι η μελέτη, σχεδίαση και ανάπτυξη μιας iOS εφαρμογής, που θα καθοδηγεί όλους του υποψήφιους χρήστες στους χαοτικούς εσωτερικούς και εξωτερικούς χώρους της Πολυτεχνειούπολης. Για να πετύχει αυτό, προστέθηκε ένα μοντέλο Incremental Learning για την βελτίωση του σκοπού της, έγινε χρήση πολλών νέων τεχνολογικών εργαλείων, όπως Flutter SDK και Firebase, και χρησιμοποιήθηκαν ιδιαίτερες αρχιτεκτονικές. Όλα αυτά είχαν σαν αποτέλεσμα, μια αυτόνομη εφαρμογή σε ερευνητικό επίπεδο, που θα μπορούσε να δώσει το έναυσμα για παρόμοιες και πιο αναπτυγμένες εφαρμογές και σε άλλους χώρους, αναγκαίους, για χαρτογράφηση.

Λέξεις κλειδιά

Επαυξημένη Πραγματικότητα, Flutter, Μηχανική Μάθηση, Firebase, Χαρτογράφηση, Serverless Computing, MVC, Cross Platform

Abstract

In the 21st century and on the occasion of the impressive development of technology, humans are trying to simplify his everyday life as much as he can, to save time and to constantly develop their way of life and living in order to cope with the rapid and continuous pace of development. A case in point of saving time and simplifying daily life is the mapping of complex indoor and outdoor spaces. Everyday people wanting to make some needs such as education, medical care, entertainment, etc. visit the respective spaces that offer these possibilities. But many times they are confronted with a chaotic environment of infinite corridors and directions, forcing them to lose precious time, get overworked and in the end miss the essence of their visit until they find the right orientation and destination.

This is where technology comes in, which can offer new and smart features and tools that, if combined properly, can create very effective applications to make these needs. To begin with, in today's world, smartphones are probably the only device that everyone owns, hence the appropriate medium through which the corresponding application should be implemented. Additionally, it would be ideal to use techniques that would make it easy to use and at the same time effective for every person who would have access to it. These come with the Augmented Reality that has entered the life of technology and has greatly changed the way applications are built. It can contribute to the creation of an easy to use and interactive app that guides each user in real time and with the camera open. Finally, there is the addition of Machine Learning, which is no longer missing from any new technological construction, and this is because it can contribute to the improvement of applications and in this case make it able to adapt to changes in the environment, which Augmented Reality uses to guide and create the corresponding paths.

For this reason, the purpose of this thesis is the study, design and development of an iOS application that will guide all potential users through the chaotic indoor and outdoor spaces of the Polytechnic City. To achieve this, several new technological tools were used, such as the Flutter SDK and Firebase, particular build architectures were used and a Incremental Learning model was added to improve its purpose. All this resulted in a standalone application at the research level, which could trigger similar and more developed applications in other areas, necessary for mapping.

Key words

Augmented Reality, Flutter, Machine Learning, Firebase, Mapping, Serverless Computing, MVC, Cross Platform

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα Καθηγητή κ. Παναγιώτη Τσανάκα για την εμπιστοσύνη που μου έδειξε με την ανάθεση αυτής της διπλωματικής εργασίας, καθώς και τον Υποψήφιο Διδάκτορα κ. Διονύσιο Κουλούρη για την πολύτιμη καθοδήγησή του κατά την εκπόνησή της.
Επίσης, θα ήθελα να απευθύνω ένα μεγάλο ευχαριστώ στην οικογένεια και τους κοντινούς μου ανθρώπους που στάθηκαν δίπλα μου καθόλη την διάρκεια των σπουδών μου.

Καλαθάς Δημήτριος,
Αθήνα, 6η Οκτωβρίου 2023

Περιεχόμενα

Ευρετήριο Εικόνων

Λίστα Ακρωνυμίων

3D: Three Dimensional - Τρισδιάστατο
AR: Augmented Reality - Επαυξημένη Πραγματικότητα
ML: Machine Learning - Μηχανική Μάθηση
VR: Virtual Reality - Εικονική Πραγματικότητα
AV: Augmented Virtuality - Επαυξημένη Εικονικότητα
IMU: Inertial Measurement Unit - Μονάδα Αδρανειακής Μέτρησης
CPU: Central Processing Unit - Κεντρική Μονάδα Επεξεργασίας
LAN: Local Area Network - Τοπικό Δίκτυο Υπολογιστών
SDK: Software Development - Κιτ Ανάπτυξης Λογισμικού
VM: Virtual Machine – Εικονική Μηχανή
UI: User Interface - Διεπαφής χρήστη
IDE: Integrated Development Environment - Ολοκληρωμένο Περιβάλλον Ανάπτυξης
API: Application Programming Interface - Διεπαφή Προγραμματισμού Εφαρμογών
GUI: Graphical User Interface - Γραφικό Περιβάλλον Χρήστη
glTF: GL Transmission Forma
GLB: Binary glTF

Κεφάλαιο 1: Θεωρητικό Πλαίσιο

1.1 Εισαγωγή

Ο κόσμος έχει δει πολλές νέες και υποσχόμενες τεχνολογίες κατά την διάρκεια των ετών. Με την ένταξη της τεχνολογίας στην καθημερινότητα των ανθρώπων, έχουν έρθει αμέτρητες εξελίξεις και αναβαθμίσεις στον τρόπο ζωής τους. Αυτή η συνεχόμενη ανάπτυξη στον συγκεκριμένο τομέα έφερε την Επαυξημένη Πραγματικότητα (AR) στο προσκήνιο, που έχει αναδειχθεί από την επιστημονική κοινότητα, ως ένας πολύ ενδιαφέρων και υποσχόμενος τομέας. Ίσως το πιο ριζοσπαστικό που έφερε, είναι η εμπειρία του χρήστη σε συστήματα πλοήγησης αλλά και στον εντοπισμό της θέσης του εντός και εκτός κτιρίων. Ειδικότερα στα κινητά τηλέφωνα, που διαθέτουν και χρησιμοποιούν πολλούς αισθητήρες, όπως η κάμερα, τους επιτρέπει να σαρώνουν τον χώρο και να αναγνωρίζουν το σημείο που βρίσκεται ο χρήστης, εμφανίζοντάς τους 3D μοντέλα που μπορεί να έχουν τοποθετηθεί εκεί για κάποιο συγκεκριμένο σκοπό. Έτσι μπορούν να δημιουργηθούν εφαρμογές όπου η Επαυξημένη Πραγματικότητα θα τους δώσει την δυνατότητα να βοηθούν τους χρήστες, στην γρήγορη και αβίαστη μετακίνησή τους, στην τοποθεσία που θα ήθελαν εντός και εκτός των κτιρίων.

Σε όλα τα παραπάνω έρχεται να προστεθεί και η Μηχανική Μάθηση (ML), η οποία από όταν εμφανίστηκε έχει αλλάξει άρδην το τρόπο και την φιλοσοφία κατασκευής των νέων εφαρμογών. Έτσι, μια πολύ ενδιαφέρουσα προσέγγιση θα ήταν η υποβοήθηση της Επαυξημένης Πραγματικότητας από την Μηχανική Μάθηση πρωτίστως ως προς την βελτιστοποίηση των διαδικασιών της ίδιας της Επαυξημένης Πραγματικότητας και έχοντας ως πρακτικό αποτέλεσμα την βελτίωση της εμπειρίας του χρήστη σε εφαρμογές πλοήγησης και αναγνώρισης θέσης, που είναι και το αντικείμενο της παρούσας διπλωματικής.

1.2 Ορισμός Επαυξημένης Πραγματικότητας

1.2.1 Γενικά

Αρχικά η φράση "Επαυξημένη Πραγματικότητα" θεωρείται ότι επινοήθηκε από τους Tom Caudell και David Mizell. Το 1990, ο Caudell εργαζόταν για την Boeing και σχεδίασε μια ψηφιακή οθόνη που τοποθετείται στο κεφάλι για να βοηθήσει τους εργάτες να συναρμολογούν αεροσκάφη, εμφανίζοντας τα σχέδια του αεροπλάνου στο χώρο του εργοστασίου. Η έννοια της AR, ωστόσο, είναι πολύ παλαιότερη από το 1990. Χρησιμοποιήθηκε στον Β' Παγκόσμιο Πόλεμο, όταν ο βρετανικός στρατός ανέπτυξε το πρόγραμμα Mark VIII Airborne Interception Radar Gunsighting. Το έργο αυτό ανέπτυξε τεχνολογία που εμφάνιζε πληροφορίες ραντάρ στο παρμπρίζ ενός μαχητικού αεροπλάνου και επέτρεπε στον πιλότο να προσδιορίζει, μεταξύ άλλων, αν τα αεροπλάνα που βρίσκονταν κοντά του ήταν φίλοι ή εχθροί [1].

1.2.2 Ορισμός

Ο επίσημος ορισμός που επικρατεί τα τελευταία χρόνια στην επιστημονική κοινότητα είναι:

«Η Επαυξημένη Πραγματικότητα (AR) είναι η ενσωμάτωση ψηφιακών πληροφοριών με το περιβάλλον του χρήστη σε πραγματικό χρόνο. Σε αντίθεση με την Εικονική Πραγματικότητα (VR), η οποία δημιουργεί ένα εντελώς τεχνητό περιβάλλον, οι χρήστες της AR βιώνουν ένα πραγματικό περιβάλλον με παραγόμενες αντιληπτικές πληροφορίες που επικαλύπτονται από πάνω του» [1].

Είναι σημαντικό να τονιστεί ότι υπάρχει διαφορά μεταξύ της Επαυξημένης Πραγματικότητας και της Επαυξημένης Εικονικότητας (AV). Το Συνεχές Πραγματικότητας-Εικονικότητας (Reality-Virtuality Continuum) του Milgram ορίστηκε από τους Paul Milgram και Fumio Kishino για να διαλευκάνει τις συγχύσεις μεταξύ των συγκεκριμένων εννοιών και τι εκφράζει η κάθε μια. Αυτό είναι ένα συνεχές που εκτείνεται μεταξύ του πραγματικού περιβάλλοντος και του εικονικού περιβάλλοντος και περιλαμβάνει την Επαυξημένη Πραγματικότητα και την Επαυξημένη Εικονικότητα στο ενδιάμεσο. Η AR είναι πιο κοντά στον πραγματικό κόσμο και η AV είναι πιο κοντά σε ένα καθαρά εικονικό περιβάλλον, όπως φαίνεται στη ***Εικόνα 1*** [2].

Εικόνα 1: *Reality-Virtuality Continuum*

1.3 Καινοτόμες Εφαρμογές AR

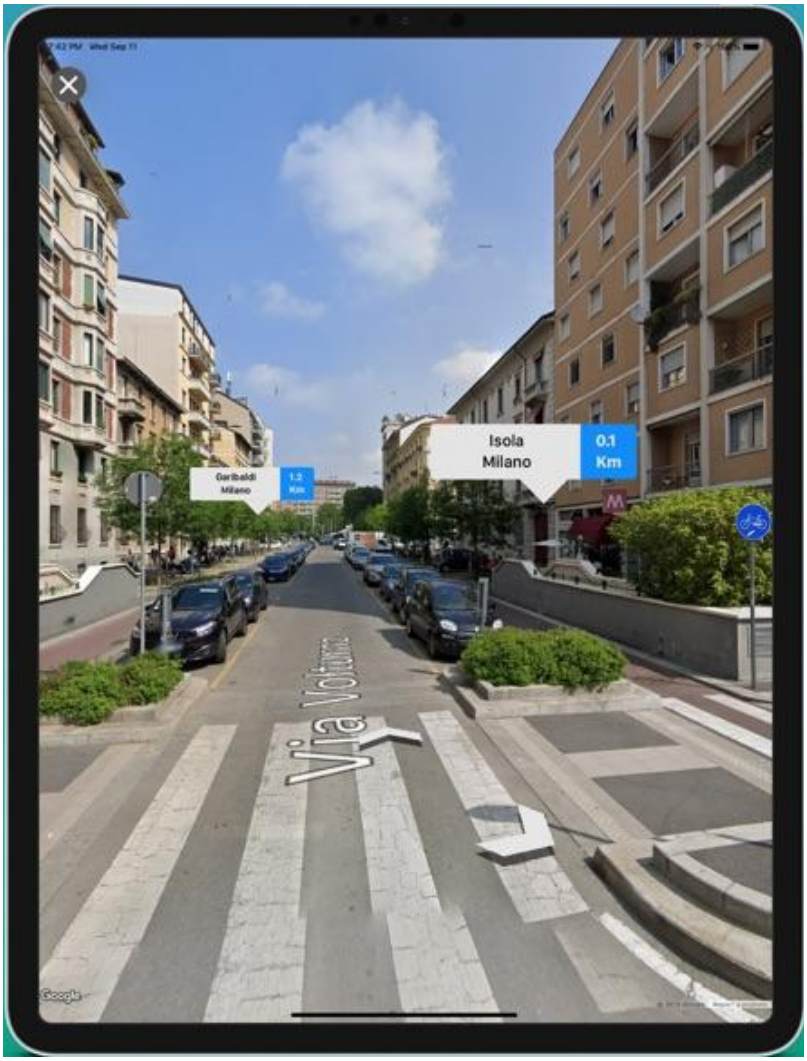
Η τεχνολογική πρόοδος έχει μεταφέρει το AR από το εργαστήριο σε όλους τους τομείς της καθημερινότητας: μάρκετινγκ, ψυχαγωγία, αξιοθέατα, βιομηχανία, μόδα και ιατρική.

Αρχικά το πρώτο σύστημα AR εμφανίστηκε τη δεκαετία του '50, όταν ο Morton Heilig, κινηματογραφιστής, σκέφτηκε ότι ο κινηματογράφος είναι μια δραστηριότητα που θα πρέπει να επιτρέπει στο άτομο να αλληλεπιδρά με το περιβάλλον χρησιμοποιώντας όλες τις αισθήσεις με αποτελεσματικό τρόπο. Το 1962, ο Heilig δημιούργησε και κατοχύρωσε με δίπλωμα έναν προσομοιωτή που ονομαζόταν Sensorama με εικόνες, ήχους, δονήσεις και οσμές. Όμως τότε δεν είχε ακόμα οριστεί ο όρος "Επαυξημένη Πραγματικότητα". Αυτό έγινε όπως προαναφέρθηκε από τον Tom Caudell και David Mizell το 1990, για χάριν της ψηφιακής οθόνης, της Boeing, που τοποθετήθηκε στο κεφάλι των κατασκευαστών για να βοηθήσει στην συναρμολόγηση αεροσκαφών

Σήμερα, με τις νέες εξελίξεις στη μικροηλεκτρονική, παράγεται ένας αυξανόμενος αριθμός συστημάτων και εφαρμογών AR . Η τεχνολογία αυτή έρχεται όλο και πιο κοντά στον χρήστη,

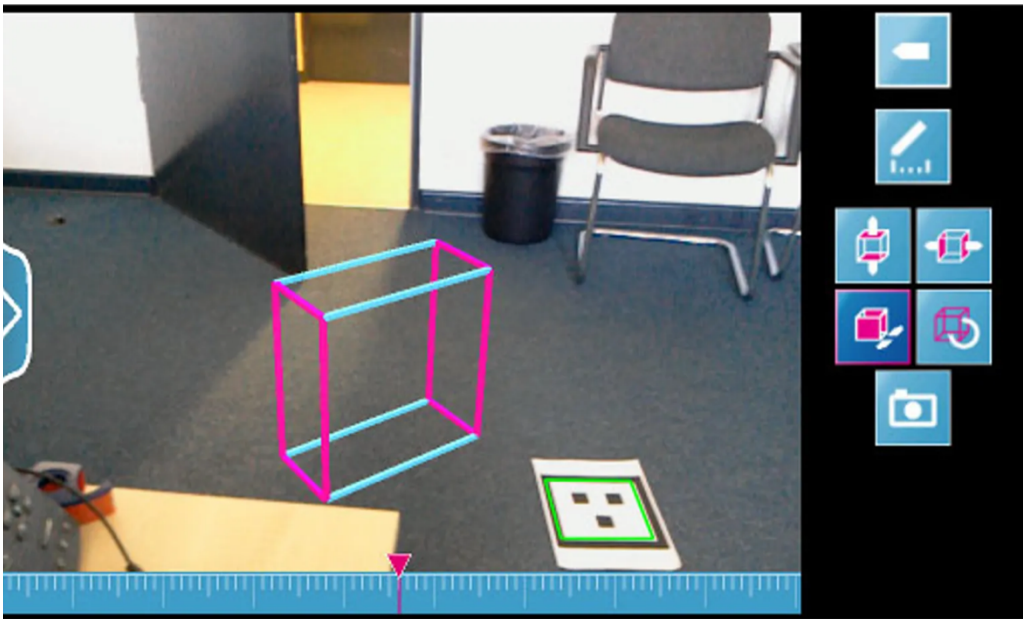
επειδή ακόμη περισσότερες κινητές συσκευές ενσωματώνουν αισθητήρες και διαθέτουν επιδόσεις επεξεργασίας που επιτρέπουν την υλοποίηση συστημάτων AR (π.χ. smartphone, tablet, γυαλιά κ.λπ.). Για παράδειγμα, τα σύγχρονα smartphones διαθέτουν όλα τα απαραίτητα χαρακτηριστικά για διάφορους τύπους οπτικών εφαρμογών AR: βιντεοκάμερα, σύνδεση στο Διαδίκτυο, GPS και Μονάδα Αδρανειακής Μέτρησης (IMU). Ερευνητές παρουσιάζουν πέντε πεδία εφαρμογών AR με βάση το smartphone: (i) αθλητισμός και παιχνίδια, (ii) πολιτιστική κληρονομιά, (iii) ιατρική, (iv) εκπαίδευση και (v) μάρκετινγκ.

Έχουν αναπτυχθεί αρκετές εφαρμογές Επαυξημένης Πραγματικότητας για πλατφόρμες smartphone, ειδικά προσανατολισμένες στην πλοήγηση του χρήστη. Για παράδειγμα, το Milan Metro Augmented Reality είναι μια εφαρμογή πλοήγησης AR που καθοδηγεί τον χρήστη να βρει σταθμούς του μετρό χρησιμοποιώντας την κάμερα και τη θέση του, όπως φαίνεται στην ***Εικόνα 2***. Μια άλλη εφαρμογή πλοήγησης είναι το Junaio Augmented Reality, το οποίο παρέχει διάφορες πληροφορίες σχετικά με τοπικές εκδηλώσεις και προσφορές, οι οποίες εμφανίζονται απευθείας στην εικόνα που καταγράφει η κάμερα του κινητού [3].



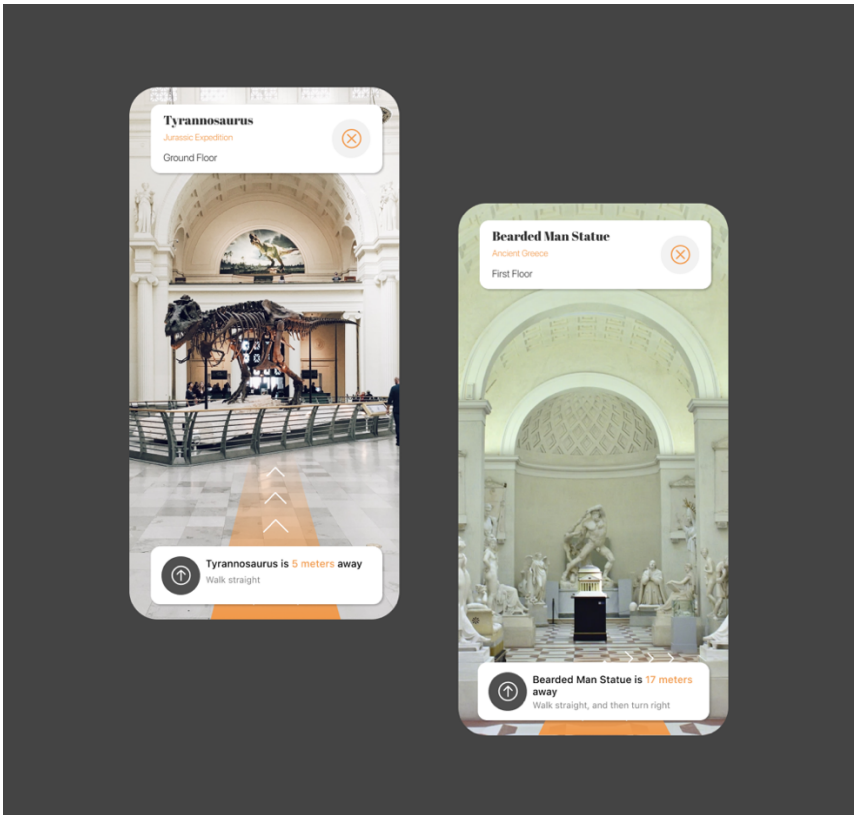
Εικόνα 2: *Milan Metro Augmented Reality*

Επιπρόσθετα, το Happy Measure είναι μια εφαρμογή AR που βοηθά τον χρήστη με τον σχεδιασμό δωματίων για την εσωτερική διακόσμηση και τον εσωτερικό σχεδιασμό. Χρησιμοποιώντας έναν δείκτη που τοποθετείται στο αντικείμενο και την εικόνα που καταγράφεται από την κάμερα του smartphone, η εφαρμογή αυτή επιτρέπει τη μέτρηση του μεγέθους του αντικειμένου. Με αυτόν τον τρόπο, επιτρέπει τη δημιουργία ενός τρισδιάστατου μοντέλου του αντικειμένου σύμφωνα με τις προηγούμενες μετρήσεις. Έτσι, ο χρήστης μπορεί να τοποθετήσει το αντικείμενο που αποτυπώθηκε σε διάφορες θέσεις του δωματίου του, ***Εικόνα 3*** [3].



Εικόνα 3: *Happy Measure*

Σε συνέχεια των παραπάνω, τα μουσεία, όπως η National Gallery of London, χρησιμοποιούν AR για να προσφέρουν εμπλουτισμένες εμπειρίες στους θαμώνες τους, παρέχοντας πρόσθετες πληροφορίες σχετικά με ένα αντικείμενο ή μια έκθεση, ***Εικόνα 4***. Η AR χρησιμοποιείται ακόμη και για την παροχή " τοποθετημένων ντοκιμαντέρ" που αφηγούνται ιστορικά γεγονότα που έλαβαν χώρα στην άμεση περιοχή του χρήστη με την επικάλυψη τρισδιάστατων γραφικών και ήχου σε ό,τι βλέπει και ακούει [1].



Εικόνα 4: *Museum AR Application*

1.4 Μηχανική Μάθηση

1.4.1 Γενικά

Η Τεχνητή Νοημοσύνη (AI) πρωτοεμφανίστηκε το 1950. Ο απώτερος στόχος της ήταν, και παραμένει, η ανάπτυξη της ανθρώπινης νοημοσύνης σε μηχανές, κάτι που μπορεί να πραγματοποιηθεί μέσω αλγορίθμων εκμάθησης που προσπαθούν να μιμηθούν τον τρόπο, με τον οποίο μαθαίνει ο ανθρώπινος εγκέφαλος. Η Μηχανική Μάθηση (ML), που είναι ένας τομέας της Τχνητής Νοημοσύνης, αλλά τα τελευταία χρόνια έχει ανεξαρτητοποιηθεί, είναι υψίστης σημασίας, καθώς επιτρέπει στις μηχανές να αποκτήσουν νοημοσύνη όπως η ανθρώπινη χωρίς ρητό προγραμματισμό. Σύμφωνα με τον Arthur Samuel, η Μηχανική Μάθηση ορίζεται ως το πεδίο μελέτης που δίνει στους υπολογιστές τη δυνατότητα να μαθαίνουν από τα δεδομένα, χωρίς να είναι προγραμματισμένοι [4].

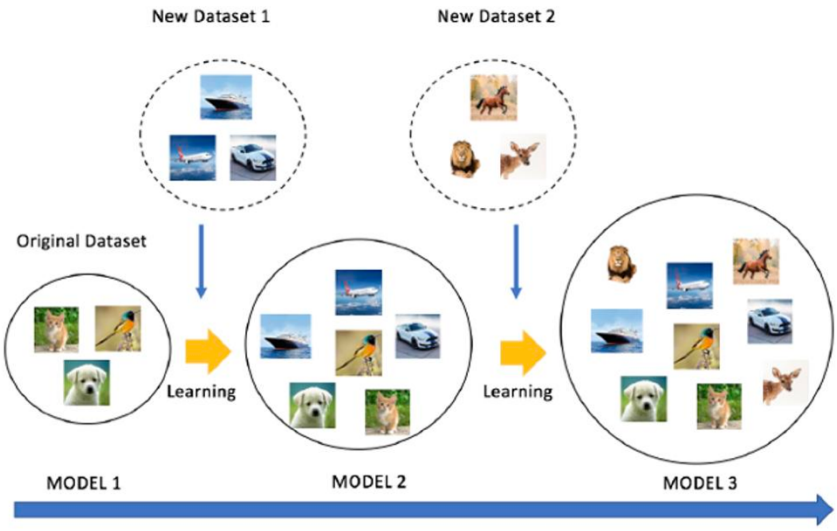
Αναλυτικότερα, τα συστήματα Μηχανικής Μάθησης μπορούν σε γενικές γραμμές να κατηγοριοποιηθούν, μεταξύ άλλων, με τους εξής τρόπους:

- Συστήματα Εποπτευόμενης Μάθησης (Supervised Learning)
- Συστήματα Εκμάθησης Χωρίς Επίβλεψη (Unsupervised Learning)
- Συστήματα Ενισχυτικής Μάθησης (Reinforcement Learning)
- Συστήματα Συστάσεων (Recommender Systems)
- Αυξητική Μάθηση (Incremental Learning)

Γενικότερα, οι τεχνικές Μηχανικής Μάθησης αναδεικνύουν τη χρησιμότητα τους σε περιπτώσεις όπου τα δεδομένα ακολουθούν κάποιο μοτίβο, το οποίο διαφοροποιείται με την πάροδο του χρόνου ή είναι ιδιαίτερα πολύπλοκο, με αποτέλεσμα να χρειάζεται πολυάριθμους υπολογισμούς πέρα από την αντίληψη του ανθρώπινου νου.

1.4.2 Αυξητική Μάθηση (Incremental Learning)

Η Αυξητική Μάθηση είναι μια μεθοδολογία Μηχανικής Μάθησης όπου ένα μοντέλο Τεχνητής Νοημοσύνης μαθαίνει και βελτιώνει τις γνώσεις του προοδευτικά, χωρίς να ξεχνά τις πληροφορίες που έχει αποκτήσει προηγουμένως. Στην ουσία, μιμείται τα ανθρώπινα πρότυπα μάθησης, αποκτώντας νέες πληροφορίες με την πάροδο του χρόνου, διατηρώντας και αξιοποιώντας τις προηγούμενες γνώσεις. Η Αυξητική Μάθηση είναι ζωτικής σημασίας σε σενάρια όπου τα δεδομένα φθάνουν με διαδοχική σειρά ή όπου η αποθήκευση όλων των δεδομένων για επεξεργασία δεν είναι εφικτή.



