ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάπτυξη και Σχεδίαση Σύστημα Διαχείρισης Αποθεμάτων Σε Περιβάλλον Android

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΣΟΥΛΕΪΜΑΝ ΑΪΚΟΥΤ

(ΑΕΜ:2309)

Επιβλέπων : **Δημήτριος Ι. Βέργαδος**

Επίκουρος Καθηγητής

Καστοριά Δεκέμβριος - 2022

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάπτυξη και Σχεδίαση Σύστημα Διαχείρισης Αποθεμάτων Σε Περιβάλλον Android

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΣΟΥΛΕΪΜΑΝ ΑΪΚΟΥΤ

(ΑΕΜ:2309)

Επιβλέπων : **Δημήτριος Ι. Βέργαδος**

Επίκουρος Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 05/12/2022

.....	.....	.....
Δημήτριος Ι. Βέργαδος	Μιχαήλ Δόσης	Νίκος Δημόκας
Επίκουρος Καθηγητής	Καθηγητής, Πρόεδρος του τμήματος	Επίκουρος Καθηγητής

Καστοριά Δεκέμβριος - 2022

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου Δημήτριος Ι. Βέργαδος (Επίκουρος Καθηγητής), για την καθοδήγηση που μου προσέφερε και το χρόνο που διέθεσε δίνοντάς μου χρήσιμες συμβουλές και οδηγίες για την ολοκλήρωση της πτυχιακής μου εργασίας. Στο ίδιο πλαίσιο ευγνωμοσύνης, θα ήθελα να ευχαριστήσω όλους τους καθηγητές του Τμήματος Πληροφορικής για τη συμβολή τους στην επιστημονική και τεχνολογική μου συγκρότηση στα χρόνια της φοίτησής μου στο Τμήμα.

Περίληψη

Στις μέρες με την εξέλιξη της τεχνολογίας, η επιχείρησης απευθύνονται όλο και περισσότερο στα σύγχρονα συστήματα διαχείρισης αποθεμάτων WMS ή στα εμπορικά προγράμματα (ERP). Έχοντας στην αντίληψη αυτή την απαίτηση, υλοποιήθηκε η παρακάτω εργασία. Στην παρακάτω εργασία θα υλοποιήσουμε και θα αναλύσουμε ένα σύγχρονο σύστημα διαχείρισης αποθήκης χρησιμοποιώντας όλες τις δυνατότητες της σύγχρονης τεχνολογίας. Αποφεύγοντας την πολυπλοκότητα, στοχοποιούμε να έχουμε καλύτερο αποτέλεσμα από ένα παλιό σύστημα όπου χρειάζεται περισσότερο χρόνο για να υλοποιηθεί και δύσκολη στην χρήση προς τον χρήστη. Η Εφαρμογή υλοποιήθηκε με βάση της απαιτήσεις που θα μπορούσε να έχει ένα μικρό – μεσαίο κατάστημα. Στην δικιά μας περίπτωση είναι το Mini Market Άριστον.

Λέξεις Κλειδιά: Android, Retrofit Client, Rest Api, Spring Boot, Βάση δεδομένων, Heroku, HTTP, GET, PUT, UPDATE, DELETE, Java, WMS, ERP, Eclipse,Plug-in, Postman, Cloud Computing, Heroku Dynos, Android Studio, Mysql, Kotlin, C++, Retrofit,

Abstract

Nowadays, with the evolution of