Εικόνα 5: *Incremental Learning*

1.5 Κατανεμημένα Συστήματα

1.5.1 Γενικά

Ο ρυθμός με τον οποίο αλλάζουν τα συστήματα πληροφορικής ήταν, είναι και συνεχίζει να είναι συγκλονιστικός. Από το 1945, όταν ξεκίνησε η σύγχρονη εποχή των υπολογιστών, μέχρι περίπου το 1985, οι υπολογιστές ήταν μεγάλοι και ακριβοί. Επιπλέον, εξαιτίας της έλλειψης τρόπου σύνδεσής τους, οι υπολογιστές αυτοί λειτουργούσαν ανεξάρτητα ο ένας από τον άλλο.

Ωστόσο, από τα μέσα της δεκαετίας του 1980, δύο τεχνολογικές εξελίξεις άρχισαν να αλλάζουν την κατάσταση αυτή. Η πρώτη ήταν η ανάπτυξη ισχυρών μικροεπεξεργαστών. Επρόκειτο για μηχανές 8-bit, αλλά σύντομα οι CPU 16, 32 και 64-bit έγιναν συνήθεια.

Η δεύτερη εξέλιξη ήταν η εφεύρεση των δικτύων υπολογιστών υψηλής ταχύτητας. Τα τοπικά δίκτυα ή LAN επιτρέπουν τη σύνδεση χιλιάδων μηχανημάτων μέσα σε ένα κτίριο ή μια πανεπιστημιούπολη με τέτοιο τρόπο ώστε μικρές ποσότητες πληροφοριών να μπορούν να μεταφερθούν σε μερικά μικροδευτερόλεπτα.

Το αποτέλεσμα αυτών των τεχνολογιών είναι, ότι όχι μόνο είναι εφικτό αλλά είναι και εύκολο να δημιουργηθεί ένα υπολογιστικό σύστημα που αποτελείται από πολλούς υπολογιστές σε δίκτυο, είτε πρόκειται για μεγάλους είτε για μικρούς. Αυτοί οι υπολογιστές είναι γενικά γεωγραφικά διασκορπισμένοι, για το λόγο αυτό λέγεται συνήθως ότι αποτελούν ένα Κατανεμημένο Σύστημα. Το μέγεθος ενός Κατανεμημένου Συστήματος μπορεί να κυμαίνεται από μια «χούφτα» συσκευές, έως εκατομμύρια υπολογιστές. Το δίκτυο διασύνδεσης μπορεί να είναι ενσύρματο, ασύρματο ή συνδυασμός και των δύο. Επιπλέον, τα Κατανεμημένα Συστήματα είναι συχνά ιδιαίτερα δυναμικά, με την έννοια ότι οι υπολογιστές μπορούν να ενταχθούν και να αποχωρήσουν, με την τοπολογία και την απόδοση του υποκείμενου δικτύου να αλλάζει σχεδόν συνεχώς, ***Εικόνα 6*** [5].

Εικόνα 6: *Κατανεμημένο Σύστημα*

1.5.2 Ορισμός

Στη βιβλιογραφία έχουν δοθεί διάφοροι ορισμοί των Κατανεμημένων Συστημάτων, κανένας από τους οποίους δεν είναι ικανοποιητικός και κανένας δεν συμφωνεί με κανέναν από τους άλλους. Αρκεί να δοθεί ένας γενικός χαρακτηρισμός:

«Ένα Κατανεμημένο Σύστημα είναι μια συλλογή αυτόνομων υπολογιστικών στοιχείων που εμφανίζεται στους χρήστες του ως ένα ενιαίο συνεκτικό σύστημα.»

Ο ορισμός του αναφέρεται σε δύο χαρακτηριστικά γνωρίσματα των Κατανεμημένων Συστημάτων. Το πρώτο είναι ότι ένα Κατανεμημένο Σύστημα είναι μια συλλογή υπολογιστικών στοιχείων που το καθένα μπορεί να συμπεριφέρεται ανεξάρτητα το ένα από το άλλο. Ένα υπολογιστικό στοιχείο μπορεί να είναι είτε μια συσκευή υλικού είτε μια διεργασία λογισμικού. Ένα δεύτερο στοιχείο είναι ότι οι χρήστες (είτε πρόκειται για ανθρώπους είτε για εφαρμογές) πιστεύουν ότι έχουν να κάνουν με ένα ενιαίο σύστημα [5].

1.6 Γενική ανάλυση εφαρμογής - Σκοπός

Όπως έχει ήδη αναφερθεί, η εποχή που διανύουμε χαρακτηρίζεται από την ραγδαία ανάπτυξη και την συνεχόμενη εμφάνιση νέων τεχνολογιών όπως, η Επαυξημένη Πραγματικότητα, Μηχανική Μάθηση, τα Κατανεμημένα Συστήματα κτλ.

Η πρόκληση άλλα και ο στόχος της παρούσας διπλωματικής είναι να καταφέρει να συνδυάσει όλα τα παραπάνω, έτσι ώστε να δημιουργεί μια αυτόνομη εφαρμογή, για κινητά τηλέφωνα, η οποία με την βοήθεια της AR θα κατευθύνει τους φοιτητές στην Πολυτεχνειούπολη. Η εφαρμογή θα είναι διαθέσιμη σε πάνω από ένα λογισμικό (Cross Platforms) και θα μπορεί να είναι λειτουργική σε εξωτερικούς και εσωτερικούς χώρους. Επίσης ενσωματώνει και ένα μοντέλο Incremental Learning, έτσι ώστε οι χρήστες να βοηθούν στην εξέλιξη της λειτουργίας της εφαρμογής και να βελτιώνεται συνεχώς η εμπειρία τους. Τέλος, μιας και μπορούν ταυτόχρονα να συνδεθούν πολλά smartphones, τα οποία αλληλεπιδρούν με μια κεντρική βάση δεδομένων, η κατασκευή αυτή, ανήκει στην οικογένεια των Κατανεμημένων Συστημάτων.

Ο βασικός σκοπός της είναι, πρώτον να διευκολύνει την μετακίνηση των φοιτητών και ειδικότερα των νέων, με την χαρτογράφηση των χαοτικών εσωτερικών και εξωτερικών χώρων του πολυτεχνείου, αλλά και να μπορέσει να συνδέσει την Τεχνητή Νοημοσύνη για να εξελίξει τους σκοπούς της.

Για να συμβούν τα παραπάνω δημιουργήθηκαν δύο παράλληλες εφαρμογές, μια που αφορά τον διαχειριστή και μια που αφορά τον χρήστη. Η πρώτη έχει ως στόχο την καταγραφή και την χαρτογράφηση του περιβάλλοντος, έτσι ώστε να χρησιμοποιηθεί στην δεύτερη, που αφορά το ευρύ φοιτητικό, και όχι μόνο κοινό.

Τέλος, μια προσέγγιση εφαρμογής πλοήγησης για εσωτερικό χώρο με τη χρήση Επαυξημένης Πραγματικότητας έχει εγκατασταθεί και εκτελείται πιλοτικά. Έχει αποδειχθεί ότι έχει αποτελέσματα στην πλοήγηση αλλά επηρεάζεται από τις αλλαγές στον χώρο. Στην παρούσα εργασία, επεκτείνεται η συγκεκριμένη ιδέα με την προσθήκη μεθόδων για την αντιμετώπιση αυτών των περιορισμών, αλλά και την προσθήκη δυνατότητας συνδυασμού εσωτερικού και εξωτερικού χώρου [6].

Κεφάλαιο 2: Τεχνολογικό Υπόβαθρο

Οι τεχνολογίες αιχμής που θα χρησιμοποιηθούν στην παρούσα διπλωματική απαιτούν την πρακτική κατανόηση ενός τεχνολογικού υποβάθρου το οποίο περιλαμβάνει συγκεκριμένη ορολογία και τεχνικές. Η Επαυξημένη Πραγματικότητα αποτελεί μια υποκατηγορία της Υπολογιστικής Όρασης και ο συνδυασμός της με την Μηχανική Μάθηση σε ανώτερο επίπεδο δημιουργεί την ανάγκη για την ανάπτυξη ειδικών αλγορίθμων, όπως αυτός που θα παρουσιαστεί στην συγκεκριμένη Θέση.

2.1 Επαυξημένη Πραγματικότητα

2.1.1 Anchors

Πολύ σημαντική για την κατανόηση και δημιουργία της εφαρμογής είναι η σημασία του όρου των Anchors. Ένα Anchor είναι ένα σημείο στο πραγματικό περιβάλλον που ακινητοποιείται ή τοποθετείται και χρησιμοποιείται στην Επαυξημένη Πραγματικότητα για να συνδέσει εικονικά αντικείμενα ή πληροφορίες με αυτό το σημείο. Τα Anchors είναι σημαντικοί για τη δημιουργία σταθερών και συνεκτικών AR εμπειριών.

Ορισμένες σημαντικές πληροφορίες σχετικά με τα Anchors περιλαμβάνουν:

- **Σύνδεση εικονικού με πραγματικού:** Τα Anchors χρησιμοποιούνται για να τοποθετήσουν εικονικά αντικείμενα ή πληροφορίες σε συγκεκριμένα σημεία του πραγματικού κόσμου. Αυτό επιτρέπει στα εικονικά αντικείμενα να διατηρούνται σε σταθερή θέση ακόμη και όταν ο χρήστης κινεί τη συσκευή του AR.

- **Κατανόηση του περιβάλλοντος:** Τα Anchors βοηθούν την εφαρμογή AR να κατανοήσει το περιβάλλον του χρήστη. Αυτά τα σημεία λειτουργούν ως σημεία αναφοράς για τον υπολογισμό των θέσεων και των κατευθύνσεων των εικονικών αντικειμένων.

- **Διασύνδεση με πραγματικά δεδομένα:** Τα Anchors μπορούν να συνδέσουν εικονικό περιεχόμενο με πραγματικά δεδομένα, όπως γεωγραφικές τοποθεσίες, αναγνωριστικά αντικειμένων, εικόνες ή πληροφορίες από πραγματικούς αισθητήρες.

- **Επαυξημένη Πραγματικότητα και πλεονεκτήματα οπτικής ανίχνευσης:** Τα Anchors είναι σημαντικά για εφαρμογές AR που χρησιμοποιούν οπτική ανίχνευση (όπως ανίχνευση εικόνας) για τον εντοπισμό και την αναγνώριση αντικειμένων στο περιβάλλον.

Τα Anchors διαδραματίζουν κρίσιμο ρόλο στη δημιουργία επικοινωνίας μεταξύ του πραγματικού και του εικονικού κόσμου σε εφαρμογές AR, προσφέροντας σταθερότητα και συνέχεια στην εμπειρία του χρήστη.

2.1.2 Anchors / Nodes

Είναι επίσης ζωτικής σημασίας να αναφερθούν οι διάφορες μεταξύ των εννοιών (σημασιολογικά και πρακτικά στην εφαρμογή) Anchors και Nodes έτσι ώστε να γίνει απόλυτα ξεκάθαρος ο σκοπός και ο λόγος ύπαρξης τους σε μια εφαρμογή AR.

Οι κύριες διαφορές μεταξύ αυτών των δύο:

Anchors:

- Τα Anchors είναι αναγνωρισμένα από το σύστημα σημεία του πραγματικού κόσμου τα οποία χρησιμοποιούνται για να τοποθετηθούν εικονικά αντικείμενα σε συγκεκριμένες θέσεις. Αυτά τα σημεία είναι σταθερά και διατηρούνται στην ίδια θέση ακόμα και όταν ο χρήστης κινεί τη συσκευή του (πχ αλλαγή δωματίου).

- Τα Anchors είναι αόρατα.

Nodes:

- Τα Nodes είναι εικονικά αντικείμενα ή οντότητες που τοποθετούνται στον κόσμο του AR. Αυτά τα αντικείμενα είναι ενεργά και αλληλεπιδρούν με το χρήστη και το περιβάλλον AR.

- Τα Nodes μπορούν να είναι 3D μοντέλα, εικόνες, κείμενο, βίντεο, ή οποιοδήποτε άλλο εικονικό περιεχόμενο που μπορεί να προστεθεί στο περιβάλλον AR.

- Τα Nodes είναι διαδραστικά και μπορούν να χρησιμοποιηθούν για να αντιδρούν στην κίνηση του χρήστη, να εκτελούν εφέ, να εμφανίζουν πληροφορίες και άλλες λειτουργίες.

Σε απλά λόγια, τα Anchors είναι ασφαλείς σημεία στον πραγματικό κόσμο που χρησιμοποιούνται για να συνδέσουν το AR με τον πραγματικό κόσμο, ενώ τα Nodes αποτελούν το εικονικό περιεχόμενο που προστίθεται στο Anchor και μπορούν να είναι διαδραστικά και εκδηλωτικά. Ιεραρχικά, ένα Node προϋποθέτει την τοποθέτηση ενός Anchor για να εμφανιστεί.

2.1.3 SLAM & Cloud Anchors

Η αναγνώριση και ο εντοπισμός τοπικής περιοχής επιτυγχάνεται με την εξαγωγή χαρακτηριστικών μιας εικόνας μαζί με δεδομένα από τους αισθητήρες IMU της συσκευής. Αρχικά, η περιοχή γύρω από τη συσκευή πρέπει να σαρώνεται αργά και σταθερά για να εφαρμοστεί ο αλγόριθμος SLAM.

Λειτουργία αλγορίθμου SLAM:

- Αναλύονται καρέ-καρέ της κάμερας και εξάγονται σημεία χαρακτηριστικών από το καθένα.

- Βελτιστοποίηση, ώστε να εστιάζουν σε συγκεκριμένα και πυκνά τμήματα κάθε εικόνας για να επιτυγχάνεται καλύτερη επεξεργασία των δεδομένων.

- Τα χαρακτηριστικά που εξάγονται συνδυάζονται με δεδομένα από τους αισθητήρες IMU για τον ακριβή προσδιορισμό της απόστασης, της περιστροφής και του προσανατολισμού των καταγεγραμμένων καρέ.

- Δημιουργείται στην έξοδο ένα **Feature Map**: 3D Map με συνδυάστηκα δεδομένα από χαρακτηριστικά των καρέ και IMU αισθητήρες.

Αποτέλεσμα:

Τα τμήματα από τα καρέ, που έχουν μια σίγουρη ποσότητα χαρακτηριστικών προσφέρουν τη δυνατότητα τοποθέτησης Anchors στη σκηνή, λαμβάνοντας υπόψη ότι τα κοντινά χαρακτηριστικά ενός Anchor μπορούν εύκολα να αναγνωριστούν μόλις η συσκευή φτάσει στο σημείο αυτό.

Cloud Anchors & SLAM

Η λειτουργία Cloud Anchors χρησιμοποιεί την έξοδο SLAM ως βάση για την αποθήκευση των θέσεων των Anchors για απομακρυσμένη χρήση. Στόχος των Cloud Anchors είναι η δυνατότητα προεπισκόπησης των Anchors που τοποθετήθηκαν από άλλη συσκευή στο παρελθόν. Ως εκ τούτου, τα σημεία χαρακτηριστικών στην εμβέλεια ενός Anchors συλλαμβάνονται και αποθηκεύονται σε μια βάση δεδομένων νέφους (cloud database). Παρέχοντας ένα καρέ κάμερας από την ίδια περιοχή και συγκρίνοντας τα τρέχοντα εξαγόμενα χαρακτηριστικά με τη βάση δεδομένων, μπορεί να εκτιμηθεί, με ακρίβεια, η ακριβής θέση ενός Anchor. Ο περιορισμός της προαναφερθείσας δυνατότητας είναι ο περιορισμένος αριθμός των Cloud Anchors που μπορούν να αναζητηθούν ταυτόχρονα.

2.1.4 Geospatial Anchors

Τα Geospatial Anchors είναι ένας ειδικός τύπος Anchor. Είναι ένα σημαντικό χαρακτηριστικό σε εφαρμογές Επαυξημένης Πραγματικότητας, που επιτρέπουν την τοποθέτηση εικονικών

αντικειμένων σε συγκεκριμένες γεωγραφικές θέσεις στον πραγματικό κόσμο, χρησιμοποιώντας τις συντεταγμένες (latitude, longitude, altitude) για τοποθέτηση.

Οι συντεταγμένες αυτές:

Latitude (Γεωγραφικό πλάτος): Είναι η απόσταση βόρεια ή νότια του ισημερινού (έκτασης 0 μοιρών) προς τα βόρεια ή νότια του πόλου (έκτασης 90 μοιρών βόρεια ή νότια). Συνήθως μετριέται σε μοίρες.

Longitude (Γεωγραφικό μήκος): Είναι η απόσταση ανατολικά ή δυτικά του μεσημβρινού (μεσημβρινής έκτασης 0 μοιρών) προς ανατολικά ή δυτικά του μεσημβρινού. Η μέτρηση γίνεται σε μοίρες.

Altitude (Υψόμετρο): Είναι η απόσταση πάνω ή κάτω από την επιφάνεια της γης. Συνήθως μετριέται σε μέτρα ή πόδια.

Με τις συντεταγμένες αυτές, μπορεί να καθοριστεί με ακρίβεια η θέση ενός Geospatial Anchors. Αυτά τα Anchors είναι χρήσιμα σε πολλές εφαρμογές, όπως πλοηγητές GPS, AR εφαρμογές που χρησιμοποιούν γεωχωρική πληροφορία, και άλλες που απαιτούν τοποθέτηση και ευαισθησία στις γεωγραφικές συντεταγμένες.

2.2 Αλγόριθμος Incremental Learning

Απαραίτητη για την έξυπνη λειτουργία εκμάθησης χώρου του προτεινόμενου συστήματος είναι η ενσωμάτωση αλγορίθμων Μηχανικής Μάθησης. Επιλέχθηκε η προσέγγιση της Αυξητικής Εκμάθησης ή Incremental Learning. Όπως αναφέρθηκε, αφορά τον τρόπο με τον οποίο μια εφαρμογή θα βελτιώνει την λειτουργία της κατά την διαδικασία επίτευξης του σκοπού της. Με άλλα λόγια, στην περίπτωση της παρούσας διπλωματικής, να βελτιώνεται η αποτελεσματικότητα της πλοήγησης κατα τη διάρκεια της εκτέλεσης της εφαρμογής από τον επισκέπτη.

Αρχικά, για την τοποθέτηση ενός Cloud Anchor πρέπει να έχει γίνει καλή σάρωση του χώρου με την κάμερα, έτσι ώστε να δημιουργηθεί όσο το δυνατόν καλύτερο Feature Map που θα αντιστοιχεί στο χώρο γύρω από αυτό. Κατά την επανατοποθέτηση του πρέπει το Feature Map να αντιστοιχηθεί με τον χώρο που καταγράφει η κάμερα εκείνη την στιγμή και στη συνέχεια να εμφανιστεί το Cloud Anchor με το 3D μοντέλο του. Όμως, η διαδικασία της αντιστοίχισης των δύο αυτών Feature Maps δύναται να έχει αρκετό χρόνο απόκρισης, διότι ο χώρος μπορεί να έχει υποστεί σημαντικές αλλαγές, οι οποίες μεταβάλλουν αρκετά το τρέχον Feature Map (πχ αλλαγή χρώματος τοίχου, μετακίνηση αντικειμένων, εμπόδια στην κάμερα κτλ).

Έτσι, ο στόχος του αλγορίθμου είναι ο εντοπισμός αυτών των αλλαγών στον χώρο και η επανα προπόνηση του συστήματος με βάση τις παρούσες αλλαγές. Η εφαρμογή την πρώτη φορά που θα εντοπιστούν αυτές οι αλλαγές ενδέχεται να δυσκολευτεί στην αντιστοίχιση των Feature Maps, όμως τις επόμενες φορές, με τη χρήση δεδομένων από νεότερες προπονήσεις, θα παραχθεί καλύτερο χρονικό αποτέλεσμα.

Για να επιτευχθεί αυτό, δόθηκε στον επισκέπτη η δυνατότητα επανατοποθέτησης Cloud Anchor στο ίδιο ακριβώς σημείο και με το ίδιο 3D μοντέλο, καθώς ακολουθεί το μονοπάτι της πλοήγησης. Το βασικό πλεονέκτημα αυτής της διαδικασίας, είναι ότι το νέο Cloud Anchor χαρακτηρίζεται με Feature Map από την καταγραφή του χρήστη με την κάμερα εκείνη την στιγμή, άρα αντιστοιχεί στη πιο πρόσφατη μορφή του χώρου. Με αυτόν τον τρόπο, το κάθε κύριο σημείο (Checkpoint) μπορεί να έχει πολλά Cloud Anchors με διαφορετικά Feature Map που το κάθε ένα θα εκπροσωπεί τον ίδιο χώρο αλλά διαφορετική στιγμή του.

Σημαντικό είναι το γεγονός ότι κατά την διαδικασία της αντιστοίχισης Feature Maps, η εφαρμογή δεν προσπαθεί να φέρει όλα τα Cloud Anchors ενός Checkpoint, αλλά μόνο εκείνα τα οποία έχουν υψηλό δείκτη επιτυχίας. Η μετρική αυτή καθορίζεται από ένα σύνολο βαρών, τα οποία αντιστοιχούν σε κάθε Cloud Anchor και αυξάνονται όταν το Anchor αυτό εντοπίζεται. Αυτομάτως, όπως γίνεται εύκολα αντιληπτό, Cloud Anchors με μηδενικό ή χαμηλό βάρος θεωρούνται ως μη ορθώς τοποθετημένα και διαγράφονται από το σύστημα μετά από ένα χρονικό διάστημα.

Η ερευνητική αξία του αλγορίθμου αυτού είναι ο συνδυασμός της Μηχανικής Μάθησης με την Επαυξημένη Πραγματικότητα. Δεν εκτελείται κάποιος αλγόριθμος για την μεταβολή δεδομένων λειτουργίας του AR, αλλά προστίθεται ένα επιπλέον επίπεδο Τεχνητής Νοημοσύνης πάνω από το επίπεδο του AR με σκοπό την βελτιστοποίηση των διαδικασιών. Η Αυξητική Μάθηση πιθανώς αυξάνει την πολυπλοκότητα του συστήματος και απαιτεί περισσότερους υπολογιστικούς πόρους αλλά έχει ως αποτέλεσμα την βελτιστοποίηση της λειτουργίας του συστήματος και δίνει μια κατεύθυνση προς τη λύση του προβλήματος μεταβολών των δεδομένων SLAM.

Κεφάλαιο 3: Αρχιτεκτονική και Υλοποίηση Συστήματος

Το σύστημα χωρίζεται σε 3 οντότητες:

- Εφαρμογή Διαχειριστή
- Εφαρμογή Χρήστη
- Εφαρμογή Backend

Για να μπορεί να θεωρηθεί λειτουργική η εφαρμογή θα πρέπει και οι τρεις να συνδυάζονται σωστά και να αλληλεπιδρούν μεταξύ τους, λαμβάνοντας υπόψιν ότι θα μπορούν πολλοί χρήστες να είναι ταυτόχρονα συνδεδεμένοι. Επίσης, καθ' όλη την χρήση της εφαρμογής, θα πρέπει να υπάρχει σύνδεση στο διαδίκτυο.

3.1 Αρχιτεκτονική

3.1.1 Εφαρμογή Διαχειριστή (Admin)

Το κομμάτι της εφαρμογής που αφορά τον διαχειριστή, έχει σκοπό να χαρτογραφήσει την περιοχή. Με άλλα λόγια ο admin δημιουργεί ένα μονοπάτι από 3D Nodes πάνω σε Cloud Anchors. Για να το κάνει αυτό, πρέπει να επιλέξει τον τελικό προορισμό, έτσι ώστε η διαδρομή που θα φτιάξει να εμφανίζεται μόνο όταν ο χρήστης κάνει την ίδια επιλογή. Η διαδικασία που ακολουθεί ο admin είναι, αφού επιλέξει την τελική τοποθεσία του,να ανοίγει αυτόματα η κάμερα και να τοποθετήσει τα απαραίτητα 3D μοντέλα στα σημεία που θέλει, αφού πρώτα κάνει πολύ καλή καταγραφή του περιβάλλοντος χώρου, μιας και το Cloud Anchor στο οποία τοποθετείται «πάνω» το 3D Node εξαρτάται από το γύρο χώρο και ένα χαρτί χαρακτηριστικών (Feature Map) που δημιουργείται αυτόματα για αυτόν. Τα συγκεκριμένα στοιχεία από το κάθε Cloud Anchor «ανεβαίνουν» στην βάση για να τα χρησιμοποιήσει ο χρήστης. Όλη αυτή η διαδικασία αφορά τους εσωτερικούς χώρους. Όσον αφορά τους εξωτερικούς, ο admin πρέπει να ενημερώσει την βάση δεδομένων με τις ακριβείς συντεταγμένες που θελει να εμφανιστούν τα 3D μοντέλα και να ορίσει κατάλληλα κάποια πεδία που ενώνουν την εξωτερική διαδρομή με την εσωτερική, μιας και ο σκοπός είναι με αρχή έναν εξωτερικό (χώρου) σημείο ο χρήστης να καταλήγει σε ένα εσωτερικό (χώρου) σημείο. Στην συγκεκριμένη περίπτωση, αφού η εφαρμογή αφορά το ΕΜΠ αρχικό εξωτερικό σημείο θα μπορεί να είναι η κεντρική πλατεία.

3.1.2 Εφαρμογή Χρήστη (User)

Το κομμάτι της εφαρμογής αφορά τον χρήστη, έχει σκοπό να επιλέξει τον τελικό προορισμό και έπειτα αφού κάνει μια καλή καταγραφή του χώρου με την κάμερα να «κατεβάσει» από την βάση τα στοιχεία για τα Cloud Anchors που φέρουν και τα 3D μοντέλα μαζί. Αυτό γίνεται πρώτα σε εξωτερικό χώρο και αφού φτάσει κοντά στο κτίριο που είναι ο τελικός στόχος η εφαρμογή το αντιλαμβάνεται και συνεχίζει την διαδικασία που αφορά τους εσωτερικούς χώρους. Όταν φτάσει στο προορισμό πάλι η εφαρμογή το καταλαβαίνει και βγάζει κάποιες πληροφορίες για το σημείο αυτό. Τέλος ο χρήστης έχει την δυνατότητα στους εσωτερικούς χώρους, αφού ηθελημένα έχει ενεργοποιήσει μια λειτουργία, πατώντας πάνω σε ένα 3D μοντέλο να προσθέσει ένα Cloud Anchor με νέο Feature Map το οποία να είχε ίσως μια καλύτερη καταγραφή του χώρου και να ανεβάσει στην βάση τα στοιχεία του, έτσι την επόμενη φορά που θα «τρέξει» για την ίδια εσωτερική διαδρομή να «κατεβάσει» τα Cloud Anchors με το καλύτερο Feature Map που ταιριάζει με το χώρο την δεδομένη στιγμή.

3.1.3 Εφαρμογή Backend

Η εφαρμογή έχει μια κεντρική βάση δεδομένων στο cloud, η οποία επικοινωνεί ταυτόχρονα και με την εφαρμογή του admin, αλλά και με την εφαρμογή του user. Το ιδιαίτερο που έχει, είναι ότι και οι δύο παράλληλες εφαρμογές αλληλεπιδρούν με αυτήν συνεχόμενα, δηλαδή γίνονται πολλές ενημερώσεις, προσθήκες αλλά και διάγραφες, και από τις δυο, οποιαδήποτε χρονική στιγμή.

3.2 Τεχνολογίες Προγραμματισμού

Δεδομένης της συνεχόμενης ανάπτυξης στο χώρο των υπολογιστών, ο άνθρωπος προσπαθεί να ενσωματώσει όλο και περισσότερες λειτουργίες σε όσο γίνεται μικρότερες φορητές συσκευές. Με άλλα λόγια, εξελίσσει καθημερινά τα κινητά τηλέφωνα, έτσι ώστε να είναι σε θέση να καλύψουν οποιαδήποτε ανάγκη έχουν οι χρήστες σε καθημερινό και όχι μόνο επίπεδο.

Για να το πετύχει, η εξέλιξη επικεντρώθηκε στον χώρο των smartphones, που ομολογουμένως είναι η συσκευή που σχεδόν κάθε άνθρωπος κατέχει. Αναπτύχθηκαν τεχνολογίες που διευκόλυναν τους επιστήμονες να δημιουργήσουν εφαρμογές που καλύπτουν και με το παραπάνω τις διαρκώς αυξανόμενες ανάγκες των χρηστών. Αξίζει να αναφερθεί ότι, εάν ένας κατασκευαστής-προγραμματιστής εργάζεται σ ’ένα περιβάλλον που είναι πολύ εξελιγμένο και εύκολο στην εκμάθηση του, γίνεται πιο αποτελεσματικός και οι ιδέες που έχει για την δημιουργία εφαρμογών γίνονται υλοποιήσιμες.

Έτσι στην παρούσα διπλωματική χρησιμοποιήθηκαν σύγχρονα τεχνολογικά εργαλεία για την δημιουργία της καινοτόμας εφαρμογής που αναφέρθηκε παραπάνω. Τα συγκεκριμένα μέσα βοήθησαν το έργο να κατασκευαστεί πιο εύκολα, αφού αναφερόμαστε σε high-level τεχνολογίες, που έχουν σαν βασικό στόχο να διευκολύνουν τον προγραμματιστή και να τον καθοδηγούν καθόλα την διάρκεια της κατασκευής της εφαρμογής του.

3.2.1 Flutter

Για την ανάπτυξη της εφαρμογής επιλέχθηκε το SDK (Software Development Kit) Flutter. Το Flutter είναι μια πλατφόρμα ανοιχτού κώδικα που δημιουργήθηκε από την Google και αποτελεί την ένωση του UI Framework, με το SDK. Χρησιμοποιείται για την ανάπτυξη διεπαφών χρήστη πολλαπλών πλατφορμών (Cross Platform) και έχει σχεδιαστεί για να επιτρέπει την επαναχρησιμοποίηση κώδικα σε διάφορα λειτουργικά συστήματα όπως το Android, το iOS, τα Linux, τα Windows και το web, ενώ επιτρέπει επίσης στις εφαρμογές να επικοινωνούν απευθείας με τις υπηρεσίες της υποκείμενης πλατφόρμας. Στόχος είναι να επιτρέπει στους προγραμματιστές να αναπτύσσουν πολύπλοκες εφαρμογές υψηλών επιδόσεων, που μπορούν να εκτελεστούν σε διαφορετικές πλατφόρμες, αφομοιώνοντας τις μεταξύ τους διαφορές, ενώ παράλληλα μοιράζονται μια κοινή βάση κώδικα.

Η πλατφόρμα του Flutter βασίζεται στην γλώσσα προγραμματισμού Dart. Η Dart είναι μια client-optimized γλώσσα προγραμματισμού. Αναπτύσσεται από την Google και χρησιμοποιείται για την κατασκευή mobile, desktop, server και web εφαρμογών. Είναι αντικειμενοστραφής και το συντακτικό της είναι παρόμοιο με αυτό της γλώσσας προγραμματισμού C. Η Dart μπορεί να μεταγλωττιστεί σε native code ή σε JavaScript.

Κατά την διάρκεια της ανάπτυξης, οι Flutter εφαρμογές τρέχουν σε μια εικονική μηχανή (VM) που προσφέρει σταθερή ενσωμάτωση στις αλλαγές (hot reload) χωρίς να απαιτείται διαρκώς πλήρης μεταγλώττιση. Οι Flutter εφαρμογές μεταγλωτίζονται κατευθείαν σε γλώσσα μηχανής, είτε σε Intel x64 ή ARM εντολές, είτε σε Javascript εάν προορίζονται για το web. Η πλατφόρμα είναι ανοιχτού κώδικα, με ανεκτική άδεια BSD, και υποστηρίζεται από μια ταχέως αναπτυσσόμενη συλλογή πακέτων τρίτων που επεκτείνουν την βασική λειτουργικότητα της βιβλιοθήκης

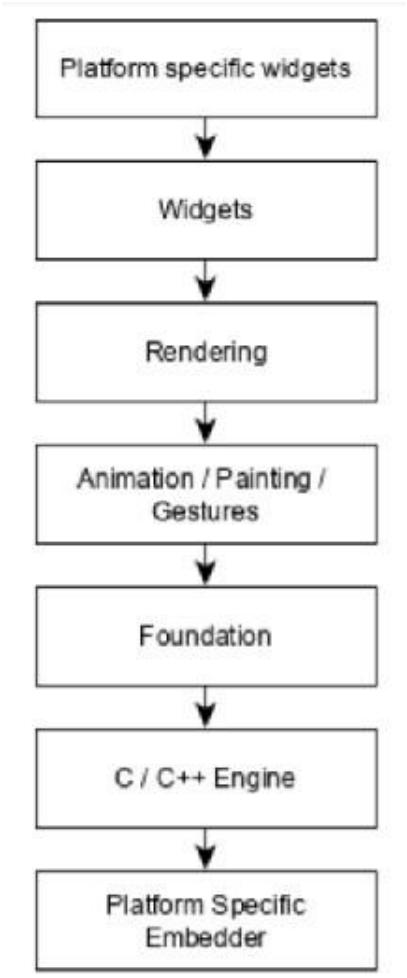
Το Flutter παρέχει πολλά έτοιμα για χρήση widgets (μονάδες σύνθεσης), ικανά να δημιουργήσουν μια όμορφη, μοντέρνα εφαρμογή. Τα πάντα στο Flutter μπορούν να υλοποιηθούν χρησιμοποιώντα widgets. Ο προσανατολισμός, η διάταξη, τα animations, τα κουμπιά κλπ υλοποιούνται με widgets. Πιο συγκεκριμένα και η ίδια η Flutter εφαρμογή είναι ένα widget, το root widget (συνήθως MaterialApp ή CupertinoApp). Η λογική της εφαρμογής στηρίζεται στο reactive programming. Το widget μπορεί προαιρετικά να έχει μια κατάσταση (state). Αλλάζοντας το state του widget, το Flutter θα συγκρίνει αυτόματα (reactive programming) το state του widget (παλιό και νέο) και θα το απεικονίσει μόνο με τις απαραίτητες αλλαγές αντί να επαναπεικονίσει ολόκληρο το widget. Τα widgets σχηματίζουν μια ιεραρχία με βάση τη σύνθεση. Κάθε widget τοποθετείται μέσα στον γονέα του και μπορεί να λαμβάνει το περιβάλλον από αυτόν. Αυτή η δομή μεταφέρεται μέχρι το root widget.

Τρία βασικά χαρακτηριστικά της αρχιτεκτονικής του Flutter που αξίζει να αναφερθούν είναι τα εξής:

- Gestures: Τα Flutter widgets υποστηρίζουν την αλληλεπίδραση με τον χρήστη μέσω ενός ειδικού widget που ονομάζεται GestureDetector. Το GestureDetector είναι ένα αόρατο widget που έχει την ικανότητα να αντιλαμβάνεται τις κινήσεις του χρήστη (tapping, dragging κλπ), στο παιδί widget. Πολλά native widgets υποστηρίζουν την αλληλεπίδραση μέσω της χρήσης του GestureDetector. Μπορούμε επίσης να ενσωματώσουμε διαδραστικά χαρακτηριστικά σε ένα υπάρχον widget, συνθέτοντας το με το GestureDetector widget.
- Concept of State: Τα widgets του Flutter υποστηρίζουν την έννοια του state, παρέχοντας ένα ειδικό widget που ονομάζεται StatefulWidget. Για να μπορεί ένα widget να υποστηρίξει την λογική του state πρέπει να παραχθεί από το StatefulWidget, αλλιώς από το StatelessWidget. Το StatefullWidget θα αυτο-επαναπεικονιστεί οποτεδήποτε το εσωτερικό του state αλλάξει. Η διαδικασία αυτή βελτιστοποιείται εντοπίζοντας τις διαφορές μεταξύ του παλιού και του νέου widget UI και επαναπεικονίζει μόνο τις απαραίτητες αλλαγές.

- Layers: Η πιο σημαντική ιδέα του Flatter framework είναι ότι είναι ξεκάθαρα οργανωμένο σε επίπεδα μειωμένης πολυπλοκότητας. Ένα επίπεδο είναι συνδεδεμένο με το αμέσως επόμενο του επίπεδο. Το πρώτο στην ιεραρχία είναι το widget που εξαρτάται από την πλατφόρμα (android, iOS κλπ). Το επόμενο είναι το layer που περιλαμβάνει όλα τα widgets του Flutter. Ακολουθεί το Rendering layer, που είναι ένα χαμηλού επιπέδου απεικονιστικό δομικό στοιχείο και απεικονίζει τα πάντα στην Flutter εφαρμογή. Τα επίπεδα συνεχίζουν μέχρι τον πυρήνα του κώδικα της πλατφόρμας.

  Μια γενική επισκόπηση ενός επιπέδου στο Flutter παρουσιάζεται στο παρακάτω διάγραμμα:



Εικόνα 7: *Διάγραμμα Flutter Layer*

Όπως έχουμε, λοιπόν ήδη αναφέρει το Flutter προσφέρει πολλά ευπαρουσίαστα και προσαρμοζόμενα widgets για την υλοποίηση υψηλής απόδοσης εφαρμογών, καλύπτοντας έτσι όλες τις ανάγκες και τις απαιτήσεις του προγραμματιστή. Πέραν αυτού όμως προσφέρει και πολλά ακόμα πλεονεκτήματα:

- Η Dart διαθέτει ένα εκτεταμένο οικοσύστημα πακέτων τρίτων που επιτρέπουν στον χρήστη να επεκτείνει τις δυνατότητες της εφαρμογής του.
- Οι προγραμματιστές πρέπει να γράψουν μόνο μια κοινή βάση κώδικα για όλες τις πλατφόρμες (Android, iOS, Windows κλπ)
- Το Flutter χρειάζεται λιγότερους ελέγχους (testing). Εξαιτίας της κοινής βάσης κώδικα, αρκεί να δημιουργήσουμε αυτόματα tests κοινά για όλες τις πλατφόρμες.
- Η απλότητα του Flutter το καθιστά καλή επιλογή για γρήγορη ανάπτυξη εφαρμογών. Η προσαρμοστικότητα και η επεκτασιμότητα του το κάνουν ακόμα πιο ισχυρό.
- Με το Flutter, οι προγραμματιστές έχουν τον πλήρη έλεγχο χάρη στα widgets και την δομή που εξασφαλίζει.
- Τέλος, το Flutter παρέχει πλήθος εργαλείων για τον προγραμματιστή καθώς και την εξαιρετικά χρήσιμη δυνατότητα για αυτόματη ενσωμάτωση των αλλαγών (hot reload).

Πέραν των πολλών πλεονεκτημάτων του, όμως, το Flutter έχει και ορισμένα μειονεκτήματα:

- Από την στιγμή που ο προγραμματισμός γίνεται στην γλώσσα Dart, ο προγραμματιστής καλείται να μάθει μια καινούργια γλώσσα. Παρόλα αυτά είναι μια εύκολη γλώσσα στην εκμάθησή της.
- Τα μοντέρνα frameworks προσπαθούν να ξεχωρίσουν την λογική με το UI όσο περισσότερο μπορούν. Στο Flutter, η διεπαφή χρήστη και η λογικη αναμειγνύονται. Αυτό μπορεί να ξεπεραστεί χρησιμοποιώντας έξυπνο κώδικα και υψηλού επιπέδου modules για να ξεχωρίσουμε την διεπαφή του χρήστη από την λογική.

3.2.2 Android Studio

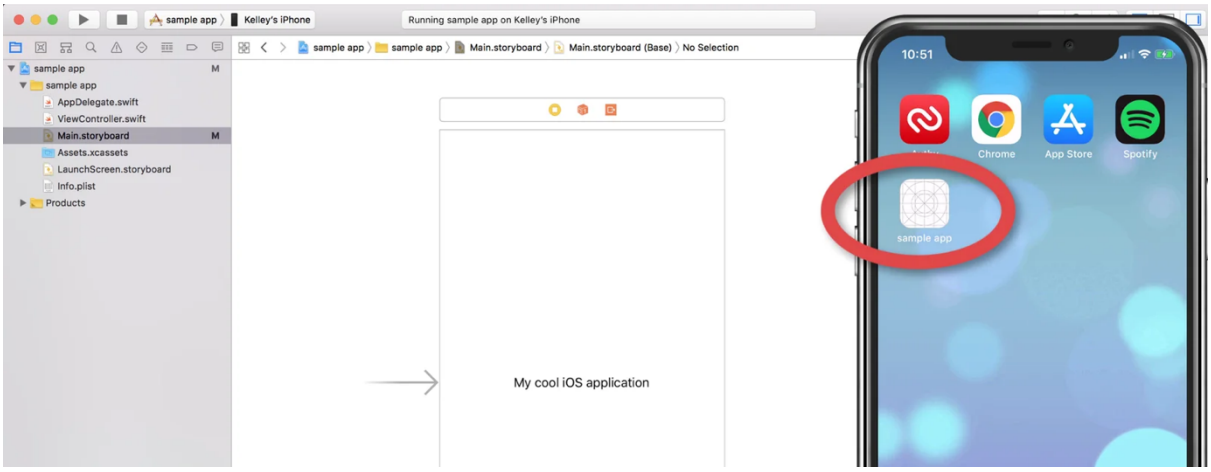
Η ανάπτυξη έγινε στο Android Studio που παρέχει στους προγραμματιστές ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) ειδικά σχεδιασμένο για εφαρμογές Android αλλά και iOS. Αρχικά το συγκεκριμένο IDE έχει την δυνατότητα να ενσωματώσει το SDK του Flutter έτσι ώστε να σε βοηθήσει με την συμπλήρωση κώδικα, διόρθωση λαθών αλλά και δυνατότητες αναδιαμόρφωσης, επιτρέποντας στους προγραμματιστές να γράφουν καθαρό και συντηρήσιμο κώδικα. Επίσης, προσφέρει μια τεράστια συλλογή από APIs, βιβλιοθήκες και εργαλεία που μπορούν να αξιοποιηθούν κατά τη δημιουργία πλούσιων σε χαρακτηριστικά και ελκυστικών εφαρμογών. Σημαντικό είναι ότι δίνεται η δυνατότητα να προσομοιώσεις τις εφαρμογές σου, σε προσομοιωτές, τόσο Android όσο και iOS λογισμικών, αλλά και σε μεγάλη γκάμα από μοντέλα κινητών που το κάθε ένα έχει και τις δικιές του διαστάσεις και χαρακτηριστικά.

Όλα αυτά είναι ιδιότητες που βοηθούν έναν προγραμματιστή, αφού εκτός από ότι το γλυτώνουν χρόνο, του δίνουν την δυνατότητα να δομήσει σωστά το έργου του, να αποφύγει περιττά λάθη και με τα απαιτούμενα σχόλια, που κάθε κώδικας πρέπει να έχει, να μπορούν και άλλοι προγραμματιστές να συμμετέχουν και να εξελίξουν την εφαρμογή.

3.2.3 Xcode

Απαραίτητο πρόγραμμα για την ανάπτυξη τέτοιων εφαρμογών, είναι το Xcode που διατίθεται μόνο σε συσκευές iOS. Το Xcode είναι το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) της Apple για macOS, που χρησιμοποιείται για την ανάπτυξη λογισμικού για macOS, iOS, iPadOS, watchOS και tvOS. Κυκλοφόρησε αρχικά στα τέλη του 2003. Οι εγγεγραμμένοι προγραμματιστές μπορούν να κάνουν λήψη εκδόσεων προεπισκόπησης και προηγούμενων εκδόσεων της σουίτας μέσω του ιστότοπου του Apple Developer. Το Xcode περιλαμβάνει εργαλεία γραμμής εντολών (Command Line Tools) μέσω της εφαρμογής Terminal στο macOS. Μπορούν επίσης να ληφθούν και να εγκατασταθούν χωρίς το GUI. Στην παρούσα διπλωματική το Xcode δεν χρησιμοποιήθηκε σαν περιβάλλον ανάπτυξης κώδικα αλλά μόνο για τους προσομοιωτές που διαθέτει για το λογισμικό iOS, αλλά και για να μπορέσουμε να δοκιμάσουμε την εφαρμογή μας σε πραγματικές συσκευές κινητών, αφού σου δίνει την δυνατότητα για ενσύρματό αλλά και ασύρματο (μέσω Wi-Fi) debugging.

Μια γενική μορφή του παρουσιάζεται στην παρακάτω Εικόνα.



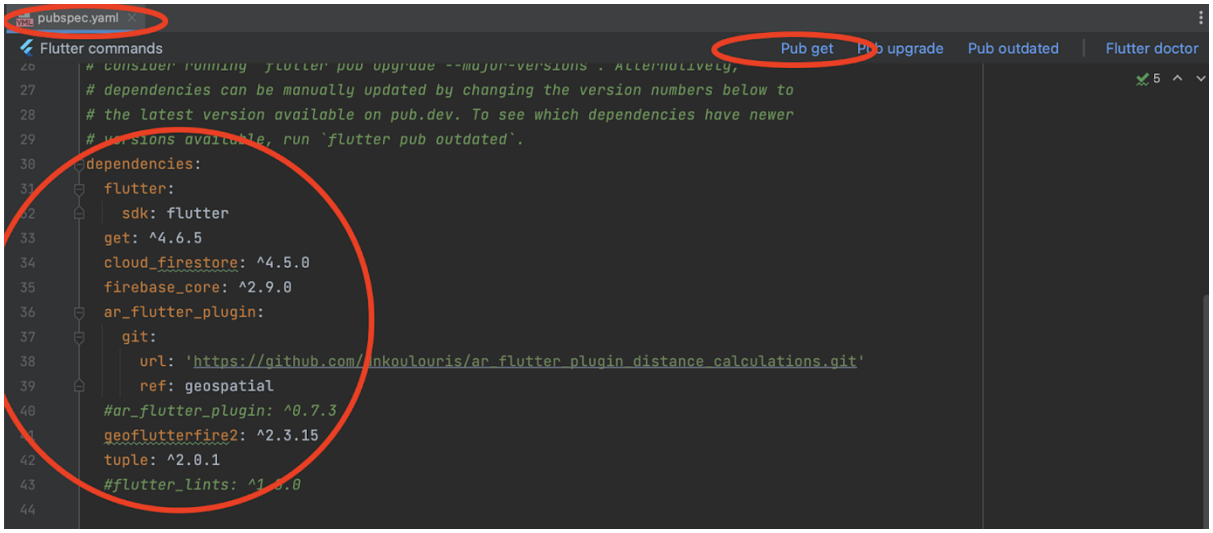
Εικόνα 8: *Xcode simulator*

3.2.4 CocoPods

Το CocoaPods είναι ένας διαχειριστής των εξαρτήσεων σε επίπεδο εφαρμογής για Objective-C και Swift που παρέχει μια τυπική μορφή για τη διαχείριση εξωτερικών βιβλιοθηκών. Η πρώτη δημόσια κυκλοφορία του έγινε την 1η Σεπτεμβρίου 2011. Το CocoaPods εστιάζει στη διανομή κώδικα τρίτων και στην αυτόματη ενσωμάτωσή τους σε έργα Xcode. Τα αναγκαία για μια εφαρμογή pods εγκαθίστανται από τη γραμμή εντολών (terminal). Με αυτό τον τρόπο, επιλύει τις εξαρτήσεις (π.χ. βιβλιοθήκες) για μια εφαρμογή βάσει προδιαγραφών που έχει, γλυτώνοντας έτσι τον προγραμματιστή από την μη αυτοματοποιημένη αντιγραφή αρχείων και κώδικα που ενδεχομένως να οδηγήσει σε λάθη και παραλείψεις.

3.2.5 Flutter Packages – Pub.dev

Ένα επίσης πολύ σημαντικό κομμάτι που αξίζει αναφορά είναι τα χιλιάδες «πακέτα» (plugins) που διαθέτει το Flutter. Μια εφαρμογή για να είναι λειτουργική εκτός από κάποια βασικά μέρη που ο προγραμματιστής μπορεί να τα δημιουργήσει μέσα από την «αρχική» έκδοση του Flutter, είναι πολύ πιθανό να χρειάζεται να προσθέσει λειτουργίες που δεν είναι άμεσα διαθέσιμες. Για να μπορούν αυτές να προσθέσουν στην εφαρμογή, θα πρέπει να εγκατασταθούν οι απαραίτητες βιβλιοθήκες που τις διαθέτουν. Όλα τα διαθέσιμα «πακέτα» που μπορούν να χρησιμοποιηθούν βρίσκονται στο Pub.dev. Το Pub.dev είναι ο διαχειριστής «πακέτων» για τη γλώσσα προγραμματισμού Dart, που περιέχει επαναχρησιμοποιήσιμες βιβλιοθήκες για Flutter, AngularDart και γενικά προγράμματα Dart. Για να ενεργοποιηθεί ένα «πακέτο» αρκεί να συμπληρωθεί κάτω από τα dependencies το όνομά του και δίπλα την έκδοση στο αρχείο “pubspec.yaml”. Στην συνέχεια πατώντας στο Terminal της εφαρμογής “pub get“ φέρνει τα «πακέτα» που έχουν οριστεί.



Εικόνα 9: *pubspec.yaml αρχείο*

3.2.6 Περιβάλλον Χρήστη

Κατά την σχεδίαση της εφαρμογής, ο προγραμματιστής καλείται να αποφασίσει ποιο θα είναι το περιβάλλον του χρήστη. Με άλλα λόγια πρέπει να αποφασιστεί σε ποια λογισμικά θα μπορεί να εγκατασταθεί αλλά και σε ποιες συσκευές θα είναι διαθέσιμη. Στην προκειμένη περίπτωση αφού υφίσταται σε πανεπιστημιακό επίπεδο με δυνατότητα χρήσης από οποιοδήποτε φοιτητή ή καθηγητή, είναι αναγκαίο η εφαρμογή να ανήκει στην κατηγορία των λογισμικών πολλαπλών πλατφορμών (Cross Platforms).

Cross Platform

Στην πληροφορική, το λογισμικό πολλαπλών πλατφορμών (που ονομάζεται επίσης λογισμικό διαγνωστικό πλατφόρμας ή λογισμικό ανεξάρτητο πλατφόρμας) είναι λογισμικό υπολογιστών που έχει σχεδιαστεί για να λειτουργεί σε διάφορες υπολογιστικές πλατφόρμες. Ορισμένα λογισμικά πολλαπλών πλατφορμών απαιτούν ξεχωριστή κατασκευή για κάθε πλατφόρμα, άλλα όμως μπορούν να εκτελεστούν απευθείας σε οποιαδήποτε πλατφόρμα χωρίς ειδική προετοιμασία, καθώς είναι γραμμένα σε μια διερμηνευμένη γλώσσα ή μεταγλωττισμένα σε φορητό bytecode για τα οποία οι διερμηνείς ή τα πακέτα χρόνου εκτέλεσης είναι κοινά ή τυποποιημένα συστατικά όλων των υποστηριζόμενων πλατφορμών.

Για παράδειγμα, μια διαπλατφορμική εφαρμογή μπορεί να τρέχει σε Linux, macOS και Microsoft Windows. Το λογισμικό πολλαπλών πλατφορμών μπορεί να τρέχει σε πολλές πλατφόρμες ή σε μόλις δύο. Ορισμένα πλαίσια για την ανάπτυξη διαπλατφορμικών εφαρμογών είναι τα Codename One, ArkUI, Kivy, Qt, Flutter, NativeScript, Xamarin, Phonegap, Ionic και React Native [7].

3.2.7 Βάση Δεδομένων

Η Firebase είναι μια πλατφόρμα για διαδικτυακές εφαρμογές, με εργαλεία και υποδομές, με τα οποία μπορεί να βοηθήσει τους προγραμματιστές να αναπτύξουν εφαρμογές υψηλής ποιότητας. Είναι μια Backend-as-a-Service (BaaS) πλατφόρμα ανάπτυξης εφαρμογών που παρέχει backend services όπως realtime database, cloud storage, authentication, crash reporting, Machine Learning, remote configuration, and hosting για τα στατικά αρχεία. Αποθηκεύει τα δεδομένα σε JavaScript Object Notation (JSON) και δεν χρησιμοποιεί ερώτημα για την εισαγωγή, ενημέρωση, διαγραφή ή προσθήκη δεδομένων μιας και μπορεί να αποτελέσει backend ενός συστήματος που χρειάζεται μια βάση δεδομένων για την αποθήκευση. Η εταιρεία ιδρύθηκε το 2011 από τον Andrew Lee και James Tamplin. Η Firebase είναι μια πραγματικού χρόνου (real-time) βάση δεδομένων, η οποία παρέχει ένα API που επιτρέπει στους προγραμματιστές να αποθηκεύουν και να συγχρονίζουν δεδομένα μεταξύ πολλαπλών clients. Με τη πάροδο του χρόνου, έχει επεκτείνει τη σειρά των προϊόντων της σε σημείο να έχει γίνει πλέον μια πλήρης σουίτα για την ανάπτυξη εφαρμογών. Η εταιρεία εξαγοράστηκε από την Google τον Οκτώβριο του 2014 και πρόσθεσε ένα σημαντικό αριθμό νέων χαρακτηριστικών το Μάιο του 2016, στο συνέδριο Google I/O [8].

Εικόνα 10: *Firebase*

Παρακάτω αναλύονται οι πιο σημαντικές υπηρεσίες που μπορεί να προσφέρει η Firebase[8]:

Firebase Authentication

Οι περισσότερες εφαρμογές πρέπει να γνωρίζουν την ταυτότητα ενός χρήστη. Η γνώση της ταυτότητας ενός χρήστη επιτρέπει σε μια εφαρμογή να αποθηκεύει με ασφάλεια τα δεδομένα του χρήστη στο cloud και να παρέχει την ίδια εξατομικευμένη εμπειρία σε όλες τις συσκευές του χρήστη.

Το Firebase Authentication παρέχει υπηρεσίες backend, εύχρηστα SDK και έτοιμες βιβλιοθήκες UI για την πιστοποίηση ταυτότητας των χρηστών στην εφαρμογή. Υποστηρίζει τον έλεγχο ταυτότητας χρησιμοποιώντας κωδικούς πρόσβασης, αριθμούς τηλεφώνου, δημοφιλείς παρόχους ομοσπονδιακής ταυτότητας όπως η Google, το Facebook και το Twitter και πολλά άλλα.

Το Firebase Authentication ενσωματώνεται στενά με άλλες υπηρεσίες της Firebase και αξιοποιεί βιομηχανικά πρότυπα όπως το OAuth 2.0 και το OpenID Connect, ώστε να μπορεί να ενσωματωθεί εύκολα με το προσαρμοσμένο backend.

Firebase Realtime Database

Η βάση δεδομένων Firebase Realtime Database είναι μια βάση δεδομένων που φιλοξενείται στο cloud. Τα δεδομένα αποθηκεύονται ως JSON και συγχρονίζονται σε πραγματικό χρόνο σε κάθε συνδεδεμένο πελάτη. Όταν κατασκευάζονται εφαρμογές πολλαπλών πλατφορμών με τα SDKs για πλατφόρμες Apple, Android και JavaScript, όλοι οι πελάτες μοιράζονται μια κοινή περίπτωση βάσης δεδομένων Realtime και λαμβάνουν αυτόματα ενημερώσεις με τα νεότερα δεδομένα.

Firebase Cloud Firestore

Η Firestore είναι μια υπηρεσία βάσης δεδομένων που παρέχεται από την Google ως μέρος της πλατφόρμας Firebase. Είναι μια NoSQL βάση δεδομένων που σχεδιάστηκε ειδικά για την ανάπτυξη εφαρμογών στον τομέα του cloud και της ιστοσελίδας. Η Firestore προσφέρει πληθώρα χαρακτηριστικών και πλεονεκτημάτων, συμπεριλαμβανομένων:

- Ευελιξία σχεδίασης: Η Firestore χρησιμοποιεί μοντέλο δεδομένων βασισμένο σε συλλογές (collections) και έγγραφα (documents), παρέχοντας έναν ευέλικτο τρόπο αποθήκευσης και ανάκτησης δεδομένων.

- Σύγχρονος συγχρονισμός: Η Firestore υποστηρίζει συγχρονισμό σε πραγματικό χρόνο, επιτρέποντας την άμεση ενημέρωση των δεδομένων σε πραγματικό χρόνο σε όλες τις συσκευές που συνδέονται.

- Σωρευτικά ερωτήματα: Μπορείτε να εκτελέσετε πολύπλοκα ερωτήματα για την ανάκτηση δεδομένων από την Firestore χωρίς να χρειάζεται να φροντίζετε για τη δομή της βάσης δεδομένων.

- Αυθεντικοποίηση και ασφάλεια: Η Firestore ενσωματώνει την αυθεντικοποίηση χρηστών και διαχείριση των δικαιωμάτων πρόσβασης, προσφέροντας έτσι ασφάλεια για τα δεδομένα.

- Ευελιξία πλατφόρμας: Η Firestore είναι διαθέσιμη για πολλές πλατφόρμες, συμπεριλαμβανομένων των ιστοσελίδων, των κινητών εφαρμογών (Android και iOS) και των εφαρμογών στον τομέα του cloud.

Firebase Cloud Storage

Το Cloud Storage for Firebase είναι μια ισχυρή, απλή και οικονομικά αποδοτική υπηρεσία αποθήκευσης αντικειμένων που δημιουργήθηκε για την κλίμακα της Google. Τα Firebase SDKs για Cloud Storage προσθέτουν ασφάλεια Google στις μεταφορτώσεις και λήψεις αρχείων για τις εφαρμογές Firebase, ανεξάρτητα από την ποιότητα του δικτύου.

Μπορεί να χρησιμοποιηθούν τα SDKs ενός πελάτη για την αποθήκευση εικόνων, ήχου, βίντεο ή άλλου περιεχομένου που δημιουργείται από χρήστες. Στο διακομιστή, μπορεί να χρησιμοποιηθεί το Firebase Admin SDK για να να δημιουργηθούν διευθύνσεις URL λήψης και να χρησιμοποιηθούν τα Google Cloud Storage APIs για να αποκτηθεί πρόσβαση στα αρχεία.

Firebase Cloud Functions

Το Cloud Functions for Firebase είναι ένα πλαίσιο χωρίς διακομιστή που επιτρέπει να εκτελείται αυτόματα κώδικας backend σε απόκριση, σε συμβάντα που προκαλούνται από τις λειτουργίες Firebase και τα αιτήματα HTTPS. Ο κώδικάς JavaScript ή TypeScript αποθηκεύεται στο cloud της Google και εκτελείται σε ένα διαχειριζόμενο περιβάλλον. Δεν χρειάζεται να διαχειρίζονται και να κλιμακώνονται οι διακομιστές.

Firebase Cloud Messaging (FCM)

Χρησιμοποιώντας το Firebase Cloud Messaging, μπορεί να ειδοποιηθεί μια εφαρμογή-πελάτη ότι νέα email ή άλλα δεδομένα είναι διαθέσιμα για συγχρονισμό. Μπορεί να στέλνει μηνύματα ειδοποίησης για να προωθεί την επαναπροσέγγιση και τη διατήρηση των χρηστών. Για περιπτώσεις χρήσης, όπως η ανταλλαγή άμεσων μηνυμάτων, ένα μήνυμα μπορεί να μεταφέρει ωφέλιμο φορτίο έως και 4000 bytes σε μια εφαρμογή-πελάτη.

Firebase Machine Learning

Το Firebase Machine Learning είναι ένα SDK για κινητά που φέρνει την τεχνογνωσία Μηχανικής Μάθησης της Google σε εφαρμογές Android και Apple σε ένα ισχυρό αλλά εύχρηστο πακέτο. Είτε νέοι, είτε έμπειροι στη Μηχανική Μάθηση, μπορούν να υλοποιούν τη λειτουργικότητα που χρειάζονται με λίγες μόνο γραμμές κώδικα. Δεν χρειάζεται να έχουν βαθιά γνώση των νευρωνικών δικτύων ή της βελτιστοποίησης μοντέλων για να ξεκινήσουν. Από την άλλη πλευρά, σε έναν έμπειρος προγραμματιστή ML, το Firebase ML παρέχει βολικά API που τον βοηθούν να χρησιμοποιεί τα προσαρμοσμένα μοντέλα TensorFlow Lite στις εφαρμογές του για κινητά.

3.2.8 API

Το "API" αναφέρεται στα αρχικά της φράσης "Application Programming Interface" στην αγγλική γλώσσα. Ένα API είναι ένα σύνολο προκαθορισμένων κανόνων, πρωτοκόλλων και εργαλείων που επιτρέπουν σε διάφορες εφαρμογές ή συστήματα λογισμικού να επικοινωνούν μεταξύ τους. Το API ορίζει τον τρόπο με τον οποίο μια εφαρμογή μπορεί να αιτηθεί και να ανταλλάξει δεδομένα με μια άλλη εφαρμογή, υπηρεσία ή συσκευή.

Ένα API μπορεί να χρησιμοποιηθεί για πολλούς σκοπούς, συμπεριλαμβανομένων των παρακάτω:

Διεπαφή Χρήστη - Εφαρμογή: Επιτρέπει σε μια εφαρμογή να αλληλεπιδρά με τον χρήστη, όπως η διεπαφή χρήστη (UI) μιας εφαρμογής.

Πρόσβαση σε Υπηρεσίες: Επιτρέπει σε μια εφαρμογή να αποκτήσει πρόσβαση σε υπηρεσίες ή δεδομένα που παρέχονται από άλλες εφαρμογές ή απομακρυσμένους διακομιστές.

Διασύνδεση Εφαρμογών: Επιτρέπει σε διάφορες εφαρμογές να συνδυάσουν τις δυνατότητές τους και να λειτουργήσουν μαζί.

Διαμοιρασμός Πόρων: Επιτρέπει το διαμοιρασμό πόρων όπως φωτογραφίες, βίντεο, δεδομένα βάσεων δεδομένων και άλλα μεταξύ διαφόρων εφαρμογών.

Τα API είναι ουσιώδη στοιχεία του σύγχρονου λογισμικού, καθώς επιτρέπουν την ανάπτυξη εφαρμογών που μπορούν να συνεργάζονται και να αλληλεπιδρούν με διάφορα συστήματα και υπηρεσίες [9].

3.3 Υλοποίηση

Στην συγκεκριμένη ενότητα, θα αναλυθεί πλήρως η εφαρμογή που κατασκευάστηκε, οι αρχιτεκτονικές της, η βάση δεδομένων της και τα «πακέτα» που χρησιμοποιήθηκαν.

3.3.1 Αρχιτεκτονικές Υλοποίησης

Serverless Computing Architecture

Το Serverless Computing είναι μια μέθοδος παροχής backend υπηρεσιών σε μόνιμη βάση. Ένας πάροχος serverless υπηρεσιών επιτρέπει στους χρήστες του να γράψουν και να εκτελέσουν κώδικα χωρίς να πρέπει να ανησυχούν για την υποκείμενη υποδομή. Μια εταιρεία που χρησιμοποιεί backend υπηρεσίες από έναν πάροχο serverless χρεώνεται βάση της χρήσης και δεν χρειάζεται να κρατήσει και να πληρώσει για ένα συγκεκριμένο εύρος ή αριθμό από servers, καθώς οι πόροι κλιμακώνονται προς τα πάνω και προς τα κάτω αυτόματα για να ανταποκριθούν στην ζήτηση. Να σημειωθεί, ότι παρά το όνομα serverless, εξακολουθούν να

χρησιμοποιούνται φυσικοί servers, αλλά οι προγραμματιστές δεν χρειάζεται να ανησυχούν για αυτούς. Το serverless computing είναι μια πλατφόρμα η οποία αποκρύπτει τη χρήση του server από τον προγραμματιστή [10].

Μια serverless πλατφόρμα μπορεί να προσφέρει είτε το μοντέλο Function as a Service (FaaS), είτε το μοντέλο Backend as a Service (BaaS), είτε και τα δύο [10].

- Function as a Service (FaaS): Το FaaS επιτρέπει στους προγραμματιστές να αναπτύξουν διακριτά block κώδικα για εκτέλεση προς απόκριση συγκεκριμένων συμβάντων χωρίς να ανησυχούν για τη διαχείριση πόρων στο backend. Οι συναρτήσεις αυτές τρέχουν σε έτοιμα περιβάλλοντα εκτέλεσης γλωσσών προγραμματισμού όπως JavaScript, Python κλπ.

- Backend as a Service (BaaS): Την ίδια στιγμή το BaaS παρέχει υπηρεσίες API που προσφέρουν βασική λειτουργικότητα της εφαρμογής. Δεδομένου ότι τα APIs παρέχονται ως υπηρεσία κλιμακούμενη αυτόματα στο παρασκήνιο, χωρίς καμία παρέμβαση από τον προγραμματιστή, οι επιχειρήσεις μπορούν να αναπτύσσουν εφαρμογές με πολύ μεγαλύτερη ταχύτητα και ευκινησία.

MVC Architecture

MVC σημαίνει Μοντέλο-Προβολή-Ελεγκτής (Model-View-Controller). Είναι ένα αρχιτεκτονικό πρότυπο που δημιουργήθηκε το 1979. Το MVC, δημιουργήθηκε για διεπαφές χρήστη σε εφαρμογές γραφείου και τώρα γίνεται δημοφιλές σε εφαρμογές ιστού (web applications). Το MVC χωρίζει την εφαρμογή σε τρία ξεχωριστά μέρη [11]:

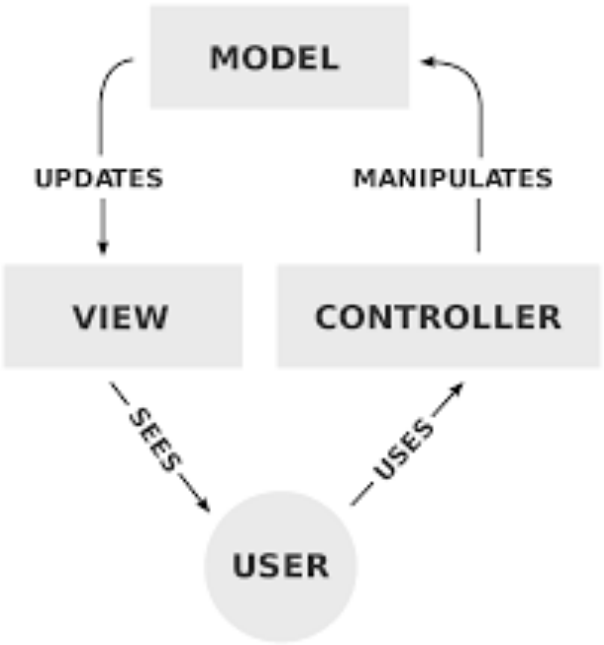
- Μοντέλο, το οποίο είναι το επίπεδο πρόσβασης στα δεδομένα.

- Προβολή, που είναι το επίπεδο παρουσίασης.

- Ελεγκτής, το κύριο σύστημα, που λαμβάνει τα εισερχόμενα αιτήματα και ανταποκρίνεται σε αυτά στέλνοντας πίσω διαφορετικές προβολές χρησιμοποιώντας τα δεδομένα που δίνονται από το μοντέλο.

Η αρχιτεκτονική MVC έχει μια σειρά από πλεονεκτήματα, το σημαντικότερο από τα οποία είναι η δυνατότητα δημιουργίας διαφορετικών προβολών για τις ταυτόχρονα πολλές συσκευές (για παράδειγμα, για επιτραπέζιες και κινητές συσκευές) χωρίς να χρειάζεται να τροποποιηθεί οποιαδήποτε άλλο στοιχείο της εφαρμογής. Έχει αποδειχθεί ότι, η δημιουργία mobile με βάση

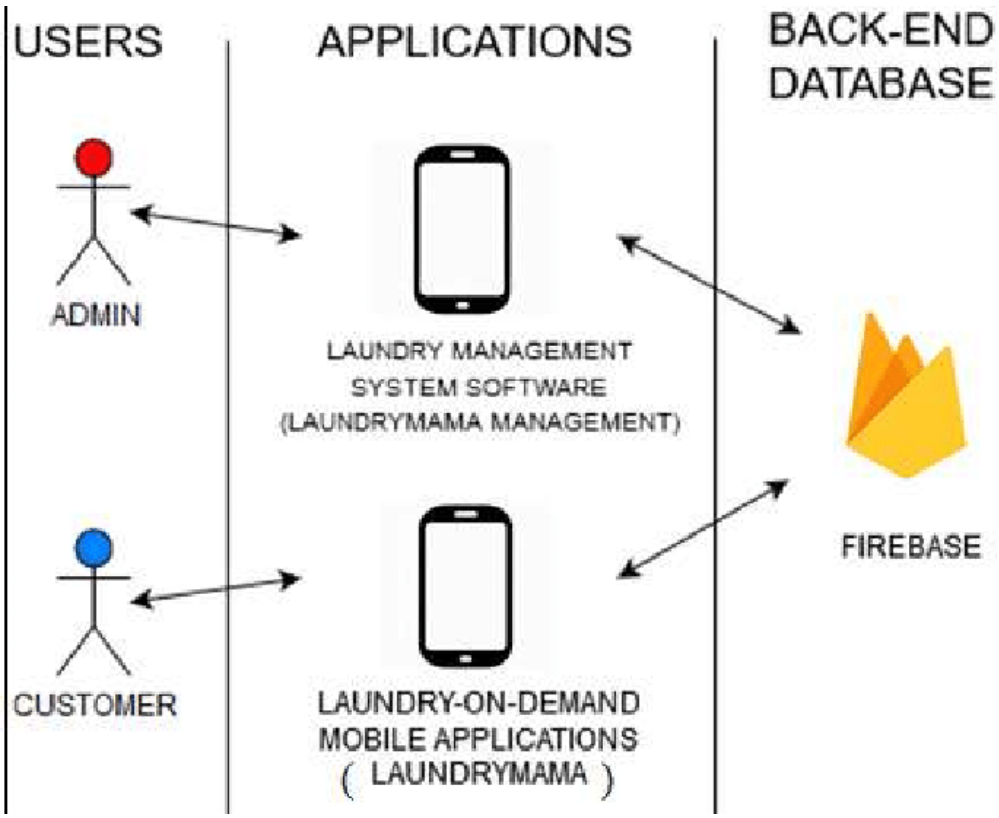
την υπάρχουσα desktop εφαρμογή μπορεί να είναι πολύ πιο γρήγορη από τη δημιουργία της έκδοσης για κινητά από την αρχή.



Εικόνα 11: *MVC Architecture*

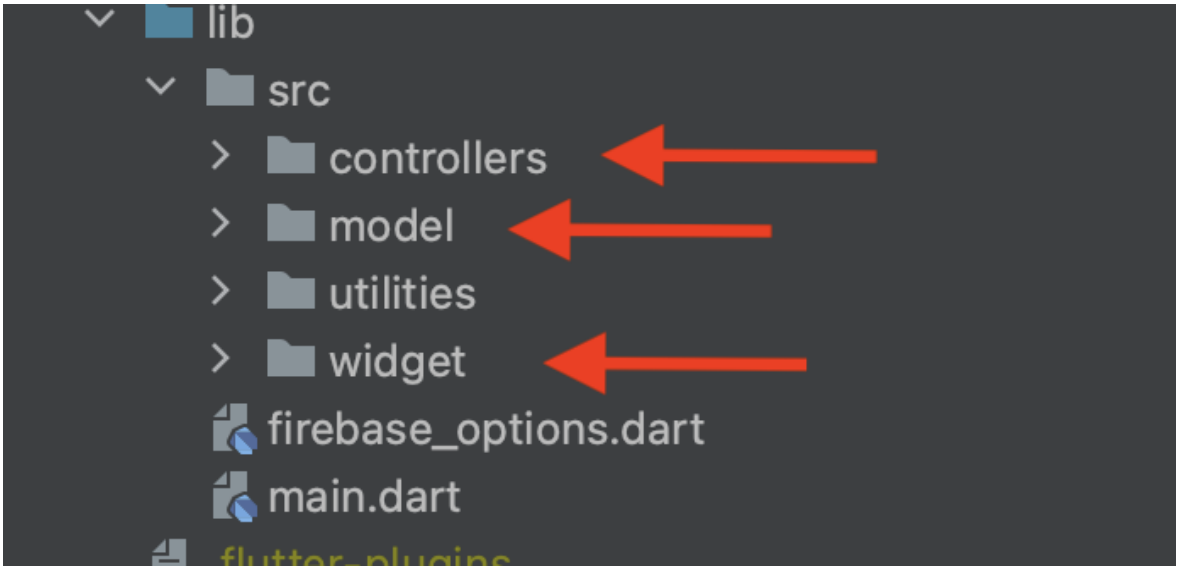
Αρχιτεκτονική της παρούσας εφαρμογής

Στη παρούσα εφαρμογή χρησιμοποιήθηκαν και οι δύο αρχιτεκτονικές που αναφέρθηκαν, αφού η υιοθέτηση της μιας δεν επηρεάζει την άλλη. Αρχικά όσον αφορά την Serverless Computing Architecture, η εφαρμογή είχε σαν BaaS την Firebase που ήταν απαραίτητο κομμάτι της, για να μπορέσει να είναι σε πρώτη φάση δυναμική, δηλαδή όλο και περισσότεροι χρήστες να μπορούν να συνδεθούν, αλλά και λειτουργική, αφού για την υλοποίηση ήταν αναγκαίο η χρήση μιας Cloud βάσης δεδομένων με εξ αποστάσεως σύνδεση συσκευών, όπως θα επισημανθεί και παρακάτω.



Εικόνα 12: *Firebase-Mobile Architecture*

Επίσης, για την καλύτερη οργάνωση του έργου χρησιμοποιήθηκε και η MVC αρχιτεκτονική, η οποία μπορεί να προσαρμοστεί ειδικά σε εφαρμογές του Flutter, μιας και υπάρχουν συγκεκριμένα «πακέτα», που επιτρέπουν τον διαμοιρασμό της εφαρμογή στα τρία μέρη που ορίζει η συγκεκριμένη αρχιτεκτονική (όπου View = Widget). Αυτό έχει ως βασικό πλεονέκτημα την σωστή οργάνωση, την επαναχρησιμοποίηση κώδικα, την διόρθωση λαθών αλλά και την εύκολη ανάγνωση κώδικα από νέους προγραμματιστές στο έργο που θέλουν να συμμετάσχουν και να το εξελίξουν.



Εικόνα 13: *MVC εφαρμογής*

3.3.2 Πακέτα (packages) , Βιβλιοθήκες & API

Packages

Παρακάτω αναλύονται τα «πακέτα» (plugins) που χρησιμοποιήθηκαν και είχαν πρωταγωνιστικό ρόλο στην κατασκευή της εφαρμογής μας, ***Εικόνα 9***.

Get or GetX package

Το GetX, ***Εικόνα 14***, είναι μια εξαιρετικά ελαφριά και ισχυρή λύση για το Flutter. Συνδυάζει τη διαχείριση καταστάσεων υψηλής απόδοσης, την έξυπνη εισαγωγή εξαρτήσεων και τη διαχείριση διαδρομών γρήγορα και πρακτικά. Το GetX έχει 3 βασικές αρχές. Αυτό σημαίνει ότι αυτές έχουν προτεραιότητα για όλους τους πόρους της βιβλιοθήκης: ΠΑΡΑΓΩΓΙΚΟΤΗΤΑ, ΑΠΌΔΟΣΗ ΚΑΙ ΟΡΓΆΝΩΣΗ [12].

- ΑΠΟΔΟΣΗ: Το GetX επικεντρώνεται στην απόδοση και την ελάχιστη κατανάλωση πόρων. Το GetX δεν χρησιμοποιεί Streams ή ChangeNotifier.

- ΠΑΡΑΓΩΓΙΚΟΤΗΤΑ: Το GetX χρησιμοποιεί μια εύκολη και ευχάριστη σύνταξη. Ανεξάρτητα από το τι θέλει να κάνει ο προγραμματιστής, υπάρχει πάντα ένας ευκολότερος τρόπος με το GetX. Εξοικονομεί ώρες ανάπτυξης και παρέχει τη μέγιστη απόδοση που μπορεί να προσφέρει η εφαρμογή.

- ΟΡΓΑΝΩΣΗ: Το GetX επιτρέπει την πλήρη κατάργηση της λογικής παρουσίασης, της επιχειρησιακής λογικής, της έγχυσης εξαρτήσεων και της πλοήγησης. Δεν χρειάζεται πλαίσιο για την πλοήγηση μεταξύ των διαδρομών, οπότε δεν εξαρτάται από το δέντρο των widget για αυτό.

Χρήση της βιβλιοθήκης στην εφαρμογή

Η συγκεκριμένη βιβλιοθήκη χρησιμοποιήθηκε αρχικά για μετατροπή όλων των οθονών σε StatelessWidget, αυτό έγινε με τις δυνατότητες που έχει στην διαχείριση της κατάστασης (State management) εφαρμογών. Δηλαδή το «πακέτο» προσφέρει στο χρήστη τη δυνατότητα να μετατρέψει όλες τις μεταβλητές σε δυναμικές άλλα και να μπορεί η οθόνη να δείχνει ζωντανά τις αλλαγές ενώ χρησιμοποιεί την κλάση StatelessWidget, στην οποία η κατάσταση δεν αλλάζει, και όχι την StatefulWidget που σύμφωνα με αυτά που αναφέρθηκαν για το Flutter είναι η προτεινόμενη. Αυτό είχε σαν αποτέλεσμα να γίνει η εφαρμογή πιο αποδοτική, αφού τα StatelessWidget είναι πιο «ελαφριά» στην εκτέλεση και όταν μιλάμε για μεγάλες και σύνθετες εφαρμογές παίζει πολύ σημαντικό ρόλο.

Επίσης, μια άλλη χρήση του ήταν για διαχείριση διαδρομών (Route management). Το GetX προσφέρει μια πιο γρήγορη εναλλαγή μεταξύ οθονών και κλάσεων της εφαρμογής. Αυτό είναι ένα πολύ σημαντικό κομμάτι, μιας και ο χρήστης δεν πρέπει να βλέπει καθυστερήσεις κατά την λειτουργία της.

Τέλος, χρησιμοποιήθηκε και για την διαχείριση εξαρτήσεων (Dependency management). Το «πακέτο» δίνει την δυνατότητα να ορίζει ο προγραμματιστής Controllers (GetXController λέγεται η κλάση), πού θα επιτελούν μια συγκεκριμένη λειτουργία. Με την διαχείριση εξαρτήσεων που προσφέρει, μπορεί να μεταφερθεί το περιεχόμενο ενός GetXController, οι μεταβλητές του, να εισαχθούν ορίσματα και στην ουσία να «τρέξει» μια λειτουργία ενός GetXController μέσα σε μία άλλη κλάση. Όλα αυτά έδωσαν την δυνατότητα να χωριστεί ο κώδικά σε αρχεία και να τον οργανωθεί όπως ορίζει η αρχιτεκτονική MVC, ***Εικόνα 13***.

Εικόνα 14: *GetX logo*

Ar_flutter_plugin

Το ar_flutter_plugin είναι ένα plugin Flutter για AR που υποστηρίζει ARCore στο Android και ARKit στις συσκευές iOS. Με αυτό μπορούν να αποκτηθούν και τα δύο ταυτόχρονα. Προφανώς είναι ένα πλεονέκτημα επειδή δεν χρειάζεται να επιλεγεί σε ποιο λογισμικό θα γίνει η ανάπτυξη [13].

- ARCore: Το ARCore είναι η πλατφόρμα της Google που επιτρέπει στο τηλέφωνό να αντιλαμβάνεται το περιβάλλον του, να κατανοεί τον κόσμο και να αλληλεπιδρά με πληροφορίες. Ορισμένα από τα παρεχόμενα API είναι προσβάσιμα σε όλες τις συσκευές Android και iOS, επιτρέποντας μια κοινή εμπειρία AR.
  Η ουσία είναι ότι τα περισσότερα iPhone που τρέχουν iOS 11.0 ή νεότερη έκδοση, και τα περισσότερα τηλέφωνα Android που τρέχουν Android 7.0 ή νεότερη έκδοση, υποστηρίζουν το ARCore. Η τεκμηρίωση του ARCore της Google το θέτει ως εξής: "Βασικά, το ARCore κάνει δύο πράγματα: παρακολουθεί τη θέση της κινητής συσκευής καθώς κινείται και δημιουργεί την κατανόηση του πραγματικού κόσμου".

- ARKit: Το ARKit είναι το σύνολο εργαλείων της Apple που επιτρέπει να δημιουργούνται εφαρμογές Επαυξημένης Πραγματικότητας για το iOS. Όποιος χρησιμοποιεί Apple A9 ή νεότερη έκδοση, iPhone 6s/7/SE/8/X, iPad 2017/Pro σε iOS 11.0 ή νεότερη έκδοση, μπορεί να χρησιμοποιήσει το ARKit. Για ορισμένες λειτουργίες απαιτείται iOS 12 ή νεότερο.

Στην ουσία το **ar_flutter_plugin** είναι το «πακέτο» που με τις κλάσεις που έχει, ενεργοποιήθηκε η Επαυξημένη Πραγματικότητα στην εφαρμογή αλλά και έγινε λειτουργική σε δύο διαφορετικά λογισμικά (Android και iOS) ώστε να ανήκει στις Cross Platforms εφαρμογές.

Παρακάτω αναλύονται οι κυριότερες κλάσεις που χρησιμοποιήθηκαν:

1. ARView (Προβολή AR): Αυτή η κλάση δημιουργεί μια πλατφορμοεξαρτημένη προβολή κάμερας χρησιμοποιώντας το **PlatformARView**. Αποτελεί την οπτική αναπαράσταση του περιβάλλοντος Επαυξημένης Πραγματικότητας στην εφαρμογή Flutter.
2. ARSessionManager (Διαχειριστής AR Session): Αυτή η κλάση διαχειρίζεται τη διαμόρφωση, τις παραμέτρους και τα συμβάντα της AR εντός του **ARView.** Ελέγχει την εμπειρία AR και χειρίζεται διάφορα γεγονότα σχετικά με το AR.

3. ARObjectManager (Διαχειριστής AR Αντικειμένων): Η **ARObjectManager** είναι υπεύθυνη για τη διαχείριση όλων των ενεργειών που σχετίζονται με τους κόμβους (nodes) εντός ενός **ARView**. Στο πλαίσιο Επαυξημένης Πραγματικότητας, οι κόμβοι είναι συνήθως 3D αντικείμενα ή οντότητες που μπορούν, από τον χρήστη, να προστεθούν, να διαμορφωθούν και να αλληλεπιδράσουν στο περιβάλλον AR.

4. ARAnchorManager (Διαχειριστής AR Anchors): Αυτή η κλάση διαχειρίζεται λειτουργίες σχετικές με τα Anchors, συμπεριλαμβανομένων των χειριστών λήψης και αποστολής. Τα Anchors είναι σημεία ή αντικείμενα στο περιβάλλον AR που είναι ακίνητα, συχνά χρησιμοποιούνται για την προσάρτηση εικονικού περιεχομένου στον πραγματικό κόσμο.

5. ARLocationManager (Διαχειριστής Τοποθεσίας AR): Η **ARLocationManager** παρέχει τη δυνατότητα να λαμβάνει και να ενημερώνει, τον χρήστη, για την τρέχουσα τοποθεσία της συσκευής στο περιβάλλον AR. Αυτό είναι σημαντικό για εφαρμογές AR που βασίζονται σε πληροφορίες βασισμένες στην τοποθεσία.

6. ARNode (Κόμβος AR): Η **ARNode** είναι μια κλάση που αντιπροσωπεύει αντικείμενα κόμβους (Nodes) εντός του περιβάλλοντος AR. Οι κόμβοι μπορούν να είναι οποιαδήποτε 3D μοντέλα, αντικείμενα ή οντότητες που θέλει ο χρήστης να προσθέσει ή να διαχειριστεί στο σκηνικό της Επαυξημένης Πραγματικότητας.

Firebase_core

Το Firebase_core plugin, είναι ένα «πακέτο» Flutter για τη χρήση του Firebase Core API, το οποίο επιτρέπει τη σύνδεση σε πολλαπλές εφαρμογές Firebase. Το συγκεκριμένο μαζί με το Cloud_firestore έδωσαν την δυνατότητα να συνδεθεί η εφαρμογή στην κεντρική και διαδικτυακή βάση δεδομένων.

Cloud_firestore

Το **cloud_firestore** είναι ένα plugin για το framework Flutter που επιτρέπει την ενσωμάτωση της βάσης δεδομένων Firestore της Google στις εφαρμογές. Το Firestore είναι μια νέα γενιά βάση δεδομένων που παρέχεται από την Google και προσφέρει σε πραγματικό χρόνο, συγχρονισμό, αποθήκευση δεδομένων σε λειτουργία cloud, και εξαιρετικά ευέλικτες δυνατότητες διαχείρισης δεδομένων [13].

Ορισμένα βασικά σημεία σχετικά με το **cloud_firestore** plugin για το Flutter [14]:

- **Σύνδεση με το Firestore:** Με το **cloud_firestore**, δίνεται η δυνατότητα για σύνδεση στο Firestore χρησιμοποιώντας τα διαπιστευτήρια της εφαρμογής.

- **Λειτουργίες Firestore:** Το plugin παρέχει μια πληθώρα λειτουργιών για την ανάγνωση, εγγραφή και ενημέρωση δεδομένων στο Firestore. Μπορεί ένας χρήστης να δημιουργήσει, να διαβάσει και να ενημερώσει εγγραφές, να διαχειριστεί συλλογές και να παρακολουθεί αλλαγές σε πραγματικό χρόνο.

- **Ασφάλεια:** Το Firestore προσφέρει ενσωματωμένη ασφάλεια και ελέγχους πρόσβασης. Το **cloud_firestore** plugin επιτρέπει τη ρύθμιση κανόνων πρόσβασης στη βάση δεδομένων για την προστασία τους.

- **Συμβατότητα με το Flutter:** Το plugin είναι σχεδιασμένο για να λειτουργεί άψογα με το Flutter και να επωφελείται από τη γρήγορη απόδοση και την εκτέλεση που προσφέρει το Flutter framework.

Το **cloud_firestore** plugin είναι ένα ισχυρό εργαλείο για τη διαχείριση δεδομένων σε εφαρμογές Flutter που απαιτούν αποθήκευση και ανάκτηση δεδομένων από μια βάση δεδομένων στο cloud. Μπορεί να χρησιμοποιηθεί για την ανάπτυξη δυναμικών εφαρμογών που απαιτούν αληθινή χρονική ενημέρωση των δεδομένων και πολλές άλλες επιχειρησιακές λειτουργίες.

ARCore API

Όσον αφορά τους εσωτερικούς χώρους, η υλοποίηση βασίστηκε στα **Cloud Anchors** που προσφέρονται μέσω του ARCore API.

Αρχικά, το ARCore είναι η πλατφόρμα της Google για τη δημιουργία εμπειριών Επαυξημένης Πραγματικότητας. Χρησιμοποιώντας διάφορα API, το ARCore επιτρέπει στο τηλέφωνό να αντιλαμβάνεται το περιβάλλον του, να κατανοεί τον κόσμο και να αλληλεπιδρά με πληροφορίες. Ορισμένα από τα API είναι διαθέσιμα σε όλο το Android και το iOS για να επιτρέψουν εμπειρίες AR.

Το ARCore χρησιμοποιεί τρεις βασικές δυνατότητες για την ενσωμάτωση του εικονικού περιεχομένου με τον πραγματικό κόσμο, όπως τον βλέπουμε μέσω της κάμερας του τηλεφώνου:

- **Η παρακολούθηση κίνησης:** επιτρέπει στο τηλέφωνο να κατανοεί και να παρακολουθεί τη θέση του σε σχέση με τον κόσμο.

- **Η κατανόηση του περιβάλλοντος:** επιτρέπει στο τηλέφωνο να ανιχνεύει το μέγεθος και τη θέση όλων των τύπων επιφανειών: οριζόντιες, κάθετες και υπό γωνία επιφάνειες, όπως το έδαφος, ένα τραπεζάκι του καφέ ή τοίχοι.

- **Η εκτίμηση φωτισμού:** επιτρέπει στο τηλέφωνο να εκτιμά τις τρέχουσες συνθήκες φωτισμού του περιβάλλοντος.

Συγκεκριμένα, το ARCore κάνει δύο πράγματα: παρακολουθεί τη θέση της κινητής συσκευής καθώς κινείται και δημιουργεί τη δική του κατανόηση του πραγματικού κόσμου.

Το ARCore παρέχει SDKs για πολλά από τα πιο δημοφιλή περιβάλλοντα ανάπτυξης. Αυτά τα SDK παρέχουν API για όλα τα βασικά χαρακτηριστικά AR, όπως η παρακολούθηση κίνησης, η κατανόηση του περιβάλλοντος και η εκτίμηση του φωτός. Με αυτές τις δυνατότητες μπορούν να δημιουργηθούν εντελώς νέες εμπειρίες AR ή να βελτιωθούν υπάρχουσες εφαρμογές με χαρακτηριστικά AR [15].

Cloud Anchors

Το Cloud Anchor είναι ένας ειδικός τύπος Anchor (φέρουν τα χαρακτηριστικά που έχουν αναφερθεί) που μπορεί να χρησιμοποιηθεί για τη διατήρηση εμπειριών AR στον πραγματικό κόσμο. Με το ARCore API μπορούν να δημιουργηθούν διαδραστικά στρώματα ψηφιακών πληροφοριών και να τοποθετηθούν σε πραγματικές τοποθεσίες, σχεδιάζοντας εμπειρίες που μπορούν να μοιραστούν με την πάροδο του χρόνου από πολλούς ανθρώπους σε πολλές διαφορετικές συσκευές. Τα Cloud Anchors συνδέουν τοποθεσίες του πραγματικού κόσμου με ψηφιακό περιεχόμενο στο οποίο μπορεί να έχει πρόσβαση ο καθένας από συμβατές κινητές συσκευές. Τόσο οι χρήστες Android όσο και οι χρήστες iOS μπορούν να συμμετέχουν στην ίδια εμπειρία και να επιστρέφουν σε αυτήν ξανά και ξανά, ακόμη και εβδομάδες ή μήνες αργότερα. Τα Cloud Anchors είναι Anchors που φιλοξενούνται στο cloud μέσω ARCore API σε συγκεκριμένα endpoints [15].

Τρόπος λειτουργίας Cloud Anchors

1. Ο χρήστης δημιουργεί ένα τοπικό Anchor στο περιβάλλον του και χαρακτηρίζεται από ένα μοναδικό χαρτί χαρακτηριστικών χώρου (Feature Map).

2. Το Anchor φιλοξενείται (hosted): το ARCore μεταφορτώνει (upload) τα δεδομένα αυτού του τοπικού Anchor στο endpoint του ARCore API στο cloud και το endpoint επιστρέφει ένα μοναδικό ID (cloudanchorid) για το εν λόγω Anchor.

3. Η εφαρμογή διανέμει αυτό το μοναδικό αναγνωριστικό σε άλλους χρήστες.

4. Το Anchor επιλύεται (resoled): οι χρήστες των οποίων οι συσκευές διαθέτουν το μοναδικό αναγνωριστικό μπορούν να δημιουργήσουν εκ νέου το ίδιο Anchor χρησιμοποιώντας το ARCore API, «χτυπώντας» το endpoint με το συγκεκριμένο ID σαν παράμετρο.

Η ειδοποιός διαφορά των Cloud Anchors με των ARAnchor, είναι ότι με τα Cloud ξεφεύγουμε από τον τοπικό χώρο της κάθε συσκευής και μπορούμε να δούμε Anchors που έχουν τοποθετηθεί από άλλους χρήστες, οποιαδήποτε χρονική στιγμή και από κάθε συσκευή αλλά και αντίστοιχα να βάλουμε τα δικά μας στα σημεία που θέλουμε.

Τέλος, για να τοποθετηθεί ένα Cloud Anchor σωστά αλλά και για το γρηγορότερο resolving του, πρέπει να γίνει καλή καταγραφή του χώρου, έτσι ώστε να αναγνωριστεί και να αντιστοιχηθεί το Feature Map που το χαρακτηρίζει.

Geospatial Anchors

Η βασική διαφορά με τα Cloud Anchors είναι ότι τα Geospatial τοποθετούνται κάθε φορά που ένας χρήστης ξεκινάει την διαδρομή, ενώ τα Cloud Anchors έχουν τοποθετηθεί από άλλους και ο χρήστης τα «κατεβάζει» μέσω το ARCore API στην προκαθορισμένη τοποθεσία τους.

glTF or GLB file

Τα αρχεία glTF (GL Transmission Format) και τα αρχεία GLB είναι δύο δημοφιλείς μορφές αρχείων 3D που χρησιμοποιούνται στην ανάπτυξη εφαρμογών Επαυξημένης Πραγματικότητας και 3D γραφικών για εφαρμογές Flutter και άλλες πλατφόρμες. Ορισμένες βιβλιοθήκες AR και τρισδιάστατων γραφικών υποστηρίζουν αυτές τις μορφές αρχείων.

glTF (GL Transmission Format):

- Το glTF είναι ένα ανοικτό πρότυπο μορφής αρχείου που χρησιμοποιείται για την αποθήκευση 3D μοντέλων και σκηνών.

- Υποστηρίζει την αποθήκευση γεωμετρίας, υλικών, κινήσεων, και άλλων πληροφοριών που απαιτούνται για την αποτύπωση ενός 3D μοντέλου.

- Είναι αποδοτικό και κατάλληλο για αποστολή στις συσκευές AR, επειδή μπορεί να φορτωθεί και να αποτυπωθεί γρήγορα.

- Υποστηρίζεται από πολλές εφαρμογές και πλατφόρμες.

GLB (Binary glTF):

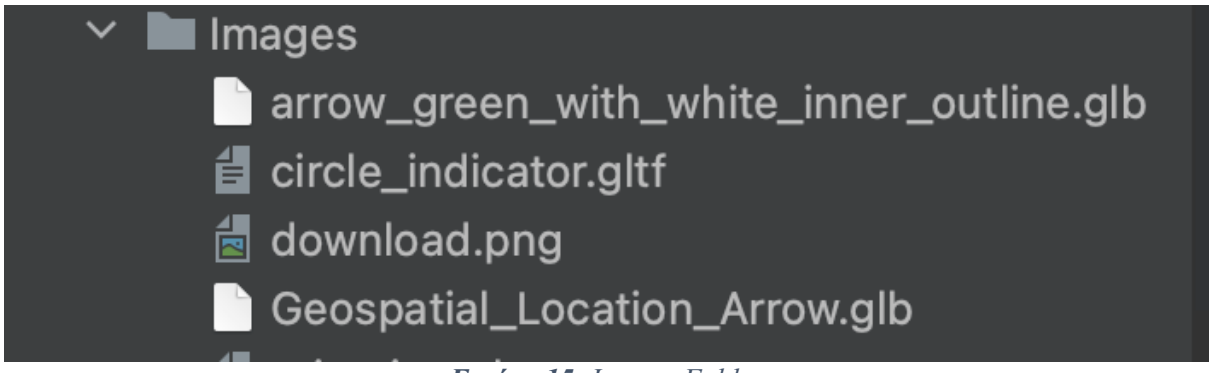
- Το GLB είναι μια δυαδική μορφή αρχείου που βασίζεται στο glTF. Συμπεριλαμβάνει το 3D μοντέλο, τα υλικά και άλλα δεδομένα σε ένα μόνο αρχείο.

- Είναι πολύ αποδοτικό για τη φόρτωση και αποθήκευση, καθώς η δυαδική μορφή του το καθιστά ευκολότερο στη διαχείριση.

- Συχνά χρησιμοποιείται σε εφαρμογές AR και παιχνίδια για τη μεταφορά 3D περιεχομένου.

Σε εφαρμογές Flutter για AR, μπορούν να χρησιμοποιηθούν βιβλιοθήκες και «πακέτα» που υποστηρίζουν το glTF και το GLB για την αποτύπωση και την εμφάνιση 3D περιεχομένου στον πραγματικό κόσμο χρησιμοποιώντας την τεχνολογία AR. Αυτά τα αρχεία επιτρέπουν να προστεθούν εικονικά αντικείμενα, όπως 3D μοντέλα, στον πραγματικό κόσμο καθώς χρησιμοποιείται μια εφαρμογή AR με το Flutter.

Τα συγκεκριμένα αρχεία υπάρχουν στο φάκελο Images, ***Εικόνα 15***.



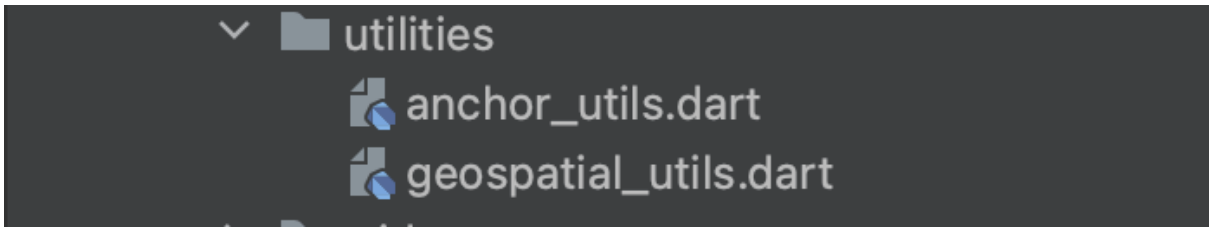
Εικόνα 15: *Images Folder*

3.3.3 Υλοποίηση Incremental Learning

Για τις ανάγκες της εφαρμογής δημιουργήθηκαν δύο μέθοδοι, ***Εικόνα 16***.

Αρχικά σε μια διαδρομή που θεωρείται σωστά χαραγμένη, έπρεπε πάρα πολλά Cloud Anchors να τοποθετηθούν για να μην υπάρχουν κενά και να είναι σαν μονοπάτι. Αυτό θα είχε σαν αποτέλεσμα να γινόντουσαν πάρα πολλές κλήσεις στο ARCore API, με πολλά συνεχόμενα downloads, uploads και ταυτόχρονα πολλές εγγραφές με στοιχεία στην Firestore. Όλα αυτά θα υπερφόρτωνε την εφαρμογή και θα μειώνε κατά πολύ την αποδοτικότητά και την απόδοσή της.

Έτσι δημιουργήθηκαν δύο μέθοδοι, που είχαν την ίδια λειτουργία, η μια για εσωτερικό και η άλλη για εξωτερικό χώρο. Ο admin τοποθετούσε Cloud Anchors μαζί με τα 3D μοντέλα τους σε κάποια κύρια σημεία (Checkpoints), μόνο αυτά γινόταν upload μέσω του API και ενημέρωναν την Firestore. Τα υπόλοιπα ενδιάμεσα 3D μοντέλα, που ένωναν τα κύρια σημεία και δημιουργούσαν το μονοπάτι, τοποθετούνταν αυτόματα με τις μεθόδους κάθε φορά εκ νέου από τον χρήστη (σε τοπικό επίπεδο), χωρίς δηλαδή να ενημερώνουν κάποια βάση. Αυτό γινόταν με την τοποθέτηση Nodes που «κουβαλούσαν» 3D αντικείμενα, στα ενδιάμεσα σημεία.



Εικόνα 16: *Utilities Folder*

Ψευδοκώδικας

Ψευδοκώδικας για τον αλγόριθμο, που αφορά το μοντέλο για Incremental Learning.

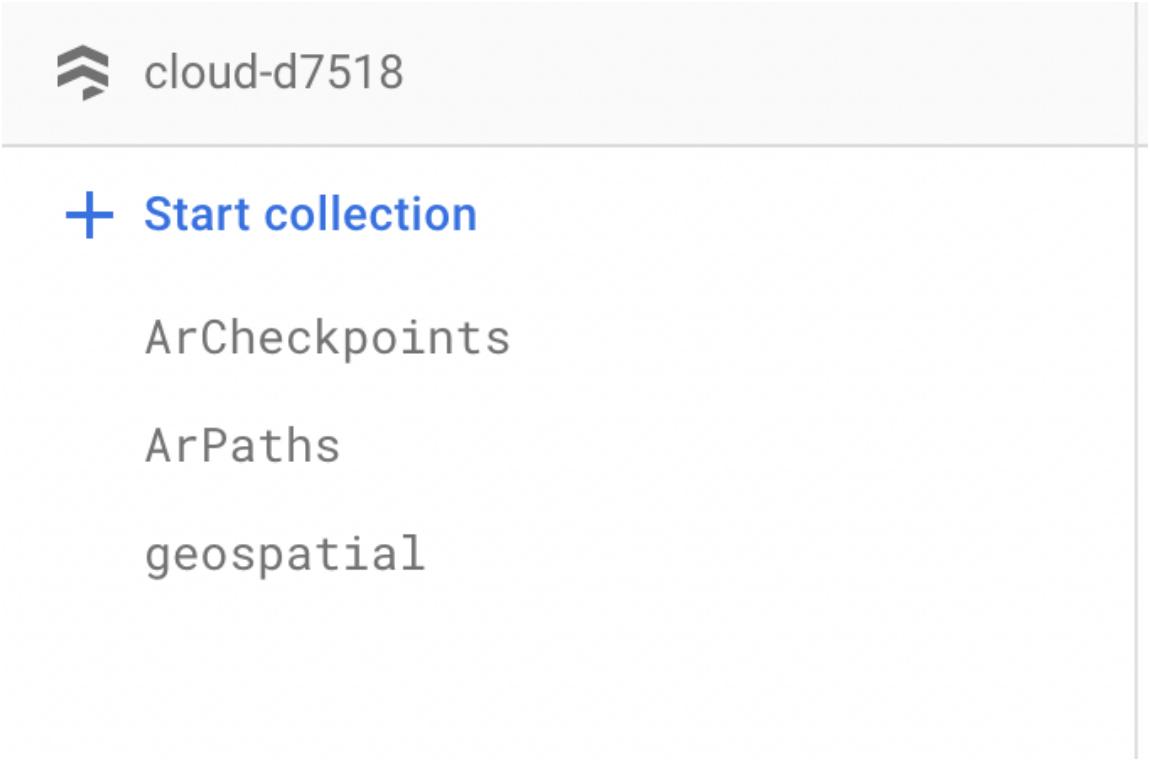
```
Checkpoints=[];

valuation=[];

Checkpoints=Download_from_Firestore('Anchors'); //Κατεβάζουμε απο την
Firestore όλο τον πίνακα με τα Anchors που έχει ενα Checkpoint

find=false; // flag για όταν κάνει resolve το Cloud Anchor

while(!find){

    for (int i=0; i<Checkpoints.length; i++){

    valuation.push(Checkpoints[i]['valuation']); //Φέρνει την τιμή της
μετρικής για κάθε Cloud Anchor

    id=Checkpoints[i]['cloudanchorid']; //cloudanchorid για κάθε Cloud
Anchor

    arAnchorManager.downloadAnchor(id); //resolving το Cloud Anchor

    }
}

//callback function για όταν ένα Cloud Anchor γίνεται resolve με παράμετρο
τα στοιχεία του

onDownloadAnchor(Cloud_Anchor){
  find=true; //stop while

  for (int i=0; i<Checkpoints.length; i++){  //πρέπει να ενημερώσει
      το σωστό Anchor στον πίνακα στην Firestore

      if(Cloud_Anchor.cloudanchorid=Checkpoints[i]['cloudanchorid']){
          new_valuation=valuation[i]+1;

          update_valuation_in_Firestore.(Cloud_Anchor.cloudanchorid)
        .('valuation')
          .parameter.(new_valuation);

          //Ενημερώνει το valuation του συγκεκριμένου Cloud Anchor
         στην Firestore

        break; // stop for-loop
      }
```

Περιγραφή Ψευδοκώδικα

- Αρχικά ορίζονται δύο άδειοι πίνακες και ένα flag Boolean.
- Έρχονται στο πίνακα ‘Checkpoint’ όλα τα στοιχεία (cloudanchorid, 3D Node, valuation) από τα Cloud Anchors που έχουν τοποθετηθεί σε ένα συγκεκριμένο χώρο (σημείο στο πραγματικό κόσμο).
- Στην συνέχεια, μπαίνει στο ‘while’ και μέσα στην ‘for’ παίρνει το cloudanchorid από κάθε Cloud Anchor και χρησιμοποιώντας εντολή από το ARCore API, κάνει resolve-download.
- Ταυτόχρονα, σώζει με την ίδια σειρά που έχει ο πίνακας Checkpoint, την τιμή από το valuation του κάθε Cloud Anchor.
- Η διαδικασία του resolving γίνεται για όλα τα Cloud Anchors του Checkpoint ταυτόχρονα, για αυτό είναι μέσα στο ‘while’.
- Το ARCore έχει μια callback function, η οποία ενεργοποιείται όταν κατέβει ένα Cloud Anchor και έχει ως παράμετρο τα στοιχεία του.
- Αυτό σημαίνει, ότι έχει βρεθεί ένα από όλα τα Cloud Anchors, που έγιναν resolve, και έχει τοποθετηθεί στο πραγματικό χώρο.
- Εκεί αλλάζει το flag για να σταματήσει το resolving για τα άλλα Cloud Anchor.
- Χρησιμοποιεί τον ήδη υπάρχων πίνακα, για να βρει την σωστή τιμή valuation του Cloud Anchor που βρέθηκε.
- Αυξάνει κατά ένα την τιμή του valuation.
- Ενημερώνει την Firestore, ότι το συγκεκριμένο Cloud Anchor έχει μια νέα τιμή.
- Τέλος σταματάει την διαδικασία ‘for-loop’.
- Όλο αυτό γίνεται για κάθε Checkpoint ενός μονοπατιού.

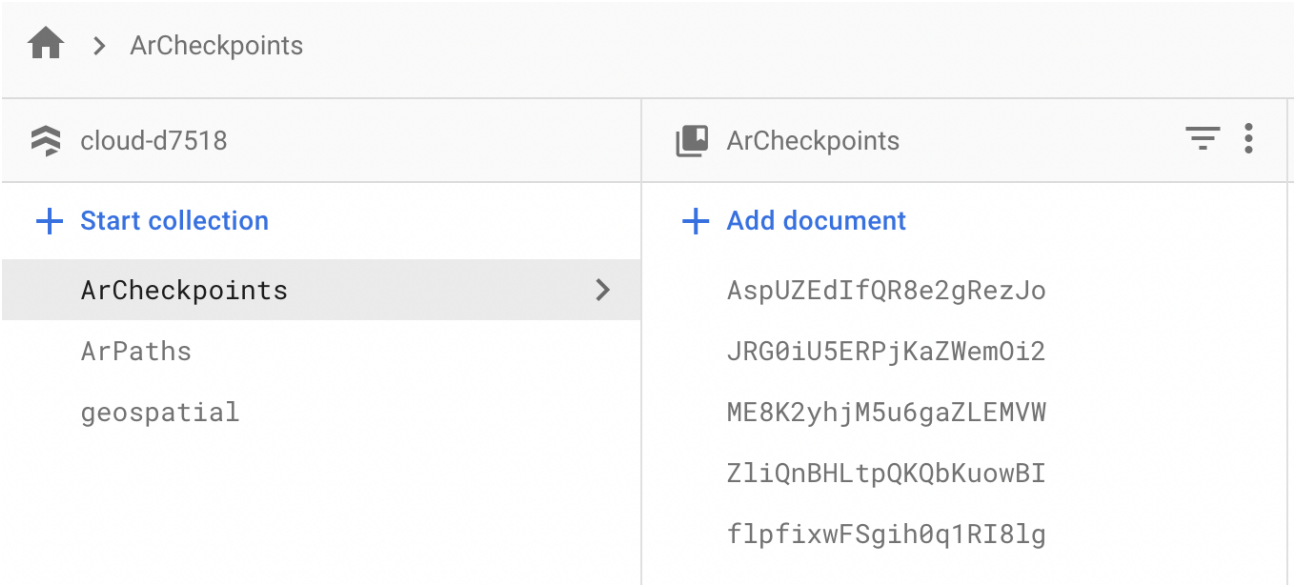
3.3.4 Βάση Δεδομένων

Για τις ανάγκες της εφαρμογής χρησιμοποιήθηκε η Firestore database της Firebase. Αρχικά δημιουργήθηκε ένα πρότζεκτ με το όνομα cloud και ορίστηκε μέσα από την εφαρμογή 3 collections. Όπως φαίνεται και στην ***Εικόνα 17***, δημιουργήθηκε η ArCheckpoints, ArPaths και geospatial. Οι δύο πρώτες αφορούσαν τις διαδρομές σε εσωτερικό χώρο, ενώ η τρίτη τους εξωτερικούς χώρους.



Εικόνα 17: *Collections in Firestore*

Στη Firestore κάθε νέα εισαγωγή (document) σε μια collection ορίζεται και από ένα τυχαίο DocumentId για να ξεχωρίζει από τις υπόλοιπες. Για παράδειγμα ***Εικόνα 18***, η ArCheckpoints φαίνεται ότι έχει 5 εισαγωγές.



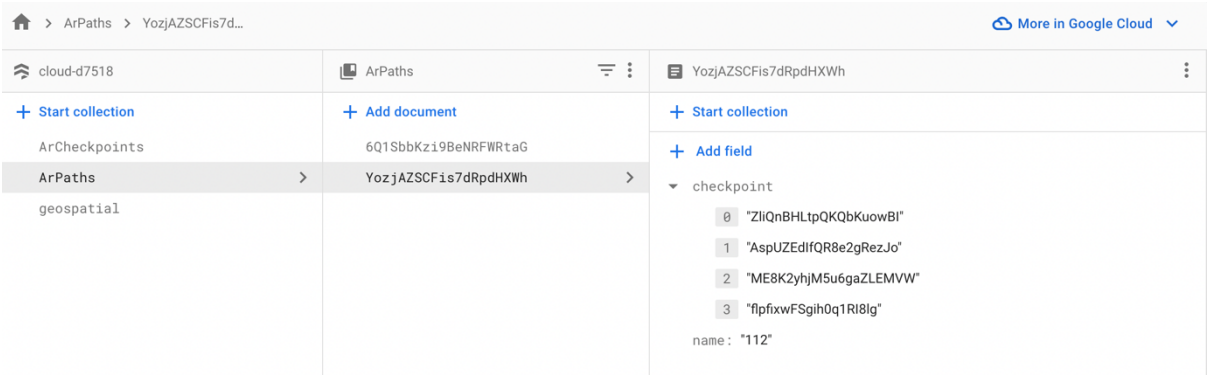
Εικόνα 18: *Documents of ArCheckpoints*

ArPaths

Αρχικά η collection ArPaths δημιουργείται κατά την πρώτη φορά που ο διαχειριστής ορίζει μια διαδρομή μέσω της εφαρμογής. Όπως φαίνεται και στην ***Εικόνα 19***, έχει δύο εγγραφές, άρα δύο διαδρομές που μπορούν να εμφανιστούν σαν επιλογές στην εφαρμογή. Το κάθε document έχει δύο πεδία:

Checkpoint: είναι ένας πίνακας που απαρτίζεται από τα εσωτερικού χώρου σημεία (Checkpoints), που χρειάζονται για την πλοήγηση στο χώρο, τα οποία τοποθετούνται από τον admin κατά την διάρκεια της χαρτογράφησης. Τα οποία όμως είναι κωδικοποιημένα με DocumentId μιας και το περιεχόμενο που τα χαρακτηρίζει θα είναι μια νέα εγγραφή στην ArCheckpoints collection. Οπότε συνοπτικά ο πίνακας Checkpoint περιέχει τις εγγραφές της ArCheckpoints, που αντιπροσωπεύουν σημεία μιας διαδρομής στο χώρο.

Name: είναι το όνομα της διαδρομής. Πιο συγκεκριμένα είναι ο τελικός προορισμός που θέλει να φτάσει ο χρήστης. Όλα τα Checkpoints που αναφέρονται στο πίνακα, πάνω από αυτό, θα εμφανιστούν μόνο όταν ο χρήστης καλέσει την διαδρομή με αυτόν τον τελικό προορισμό.



Εικόνα 19: *Fields of ArPaths*

ArCheckpoints

Η δεύτερη collection που είχε η βάση είναι η ArCheckpoints. Η συγκεκριμένη περιέχει τα χαρακτηριστικά που έχει κάθε σημείο μιας διαδρομής. Όπως αναφέρθηκε και παραπάνω, τα DocumentId, δηλαδή οι εγγραφές που έχει, άμα ομαδοποιηθούν απαρτίζουν μια διαδρομή και περιέχονται συγκεντρωμένα σε ένα πίνακα στην ArPaths. Το κάθε Checkpoint περιέχει τα εξής πεδία:

Anchors: είναι ένας πίνακας με χαρακτηριστικά των Cloud Anchors. Το πρώτο Cloud Anchor έχει τοποθετηθεί από τον admin στο συγκεκριμένο, εσωτερικού χώρου, σημείο. Όπως ήδη ειπώθηκε ένας χρήστης μπορεί κατά τη διάρκεια της διαδρομής να τοποθετήσει εκ νέου Cloud Anchors πάνω στο ήδη υπάρχον, με τυχόν καλύτερο, Feature Map. Έτσι ένα σημείο μπορεί να έχει πάνω από ένα Cloud Anchor καρφιτσωμένο. Για αυτό και ο πίνακας έχει πολλά Cloud Anchors που το κάθε ένα έχει κάποια δικά του χαρακτηριστικά:

- childNodes: είναι ένας πίνακας με ονόματα από Nodes (by default δίνονται) τα οποία έχουν τοποθετηθεί πάνω στο Cloud Anchor και κατ' επέκταση στο σημείο και στην προκειμένη περίπτωση κάθε Node είναι ένα 3D μοντέλο.

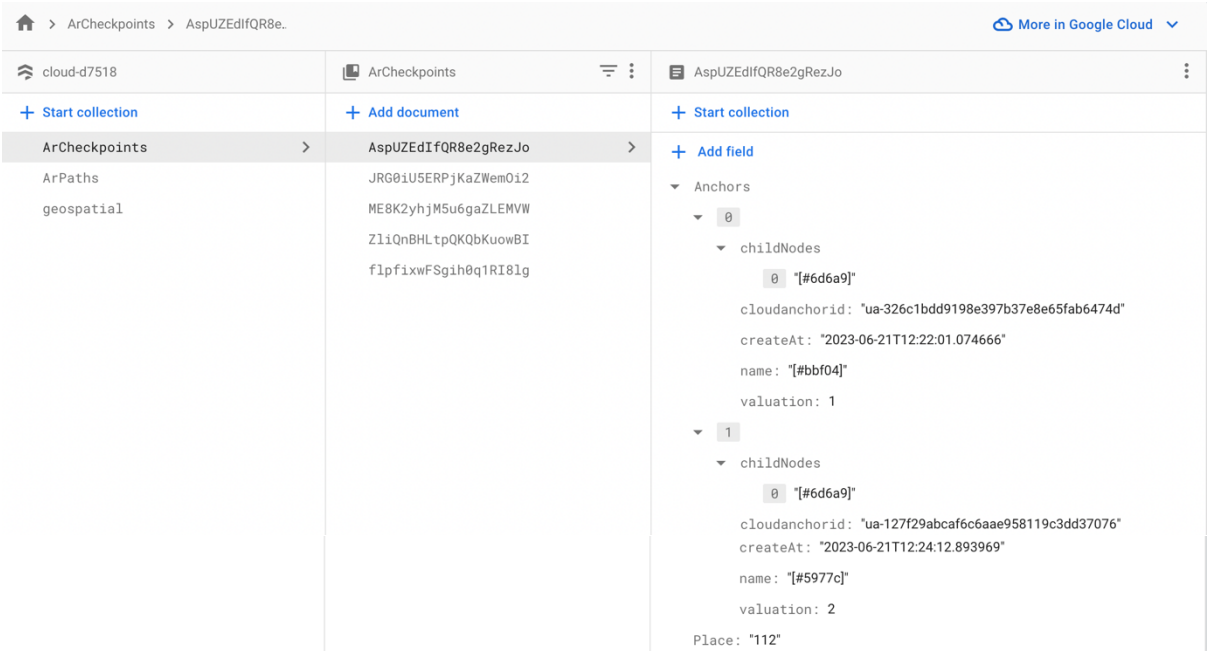
- cloudanchorid: το cloudAnchorId είναι μια μοναδική ταυτότητα που χρησιμοποιείται στην Επαυξημένη Πραγματικότητα για να αναγνωρίσει ένα συγκεκριμένο (Cloud Anchor) που έχει αποθηκευτεί στο cloud. Όταν ο χρήστη θέλει να φέρει τα Cloud Anchors που αντιστοιχούν στην διαδρομή που έχει επιλέξει, στην ουσία ζητάει «να κατέβουν» μέσω του ARCore API με το συγκεκριμένο cloudanchorid.

- createAt: η ημερομηνία και η ώρα που δημιουργήθηκε το Cloud Anchor.

- name: το όνομα που δίνεται αυτόματα στο Cloud Anchor.

- valuations: μια μετρική που να κρίνει ποιο από τα Cloud Anchors που έχει τοποθετηθεί πάνω σε ένα σημείο είναι το καλύτερο και έρχεται πιο πολλές φορές, έτσι ώστε μετέπειτα να γίνουν κάποια τεστ πάνω στο Incremental Learning της εφαρμογής.

Place: είναι το όνομα της διαδρομής. Πιο συγκεκριμένα είναι ο τελικός προορισμός που θέλει να φτάσει ο χρήστης. Όλα τα Checkpoints χαρακτηρίζονται από την διαδρομή που ανήκουν.



Εικόνα 20: *Fields of ArCheckpoints*

Geospatial

Η τρίτη και τελευταία collection που έχει στην βάση είναι geospatial. Η συγκεκριμένη έχει όλες τις πληροφορίες για τα Geospatial Anchors, άρα αφορά τα σημεία σε εξωτερικούς χώρους. Κάθε document αφορά μια συγκεκριμένη διαδρομή, η οποία πάντα καταλήγει σε ένα σημείο εσωτερικού χώρου. Τα πεδία που έχει κάθε εγγραφή είναι:

Coordinates: ένα πίνακας με τα χαρακτηριστικά που πρέπει να έχουν τα Geospatial Anchors για να εμφανιστούν στο σωστό σημείο του AR περιβάλλοντος. Επίσης ο πίνακας περιέχει τόσα στοιχεία όσα και τα Geospatial Anchors για μια εξωτερικού χώρου διαδρομή.

- Alt: το ύψος από την επιφάνεια της Γης.

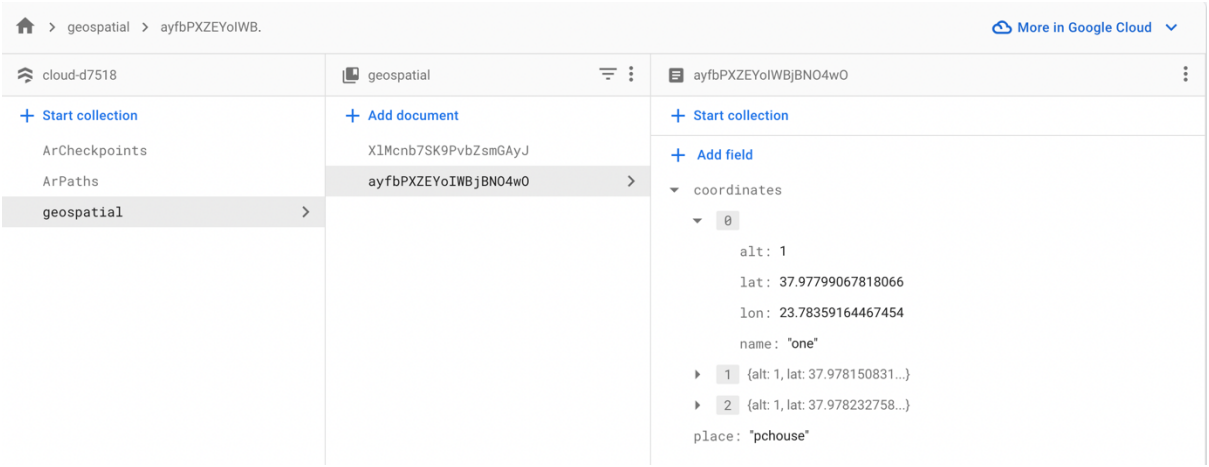
- Lat: το γεωγραφικό πλάτος του σημείου στην επιφάνεια της Γης.

- Lon: το γεωγραφικό μήκος του σημείου στην επιφάνεια της Γης.

- Name: η σειρά με την οποία θέλουμε να εμφανιστούν τα Geospatial Anchors.

Στην ουσία το alt/lat/lon αντιπροσωπεύουν ένα μοναδικό σημείο στην γη, στο οποίο πρέπει να εμφανιστεί το Geospatial Anchor.

Place: το μέρος-κτίριο που καταλήγει αυτή η εξωτερική διαδρομή από σημεία. Γενικά αυτό το χαρακτηριστικό είναι πολύ χρήσιμο, γιατί αν υπάρχουν πολλά τελικά σημεία στο ίδιο εσωτερικό χώρο, θα μπορούσαν να χρησιμοποιηθούν οι ίδιες συντεταγμένες (ίδιο document της βάσης) για να φτιαχτεί η διαδρομή που οδηγεί στο κτίριο που βρίσκονται.



Εικόνα 21: *Fields of geospatial*

Κεφάλαιο 4: Το σύστημα στην πράξη

4.1 Εγκατάσταση Συστήματος

Το σύστημα έχει εγκατασταθεί στο κεντρικό χώρο του ΕΜΠ. Πιο συγκεκριμένα, σαν αρχικό εξωτερικό σημείο στο οποίο στην ουσία ξεκινάει η πλοήγηση εξωτερικού χώρου, είναι η κεντρική πλατεία έξω από την βιβλιοθήκη. Επίσης, πιλοτικά επιλέχθηκαν δύο παραδείγματα για σημεία τελικού στόχου σε εσωτερικό κτίριο. Το ένα είναι το γραφείο του κ. Τσανάκα στο κτίριο υπολογιστών και το άλλο το γραφείο του πρύτανη στο κτίριο διοίκησης.

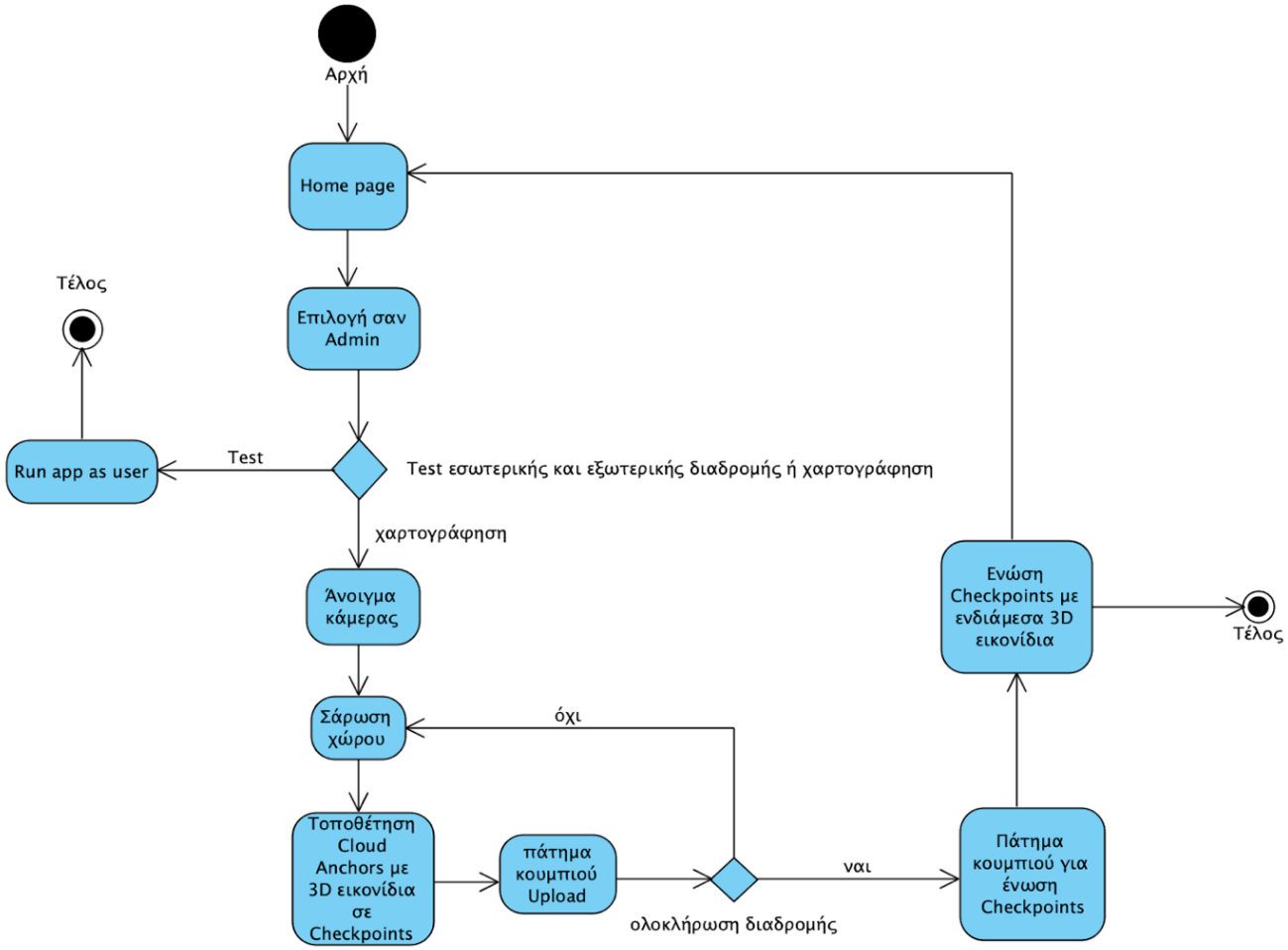
4.2 App Flow

Στα παρακάτω διαγράμματα θα δείξουμε τα App Flow της εφαρμογής, αρχικά για τον admin και έπειτα για τον user. Μπορεί να συνυπάρχουν στο ίδιο συνολικό UI, αλλά θα μπορούσαν να ήταν και τελείως αυτόνομες εφαρμογές.

Admin

Στη περίπτωση του admin θα αναφέρουμε μόνο την χαρτογράφηση, μιας και το τεστ κομμάτι που μπορούσε να κάνει, συμπεριλαμβάνεται στο App Flow του user.

- Στη αρχική οθόνη ο admin επιλέγει το κουμπί που τον μεταφέρει στην οθόνη του διαχειριστή.
- Εκεί μπορεί να επιλέξει από μια γκάμα κουμπιών, μερικά είναι για να χαρτογραφήσει μια διαδρομή και άλλα για να την τεστάρει.
- Επιλέγει μια εσωτερική διαδρομή για χαρτογράφηση.
- Ανοίγει αυτόματα η κάμερα.
- Σαρώνει την περιοχή με την κάμερα και τοποθετεί ένα Cloud Anchor μαζί με το 3D μοντέλο του, αγγίζοντας ένα σημείο σε μια από τις ειδικές περιοχές που θα του εμφανίσει το AR περιβάλλον.
- Στην συνεχεία πατάει το κουμπί upload για να στείλει με το ARCore API στην cloud βάση της ARCore το συγκεκριμένο Anchor με το Feature Map του και ταυτόχρονα να ενημερώσει την δικιά μας βάση με τις πληροφορίες του.
- Επαναλαμβάνει την διαδικασία μέχρι να φτάσει στον προορισμό τοποθετώντας σε σχετικά κοντινές αποστάσεις Cloud Anchors μαζί με το 3D μοντέλο τους.
- Τέλος ενεργοποιεί τη μέθοδο, που είχαμε δημιουργήσει για να ενώνονταν τα Cloud Anchors με νέα 3D αντικείμενα στο ενδιάμεσο.

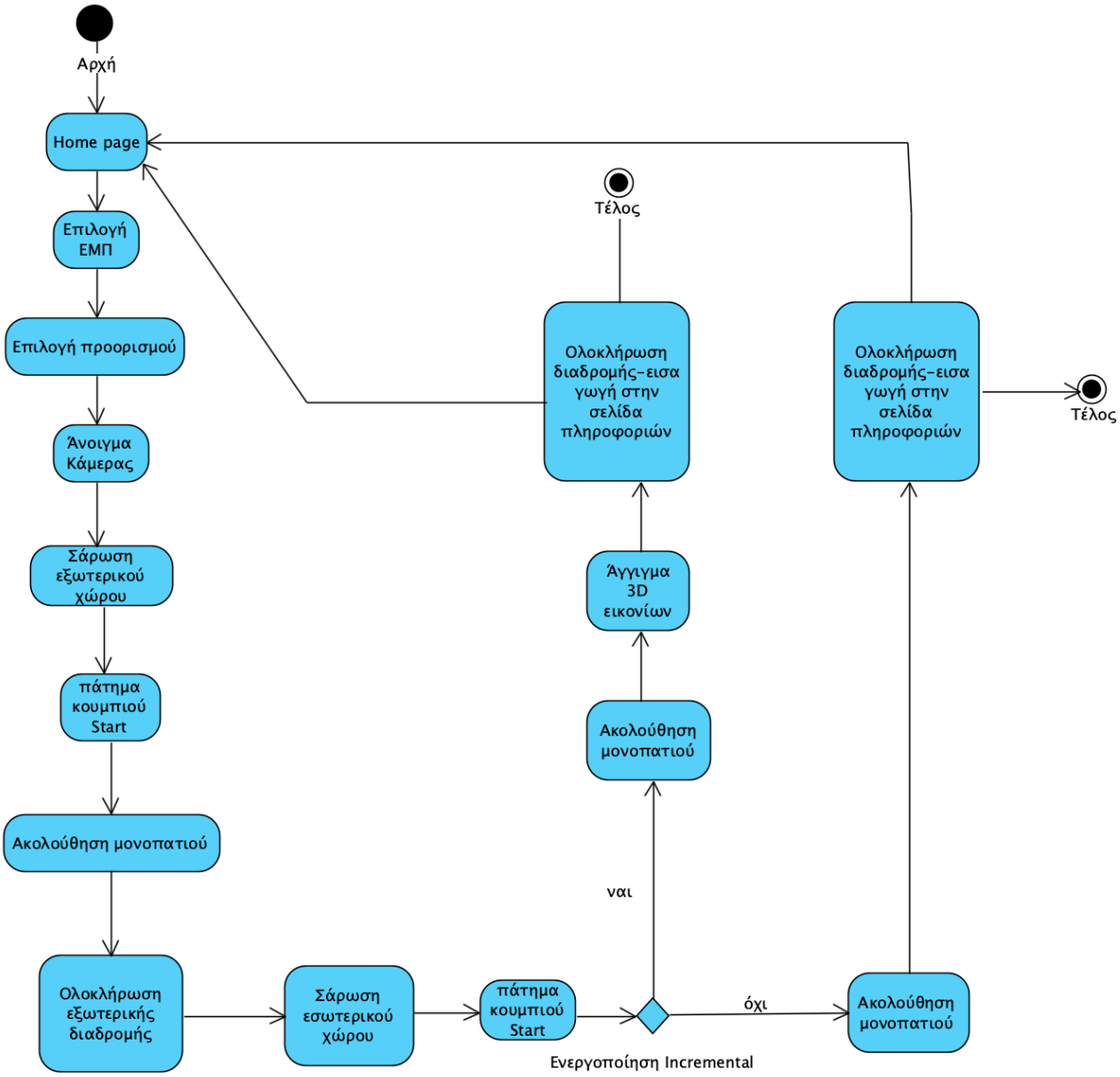


Εικόνα 22: *Admin App Flow Diagram*

User

Ο χρήστης καλό θα ήταν να βρίσκεται κοντά στο αρχικό σημείο (κεντρική πλατεία ΕΜΠ) για να μπορέσει να διακρίνει το πρώτο εξωτερικό 3D μοντέλο.

- Στη αρχική οθόνη ο user επιλέγει το κουμπί ΕΜΠ για την επόμενη οθόνη.
- Εκεί μπορεί να επιλέξει από μια γκάμα κουμπιών, που αντιστοιχούν σε πιθανούς τελικούς προορισμούς μέσα στο campus του ΕΜΠ.
- Αφού κάνει την επιλογή, ανοίγει η κάμερα.
- Σαρώνει τον εξωτερικό χώρο.
- Πατώντας το κουμπί ‘Start’ εμφανίζονται τα 3D μοντέλα και δημιουργείται το μονοπάτι.
- Ακολουθάει το μονοπάτι.
- Όταν πλησιάζει στο κτίριο, εμφανίζεται μια προειδοποίηση και στην συνέχεια σε βάζει απευθείας εντός.
- Σαρώνει τον εσωτερικό χώρο.
- Πατώντας το κουμπί ‘Start’ εμφανίζονται τα 3D μοντέλα μοντέλο και δημιουργείται το μονοπάτι.
- Ακολουθεί το μονοπάτι και αν έχει κάνει την επιλογή για να ενεργοποιήσει το Incremental Learning, μπορεί να ακουμπάει κάθε 3D μοντέλο της διαδρομής.
- Όταν πλησιάζει τον στόχο, εμφανίζεται μια προειδοποίηση και ένα νέο κουμπί, που σε πηγαίνει στην οθόνη με τις πληροφορίες για τον τελικό στόχο.



Εικόνα 23: *User App Flow Diagram*

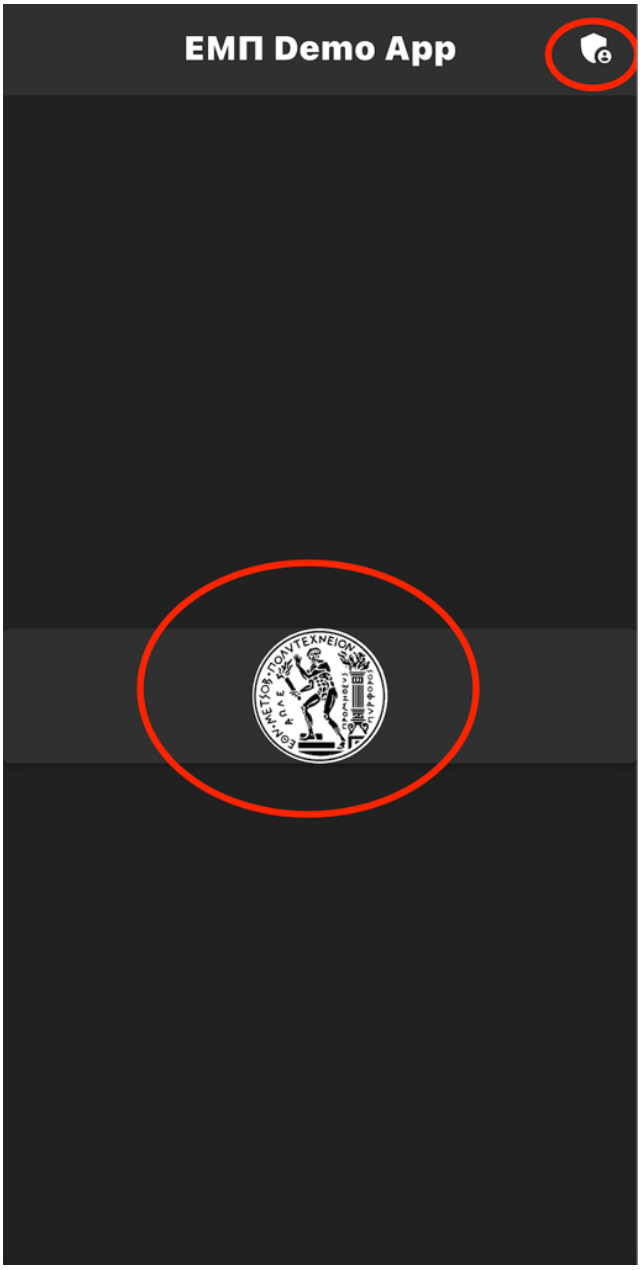
4.3 Οθόνες Εφαρμογής

Σε αυτή την ενότητα παρουσιάζονται αναλυτικά οι βασικές οθόνες της εφαρμογής, ώστε να δώσουμε μια ολοκληρωμένη εικόνα των λειτουργιών της τόσο θεωρητικά όσο και οπτικά. Ακολουθούν οι εικόνες και οι περιγραφές τους.

Αρχική οθόνη

Αρχικά στην ***Εικόνα 24***, φαίνεται η πρώτη οθόνη της εφαρμογής. Στην ουσία εδώ μπορεί να επιλεχθεί, αν θα τρέξει η εφαρμογή από την μεριά του χρήστη ή του διαχειριστή. Υπάρχουν δύο βασικά κουμπιά, ένα κεντρικό που σε βάζει σαν χρήστη και έχει το ΕΜΠ σαν εικόνα, μιας

και για αυτό είναι η εφαρμογή και ένα δεύτερο πάνω δεξιά (με μια ασπίδα για εικονίδιο), που σε βάζει στο admin panel και αφορά καθαρά το κομμάτι του διαχειριστή.



Εικόνα 24: *Αρχική Οθόνη*

Admin panel

Αν στην αρχική οθόνη επιλέξουμε το κουμπί για το admin panel τότε μας βάζει στην οθόνη που φαίνεται στη ***Εικόνα 25***. Εδώ ο admin έχει πολλές δυνατότητες. Καταρχήν έχει 7 κουμπιά, ένα πάνω αριστερά που είναι το βελάκι, το οποίο οδηγεί στην προηγούμενη οθόνη και τα υπόλοιπα 6 που αφορούν τις βασικές λειτουργίες της εφαρμογής. Τα πρώτα δύο είναι για την κύρια λειτουργία του admin, που είναι η χαρτογράφηση. Για το συγκεκριμένο version της εφαρμογής έχουμε μόνο δύο (εσωτερικά) τελικά σημεία, οπότε και δύο χαρτογραφήσεις που πρέπει να γίνουν. Τα υπόλοιπα 4 είναι κυρίως για το τεστάρισα, αφού τα δύο είναι για τον

έλεγχο τον εξωτερικών διαδρομών και αν πρέπει να γίνουν αλλαγές στις συντεταγμένες στην βάση, ενώ τα άλλα δύο είναι για τον έλεγχο των εσωτερικών, σαν να ήταν ο user.

Εικόνα 25: *Admin panel*

Από τα παραπάνω κουμπιά θα αναλύσουμε για τον admin μόνο ένα εκ των δύο πρώτων, που αφορά τη χαρτογράφηση, γιατί τα υπόλοιπα 4 θα τα δείξουμε μέσα από τις οθόνες που αφορούν τον user.

Χαρτογράφηση

Για παράδειγμα διαλέγουμε από τα παραπάνω, το δεύτερο κουμπί που αφορά τη χαρτογράφηση για το γραφείο του πρύτανη. Στην πρώτη εικόνα φαίνεται ότι ανοίγει αυτόματα η κάμερα και αφού σαρώσουμε τον χώρο εμφανίζονται οι περιοχές (σαν αχανή ορθογώνια)

στα οποία με το άγγιγμα ο admin μπορεί να τοποθετήσει ένα Cloud Anchor με το 3D Node του. Επίσης έχει και 4 κουμπιά, το πρώτο είναι το βελάκι που οδηγεί στην προηγούμενη οθόνη, το δεύτερο που είναι πάνω δεξιά και μοιάζει σαν τικ, είναι για την ενεργοποίηση της μεθόδου ένωσης των Cloud Anchors που είναι στα Checkpoints και γίνεται στο τέλος της χαρτογράφησης, το κουμπί Download είναι αυτό που κατεβάζει τα Cloud Anchors μέσω του API χρησιμοποιώντας τα στοιχεία που έχει η Firestore και τέλος υπάρχει και το Remove Everything που διαγράφει τα Cloud Anchors πριν όμως γίνουν Upload.

Εικόνα 26: *Οθόνη χαρτογράφησης 1*

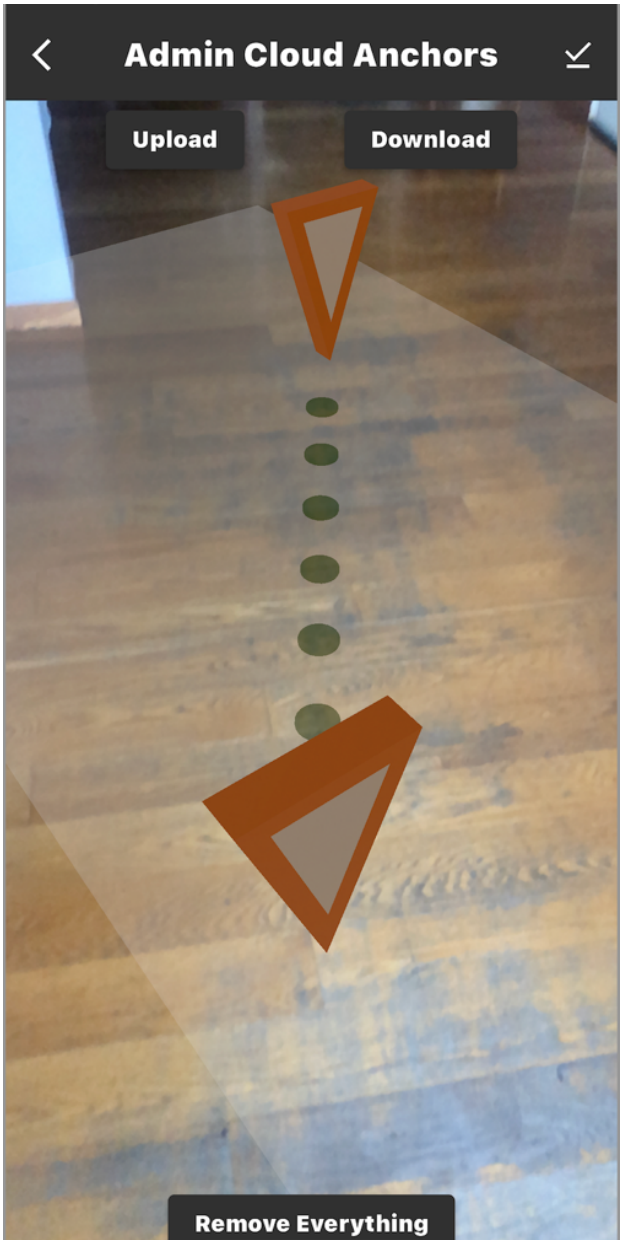
Εφόσον γίνει καλή καταγραφή του γύρου χώρου αγγίζοντας ένα σημείο πάνω στο αχνό πλαίσιο, τοποθετείται ένα Cloud Anchor με το 3D μοντέλο του (ένα βέλος πορτοκαλί) που αντιπροσωπεύει ένα Checkpoint. Επίσης εμφανίζεται άλλο ένα κουμπί, το Upload, που πατώντας το, το ανεβάζει στο συγκεκριμένο endpoint του API με το Feature Map και επιστρέφει το κλειδί του, που το αποθηκεύουμε στην Firestore.

Εικόνα 27: *Οθόνη χαρτογράφησης 2*

Ένα ένα μετά την τοποθέτηση πατώντας το κουμπί Upload γίνεται η διαδικασία ενημέρωσης της Cloud βάσης της ARCore μέσω του API αλλά και της δικιά μας Firestore. Εφόσον όλα πάνε καλά και δεν υπάρξει κάποιο error εμφανίζεται το μήνυμα που φαίνεται και στην παρακάτω Εικόνα.

Εικόνα 28: *Οθόνη χαρτογράφησης 3*

Μόλις τοποθετήσουμε όλα τα Checkpoints που θέλουμε για να φτάσουμε κοντά στο τελικό στόχο, μπορούμε να πατήσουμε το πάνω δεξιά κουμπί για να ενωθούν τα βελάκια με νέα ενδιάμεσα 3D Node (χωρίς να ανεβαίνουν σε κάποια βάση) τα οποία θα είναι πράσινες κουκίδες.

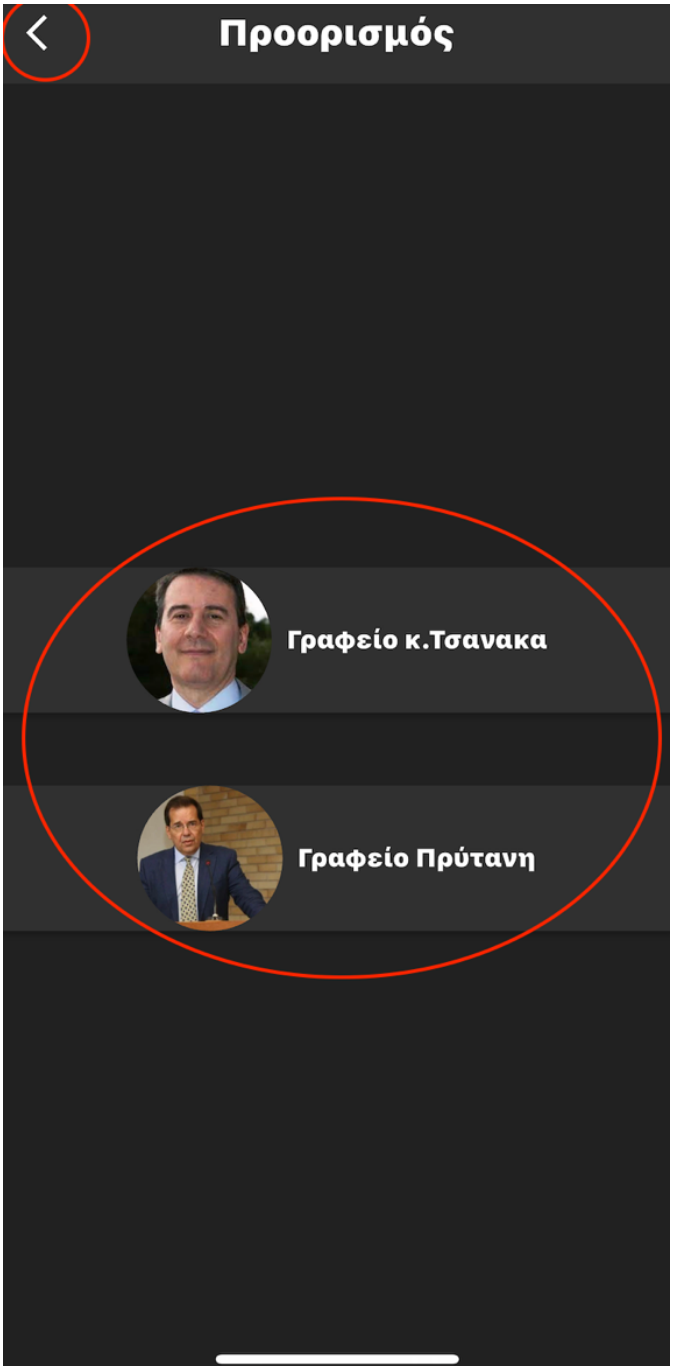


Εικόνα 29: *Οθόνη χαρτογράφησης 4*

Μετά από την ένωση ο Admin μπορεί να πάει στην πίσω οθόνη και να δοκιμάσει τη διαδρομή σαν χρήστης ή να κάνει Remove Everything και με το Download να εμφανίσει ξανά το μονοπάτι από την αρχή.

User mode

Ο User της εφαρμογής καθώς βρίσκεται στην αρχική οθόνη, μπορεί με το κεντρικό κουμπί που έχει το σήμα το ΕΜΠ για να προχωρήσει στην επόμενη. Εκεί θα βρει 3 κουμπιά, το ένα είναι το βελάκι για την προηγούμενη οθόνη και τα άλλα δύο είναι επιλογές που μπορεί να κάνει για τελικούς προορισμούς στον χώρο του πολυτεχνείου σε αυτό το version της εφαρμογής, ***Εικόνα 33***.

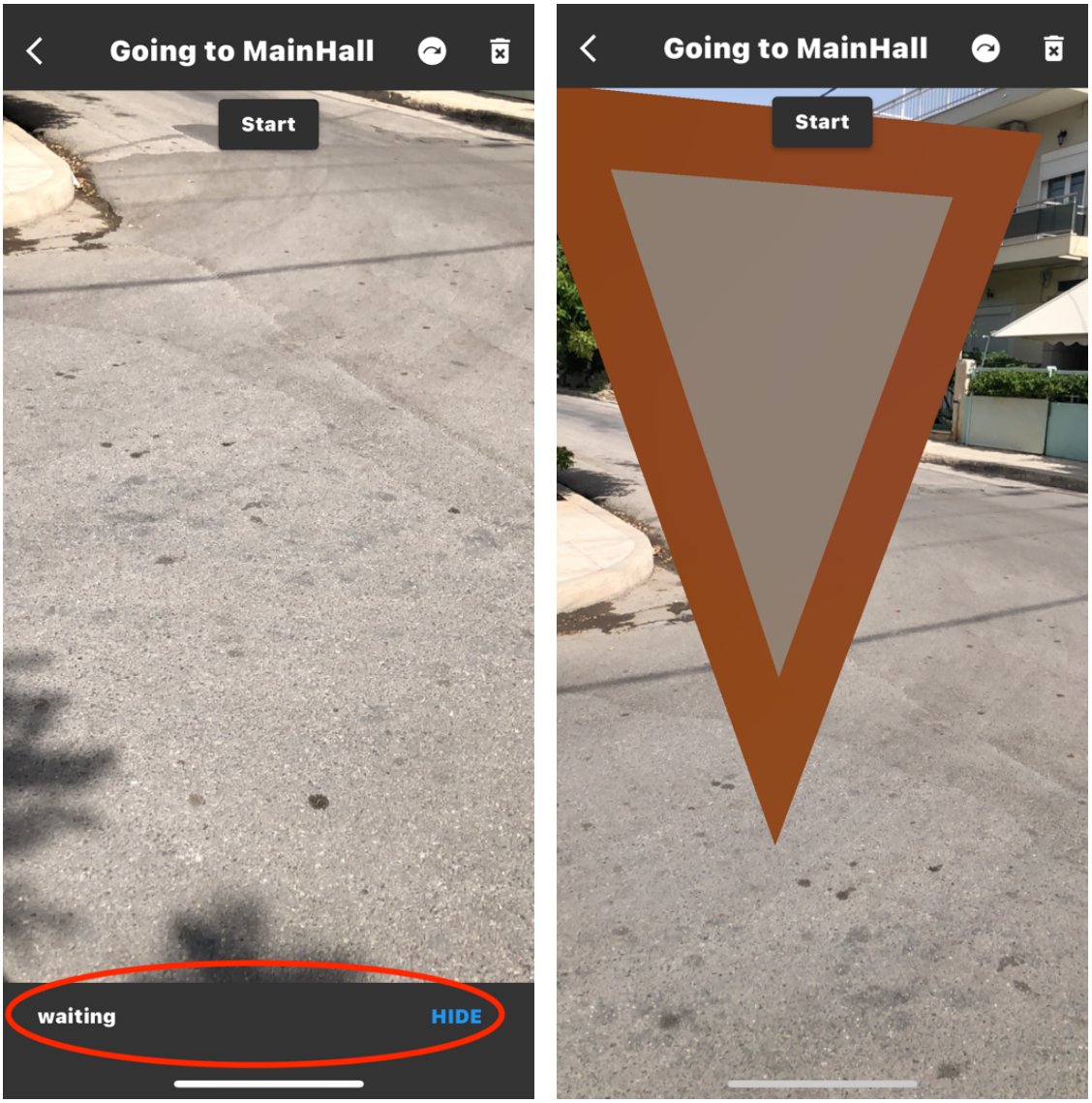


Εικόνα 30: *Οθόνη επιλογής Προορισμού*

Εφόσον γίνει κάποια επιλογή, ανοίγει η κάμερα και ξεκινάει η διαδικασία από την εξωτερική διαδρομή. Εκεί αρχικά υπάρχουν 4 κουμπιά, το πρώτο είναι πάλι το βελάκι, μετά υπάρχει και ένα διαφορετικό βελάκι στην πάνω δεξιά μεριά, το οποίο μπορεί να παρακάμψει την διαδικασία εξωτερικού χώρου, μιας και μπορεί ο χρήστης να βρίσκεται ήδη στο κτίριο όπου βρίσκεται ο τελικός στόχος. Το τρίτο κουμπί που είναι δίπλα στο προηγούμενο και έχει ένα κάδο για εικονίδιο, αφορά τη διαγραφή των Geospatial Anchors που θα εμφανιστούν. Τέλος υπάρχει και το κουμπί start το οποίο εμφανίζει-τοποθετεί σύμφωνα με τις συντεταγμένες στη βάση τα Geospatial Anchors με τα 3D Nodes τους (βελάκια πορτοκαλί) στον εξωτερικό χώρο του πολυτεχνείου.

Εικόνα 31: *Οθόνη για διαδρομή εξωτερικού χώρου 1*

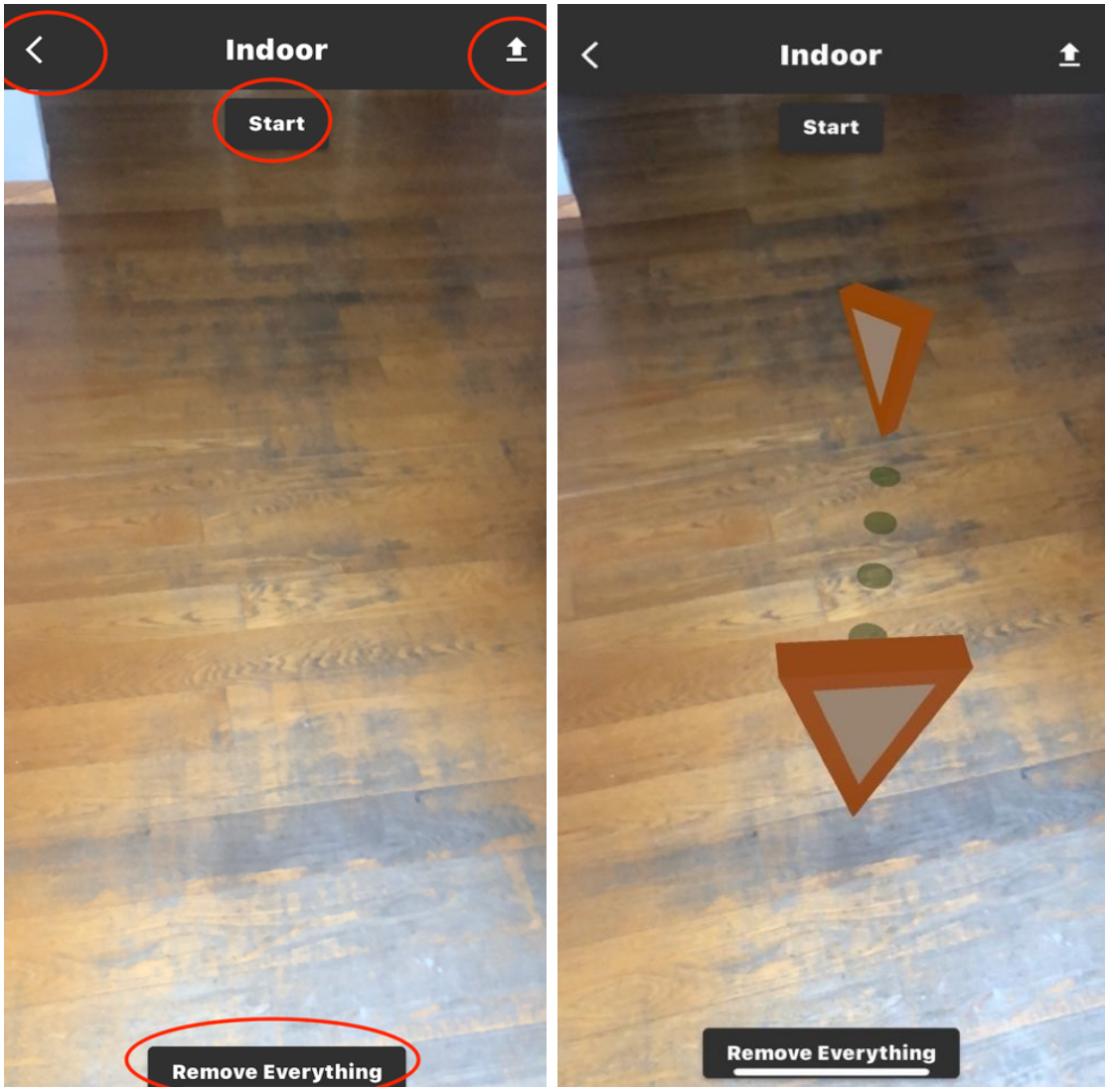
Μέχρι να εμφανιστούν όλα τα Geospatial Anchors της διαδρομής, βγαίνει μια ειδοποίηση που λέει waiting και όταν τελειώσει αυτή η διαδικασία, αυτόματα ενώνονται με την μέθοδο που έχει δημιουργηθεί (πράσινες κουκίδες) και ο χρήστης πρέπει να αρχίσει να ακολουθεί το μονοπάτι. Πλησιάζοντας στο τελευταίο Geospatial Anchor εμφανίζεται ένα μήνυμα που ενημερώνει τον user ότι πλησιάζει το κτίριο και όταν πλέον είναι σχεδόν από πάνω η εφαρμογή τον βάζει στο εσωτερικό του για την δεύτερη διαδρομή.



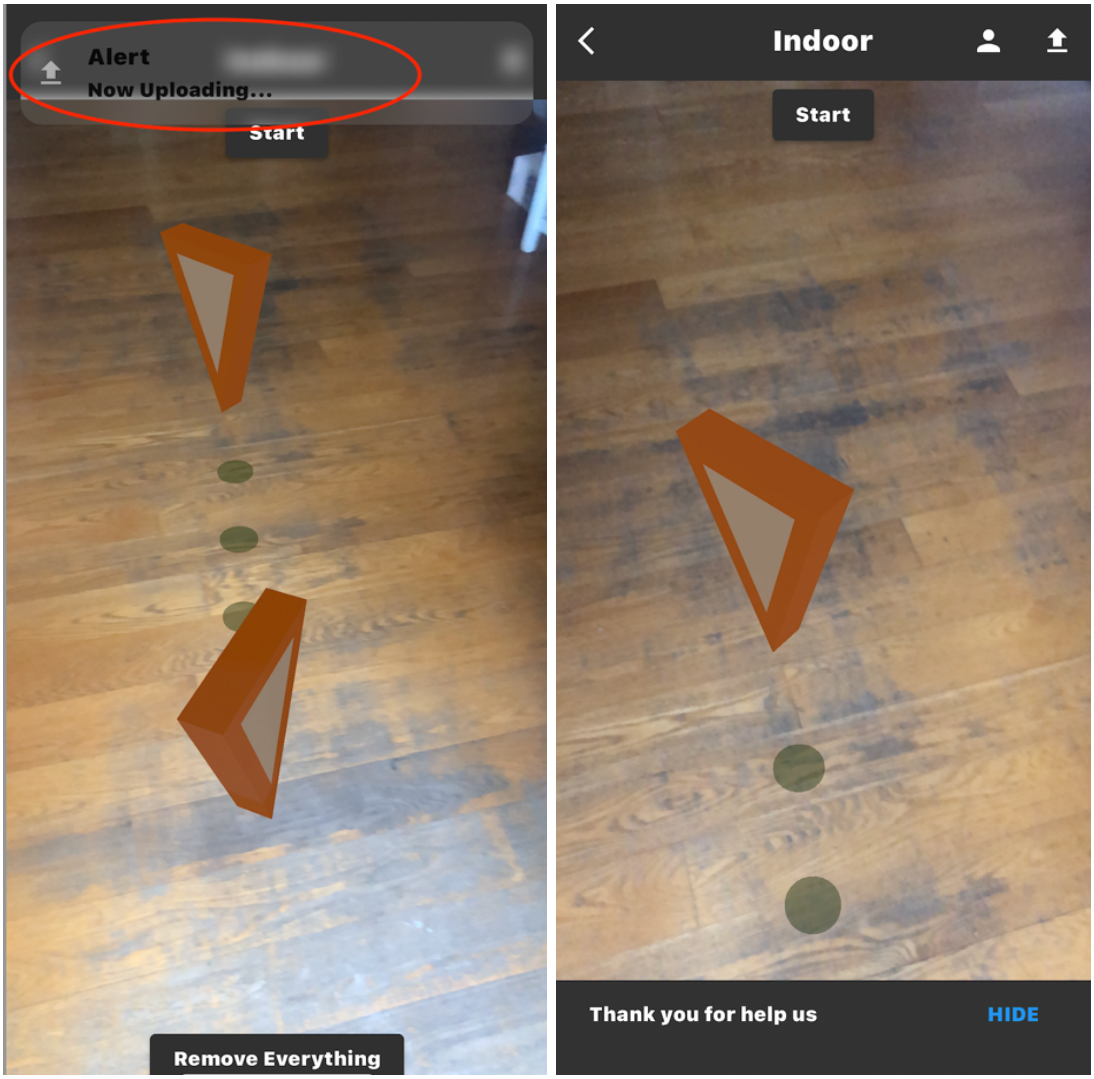
Εικόνα 32: *Οθόνη 'waiting' για διαδρομή εξωτερικού χώρου και εμφάνιση 3D μοντέλο*

Εικόνα 33: *Οθόνη Alert για διαδρομή εξωτερικού χώρου*

Στην συνέχεια, η επόμενη οθόνη αφορά διαδρομή εσωτερικού χώρου. Έχει 4 κουμπιά, τα δυο είναι το βελάκι για πίσω και το Remove Everything, όπως πριν. Επίσης υπάρχει το κουμπί start που κατεβάζει τα Cloud Anchors με τα 3D Nodes και ενεργοποιεί και την μέθοδο που τα ενώνει. Τέλος πάνω δεξιά έχει ένα κουμπί με εικονίδιο, ένα μεγάλο βέλος προς τα πάνω (σαν Upload), εφόσον ο χρήστης το πατήσει εμφανίζεται ένα μήνυμα που γράφει (now Uploading) και αφορά την ενεργοποίηση του Incremental Learning. Ο user μπορεί να ακουμπάει κάθε 3D μοντέλο για να ανανεώνει το Feature Map του Cloud Anchor όπως έχει αναφερθεί. Αν το κάνει για όλα εμφανίζεται και ένα μήνυμα με ευχαριστίες.

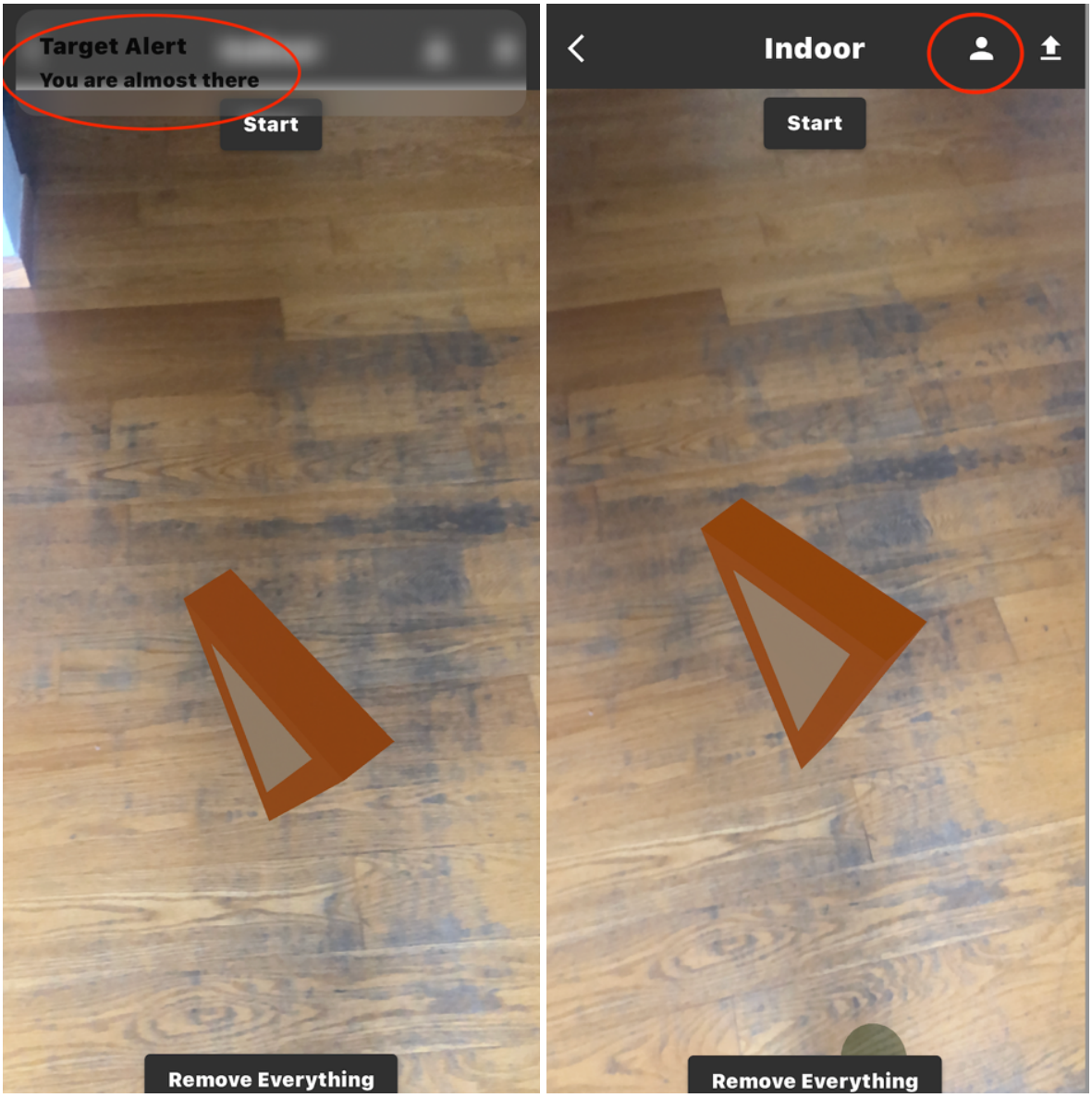


Εικόνα 34: *Οθόνη για διαδρομή εσωτερικού χώρου 1 και οθόνη μετά στο κουμπί Start*



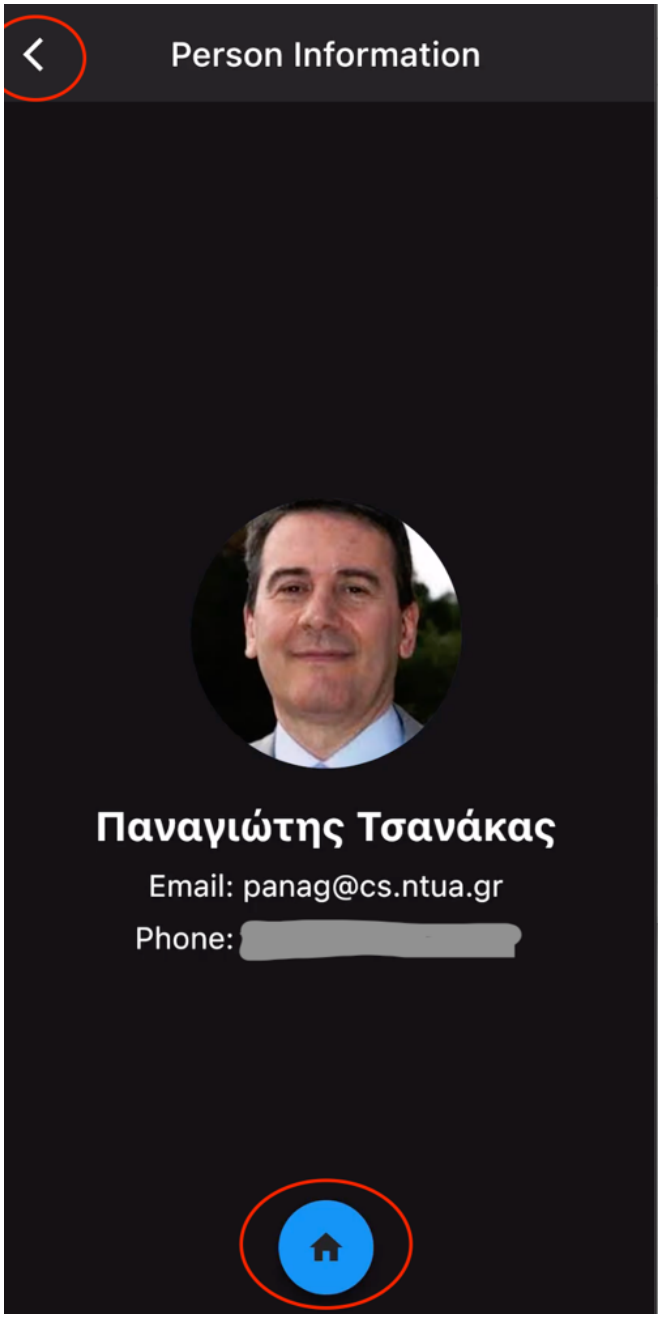
Εικόνα 35: *Μετά το Upload button και μήνυμα μετά το Incremental Learning*

Όταν φτάσει κοντά στον τελικό στόχο, πάλι εμφανίζεται ένα μήνυμα που τον ενημερώνει ότι είναι κοντά και όταν πάει σχεδόν από πάνω εμφανίζεται ένα νέο κουμπί πάνω δεξιά με το σχήμα ατόμου, που τον βάζει στην οθόνη πληροφοριών.



Εικόνα 36: *Μήνυμα όταν είναι κοντά στο τελικό στόχο και ενεργοποίηση κουμπιού για πληροφορίες*

Τέλος, εκεί μπορεί να βρει πληροφορίες για αυτόν που διάλεξε σαν τελικό προορισμό. Υπάρχει επίσης και ένα μεγάλο κουμπί με εικονίδιο σπιτιού που επιστρέφει τον user στην αρχική οθόνη.



Εικόνα 37: *Σελίδα πληροφοριών*

Κεφάλαιο 5: Σύνοψη

5.1 Αποτελέσματα & Μετρήσεις

Αφού ολοκληρώθηκε η κατασκευή της εφαρμογής, έγιναν 50 δοκιμές που αφορούσαν το μοντέλο Incremental Learning και κατά ποσό βελτίωνε την εφαρμογή. Σαν κριτήριο για την βελτίωση, είχαμε τον χρόνο που χρειάζεται για να κάνει resolve ένα Cloud Anchor και να εμφανιστεί στο χρήστη. Τα πειράματα έγιναν με δύο κινητά τηλέφωνα με iOS λογισμικά, ένα iPhone X (40 δοκιμές) και ένα iPhone 12 (10 δοκιμές).

Στους παρακάτω πίνακες παρουσιάζεται ο χρόνος του resolve ανάλογα με την κάθε περίπτωση.

Γραμμές: Αλλαγές στο χώρο (ναι/όχι)
Στήλες: Cloud Anchors σε κάθε Checkpoint (νούμερο)

iPhone X

Resolving time	1 (admin)	2	4	6	10	15	20
OXI	5,5 sec	4,5 sec	2,2 sec	1,5 sec	1,4 sec	1,3 sec	1,3 sec
NAI	10 sec	7 sec	4,1 sec	2 sec	1,9 sec	1.8 sec	1,8 sec

Πίνακας μετρήσεων 1

iPhone 12

Resolving time	1 (admin)	2	4	6	10	15	20
OXI	5,2 sec	4,3 sec	2,1 sec	1,2 sec	1,2 sec	1,2 sec	1,2 sec
NAI	9,5 sec	6,8 sec	4 sec	1,9 sec	1,7 sec	1.7 sec	1,7 sec

Πίνακας μετρήσεων 2

Συμπέρασμα

Από τις παραπάνω μετρήσεις μπορούν να βγουν αρκετά πορίσματα. Αρχικά μπορεί να γίνει αντιληπτό, ότι και στις 2 περιπτώσεις, καθώς ο χρήστης προσθέτει Cloud Anchors, δηλαδή κάνει χρήση του μοντέλου, οι χρόνοι ελαττώνονται αρκετά. Μέχρι 6 Cloud Anchors σε ένα Checkpoint υπάρχει συνεχόμενη και υπολογίσιμη μείωση χρόνου, ενώ στη συνέχεια όχι και τόσο. Αυτό είναι αναμενόμενο γιατί στην αρχή ο χρήστης με την προσθήκη Cloud Anchors, προσπαθεί να καλύψει όλες τις πιθανές γωνίες λήψης και να δημιουργήσει όσο το δυνατόν καλύτερα Feature Map στο συγκεκριμένο καρέ που είναι το Checkpoint. Μετά από ένα σημείο δεν βελτιώνεται η εφαρμογή, μιας και η αντιστοίχιση με ένα Feature Map γίνεται πιο πιθανή με τα ήδη υπάρχοντα. Επιπρόσθετα μπορεί να γίνει σύγκριση με το αν υπήρχαν ή όχι αλλαγές στο χώρο. Στην περίπτωση που δεν είχαμε αλλαγές στο χώρο, οι χρόνοι για το resolve είναι αισθητά μικρότεροι από όταν είχαμε αλλαγές, ανεξάρτητος αριθμού Cloud Anchors στο

Checkpoint. Τέλος παρατηρούνται μικρές διαφορές στους χρόνους μεταξύ των 2 μοντέλων κινητών, αλλά αυτό μπορεί να θεωρηθεί αμελητέο μιας και εξαρτάται από τα χαρακτηριστικά της κάμερας του κάθε μοντέλου.

5.2 Αξιολόγηση

Προκύπτουν ορισμένα συμπεράσματα αναφορικά με τη χρησιμότητα που το σύστημα αυτό μπορεί να έχει για τα άτομα που βρίσκονται καθημερινά στο ΕΜΠ, καθώς και για την αξία των ιδιαίτερων τεχνολογικών χαρακτηριστικών του.

Αρχικά σε ότι αφορά τη λειτουργία της εφαρμογής, μπορεί να θεωρηθεί απαραίτητη στα πλαίσια μιας χαρτογράφηση για όλους τους εξωτερικούς και εσωτερικούς χώρου του campus του πολυτεχνείου. Επίσης, είναι από όλους αποδεκτό ότι στον 21ο αιώνα και με τη εξέλιξης της τεχνολογίας, οι ρυθμοί ζωής αυξάνονται καθημερινά, ο άνθρωπος αναζητά την απλότητα σε αυτά που θέλει να κάνει και συνεχώς ψάχνει και δημιουργεί τρόπους να επιταχύνει της διαδικασίες του. Όλα αυτά μπόρεσαν να προωθηθούν μέσα από καινοτόμα και σύγχρονη εφαρμογή που δημιουργήθηκε.

Επιπρόσθετα όμως, αξίζει να σημειωθούν κάποια συμπεράσματα που βγήκαν από τη χρήση των παραπάνω τεχνολογικών εργαλείων:

- Έγιναν πολλές δοκιμές για την τοποθέτηση των Geospatial Anchors. Τα συγκεκριμένα τοποθετούνται στον εξωτερικό χώρο με βάση τις γεωγραφικές συντεταγμένες, όμως σε αυτές ήταν αρκετά δύσκολο να υπολογιστεί το ύψος που έπρεπε να τοποθετηθεί το Anchor. Το ύψος μετράει πάντα από τη επιφάνεια της γης και αυτό είχε σαν αποτέλεσμα να μην τοποθετείται σωστά ένα Anchor όταν το ήθελες πάνω σε μια γέφυρα, όπου το ύψος δεν είναι 0. Μερικές φορές όμως και ανάλογα την περίπτωση, ακόμα και σε σημεία που ήταν υπερυψωμένα, το AR τα λάμβανε σαν 0 υψόμετρο οπότε το Anchor εμφανιζόταν από κάτω τους. Έτσι χρειάστηκαν πολλές δοκιμές για να πετύχουμε σωστό αποτέλεσμα και να αντιληφθούμε την ιδιαιτερότητα των Geospatial Anchors.

- Επίσης, κατά την διάρκεια της κατασκευής της εφαρμογής, χρειάστηκε να πραγματοποιηθουν πολλές δοκιμές για την καλύτερη τοποθέτηση των εσωτερικών Cloud Anchors. Δημιουργήθηκαν δύο παράλληλες υλοποιήσεις, στη μια όταν τοποθετούσε ο admin ένα Cloud Anchor σε ένα Checkpoint , κατευθείαν το έκανε upload μέσο του ARCore API, ενώ στην δεύτερη τοποθετούσε όλα τα Cloud Anchors στα Checkpoints για ένα μονοπάτι και τα έκανε upload συνολικά. Η κάθε μια υλοποίηση είχε τα δικά της θετικά και αρνητικά, και έπρεπε να γίνουν πολλές δοκιμές για να καταλήξουμε στην καλύτερη περίπτωση. Αρχικά ένα θετικό που είχε η πρώτη, ήταν ότι όταν κάναμε μεμονωμένα το uploading μπορούσαμε να κάνουμε καλύτερη καταγραφή του χώρου για το συγκεκριμένο Cloud Anchor και να έχει πολύ καλό Feature Map. Όμως σε αυτή την περίπτωση, υπήρχαν φορές που αργούσε πολύ το resolving, ειδικά αν υπήρχαν αλλαγές στο χώρο. Στην δεύτερη περίπτωση το θετικό ήταν, ότι όταν «ανεβαίνουν» τα Cloud Anchors όλα μαζί υπήρχε μια σύνδεση μεταξύ τους και αυτό βοηθούσε το resolving να τα εμφανίζει πιο γρήγορα, σαν αλυσίδα.

Βέβαια και σε αυτή την περίπτωση υπήρχαν, ίσως μεγαλύτερες καθυστερήσεις, γιατί όταν κάναμε την καταγραφή εστιάζαμε στο γενικό μονοπάτι και όχι σε κάθε ένα ξεχωριστά. Οπότε, ειδικά το πρώτο Cloud Anchor, αργούσε πολύ να εμφανιστεί. Εμείς διαλέξαμε την πρώτη περίπτωση, μιας και την κρίναμε πιο αποδοτική, ειδικά με την βελτίωση που παρατηρήσαμε με την χρήση του μοντέλου Μηχανικής Μάθησης.

- Πραγματοποιήθηκαν δοκιμές κατα τη χρήση των Μεθόδων Οπτικοποίησης. Τις δημιουργήσαμε για να ενώνονται τα Cloud Anchors μεταξύ τους, που τοποθετούνται στα κύρια Checkpoints. Εκεί θέλαμε να τοποθετήσουμε Nodes με glTF (πράσινες κουκίδες). Η μέθοδος έπρεπε αυτόματα να βρίσκει ανά δυο τις θέσεις των Cloud Anchors, έτσι ώστε να όριζε σωστά την αρχή και το τέλος των Nodes. Μετέπειτα έπρεπε να τα τοποθετήσει στις σωστές αποστάσεις μεταξύ τους, με σωστό μέγεθος και σε ευθεία γραμμή. Έτσι, για να πετύχουμε αυτό το αποτέλεσμα, κάναμε πολλές δοκιμές και αλλαγές στις αντίστοιχες παραμέτρους.

- Τέλος, ένα πολύ σημαντικό συμπέρασμα έχει να κάνει με το μοντέλο Incremental Learning και τη μετρική που είχαμε ορίσει για την αξιολόγηση του. Πιο συγκεκριμένα η συγκεκριμένη μετρική μας έδωσε την δυνατότητα να γνωρίζουμε ποιο Cloud Anchor τοποθετείται κάθε φορά σε ένα Checkpoint. Έτσι κρίνουμε, αν μια καταγραφή που κάναμε για να τοποθετήσουμε ένα Cloud Anchor είναι καλή, υπό ποια γωνία λήψης έχει γίνει, πώς επηρεάζουν οι αλλαγές στο χώρο και πόσο χρόνο χρειάζεται το ARCore API όταν γίνεται το resolving για να αντιστοίχιση τα Feature Map και να εμφανίσει το Cloud Anchor. Τέλος με αυτήν την διαδικασία μπορούσαμε να διαγράφουμε όσο Cloud Anchor δεν χρησιμοποιούνταν συχνά δηλαδή δεν είχε γίνει καλή καταγραφή του χώρου για να μην υπερφορτώνονται και οι βάσεις που χρησιμοποιήθηκαν.

5.3 Προτάσεις για μελλοντικές επεκτάσεις

Παρακάτω παρουσιάζονται κάποιες προτάσεις και βελτιώσεις που μπορούν να υλοποιηθούν σε μεταγενέστερο στάδιο, με σκοπό η εφαρμογή ΕΜΠ Demo App να γίνει πιο εύχρηστη και αποτελεσματική:

- Θα πρέπει οι δυο παράλληλες εφαρμογές admin και user να χωριστούν σε ξεχωριστά UI και να είναι τελείως αυτόνομες.
- Έχει αναφερθεί ότι η εφαρμογή ανήκει στη οικογένεια των Cross Platform, όμως πρέπει να γίνουν κάποιες αλλαγές στα permissions που αφορούν την σύνδεση στην βάση Firestore αλλά και στο ARCore API για να γίνει λειτουργική στο λογισμικό Android.
- Επίσης μπορούν να γίνουν βελτιώσεις στο Incremental Learning, έτσι ώστε να μην γίνεται αντιληπτό από τους users ότι συνεισφέρουν στην εξέλιξη της εφαρμογής. Αυτό μπορεί να γίνει με την αυτόματη τοποθέτηση Cloud Anchor χωρίς να χρειάζεται κάποιο άγγιγμα στο 3D μοντέλο.
- Είναι δυνατή η εξέταση προτεινόμενων από τη βιβλιογραφία αλγορίθμων Incremental Learning σχετικά με τα πιθανά αποτελέσματα που θα έχουν στο συγκεκριμένο παράδειγμα σεναρίου χρήσης.

- Μπορούν επίσης να προστεθούν και άλλοι στόχοι, εξωτερικοί και εσωτερικοί και να δημιουργηθούν ομάδες φοιτητών που θα αναλάβουν τη χάραξη και την χαρτογράφηση όλων των αναγκαίων διαδρομών στο ΕΜΠ.
- Τέλος η εφαρμογή είχε και ένα πιο γενικό ερευνητικό ρόλο για το αν μπορούν όλες αυτές οι τεχνολογίες να συνυπάρξουν μαζί. Γιατί με γνώμονα την συγκεκριμένη υλοποίηση μπορούν να δημιουργηθούν νέες που αφορούν χώρους μουσείων, ιατρικά κέντρα, πάρκα και γενικά τοποθεσίες που περιλαμβάνουν εσωτερικές και εξωτερικές τοποθεσίες με χαοτικές διαδρομές και συνεχόμενες αλλαγές στον περιβάλλοντα χώρο τους.

Κεφάλαιο 6: Κατακλείδα

Σκοπός της παρούσας διπλωματικής ήταν να μελετηθεί η ανάπτυξη μιας αυτόνομης εφαρμογής η οποία θα έχει την δυνατότητα να καθοδηγεί τους φοιτητές της Πολυτεχνειούπολης σε εξωτερικούς και εσωτερικούς χώρους, συνδυάζοντας τις τεχνολογίες της Επαυξημένης Πραγματικότητας με τη Μηχανική Μάθηση.

Αρχικά το Πολυτεχνείο θεωρείται από πολλούς ένας χαοτικός χώρος, με αμέτρητα κτίρια, δρόμους, γραφεία και αίθουσες. Ένας νέος επισκέπτης, αλλά ακόμα και άνθρωποι που βρίσκονται χρόνια στο campus, δυσκολεύονται στην προσπάθειά τους να βρουν τον προορισμό τους λόγω της πολυπλοκότητας της περιοχής και της έλλειψης σημάνσεων. Έτσι, δημιουργήθηκε η ανάγκη για τη μελέτη και υλοποίηση μιας εφαρμογής η οποία θα αποτελεί τον φορητό πλοηγό του επισκέπτη. Βασικά χαρακτηριστικά της εφαρμογής είναι η δυνατότητα εγκατάστασής της σε commodity devices, δηλαδή σε συσκευές που ο χρήστης ήδη έχει, αλλά και η εξάλειψη κόστους και διαδικασιών χάρη στην μηδαμινή ανάγκη για εγκαταστάσεις εξοπλισμού στον χώρο.

Στο πλαίσιο, λοιπόν, της παρούσας διπλωματικής επιλέξαμε να επικεντρωθούμε στο πώς θα υποστηρίξουμε την καθημερινότητα των ατόμων του Πολυτεχνείου αλλά να βρούμε έξυπνους και καινοτόμους τρόπους για να το πετύχουμε. Παρακάτω αναφέρονται τα βασικά στοιχεία των όσων επιτεύχθηκαν από την παρούσα μελέτη και υλοποίηση:

- Υλοποίηση Cross Platform εφαρμογής (Android/iOS).
- Δημιουργία εξωτερικής διαδρομής με τη χρήση Geospatial Anchors.
- Ανίχνευση ακριβής τοποθεσίας του χρήστη σε εξωτερικό χώρο.
- Δημιουργία εσωτερικής διαδρομής με τη χρήση Cloud Anchors.
- Ανίχνευση ακριβής τοποθεσίας του χρήστη σε εσωτερικό χώρο.
- Μοντέλο Incremental Learning για τη βελτίωση της εσωτερικής πλοήγησης.
- Ερευνητική χρήση εφαρμογής για την τοποθεσία Cloud Anchor μέσω του μοντέλου.
- Συνδυασμός Geospatial Anchors και Cloud Anchors σε μια ενιαία εφαρμογή.
- Γρήγορη και βέλτιστη καθοδήγηση στον τελικό προορισμό.

Η κεντρική ερευνητική συμβολή της παρούσας διπλωματικής είναι πρωτίστως η συνδυαστική χρήση αλγορίθμων Μηχανικής Μάθησης, και δη η Αυξητική Μάθηση, με την Επαυξημένη Πραγματικότητα. Οι καινοτόμες τεχνικές που παρουσιάζονται, έχουν υλοποιηθεί ως ένα ανώτερο επίπεδο Τεχνητής Νοημοσύνης το οποίο βελτιστοποιεί την Επαυξημένη Πραγματικότητα στην αναγνώριση εσωτερικού χώρου. Επιπλέον, υλοποιήθηκε πρωτότυπη

εφαρμογή η οποία αποδεικνύει στην πράξη τους ανωτέρω ισχυρισμούς αλλά προτείνει και τη λύση του πρακτικού προβλήματος της Πολυτεχνειούπολης, δηλαδή της πλοήγησης συνδυαστικά σε εσωτερικό και εξωτερικό χώρο χωρίς τοπικές παρεμβάσεις και με τις ελάχιστες ανάγκες σε υλικό και λογισμικό.

Βιβλιογραφία

[1] D. R. Berryman, “Augmented Reality: A Review,” *Medical Reference Services Quarterly*, vol. 31, no. 2. pp. 212–218, Apr. 2012. doi: 10.1080/02763869.2012.670604.

[2] J. Carmigniani, B. Furht, M. Anisetti, P. Ceravolo, E. Damiani, and M. Ivkovic, “Augmented reality technologies, systems and applications,” *Multimed Tools Appl*, vol. 51, no. 1, pp. 341–377, Jan. 2011, doi: 10.1007/s11042-010-0660-6.

[3] P. Daponte, L. De Vito, F. Picariello, and M. Riccio, “State of the art and future developments of the Augmented Reality for measurement applications,” *Measurement: Journal of the International Measurement Confederation*, vol. 57. Elsevier B.V., pp. 53–70, 2014. doi: 10.1016/j.measurement.2014.07.009.

[4] S. Das, A. Dey, A. Pal, and N. Roy, “Applications of Artificial Intelligence in Machine Learning: Review and Prospect,” *Int J Comput Appl*, vol. 115, no. 9, pp. 31–41, 2015, doi: 10.5120/20182-2402.

[5] M. van Steen and A. S. Tanenbaum, “A brief introduction to distributed systems,” *Computing*, vol. 98, no. 10, pp. 967–1009, Oct. 2016, doi: 10.1007/s00607-016-0508-7.

[6] Koulouris Dionysios, Andreas Menychta, and Ilias Maglogiannis, “‘Augmented Reality for Indoor Localization and Navigation: The Case of UNIPI AR Experience.,’” in *International Conference on Computer Analysis of Images and Patterns.*

[7] “Cross-Platform_Development_Software_that_Lasts”.

[8] C. Khawas and P. Shah, “Application of Firebase in Android App Development-A Study,” *Int J Comput Appl*, vol. 179, no. 46, pp. 49–53, Jun. 2018, doi: 10.5120/ijca2018917200.

[9] X. Gu, H. Zhang, D. Zhang, and S. Kim, “Deep API Learning,” in *Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, Association for Computing Machinery, Nov. 2016, pp. 631–642. doi: 10.1145/2950290.2950334.

[10] M. Kumar, “Serverless Architectures Review, Future Trend and the Solutions to Open Problems,” *American Journal of Software Engineering*, vol. 6, no. 1, pp. 1–10, Mar. 2019, doi: 10.12691/ajse-6-1-1.

[11] M. Badurowicz, “MVC architectural pattern in mobile web applications.”

[12] “Getx package.” [Online]. Available: https://pub.dev/packages/get

[13] “ar_flutter_plugin.” [Online]. Available: https://pub.dev/packages/ar_flutter_plugin

[14] “cloud_firestore.” [Online]. Available: https://pub.dev/packages/cloud_firestore

[15] I. B. Kerthyayana Manuaba, “Mobile based Augmented Reality Application Prototype for Remote Collaboration Scenario Using ARCore Cloud Anchor,” in *Procedia Computer Science*, Elsevier B.V., 2021, pp. 289–296. doi: 10.1016/j.procs.2021.01.008.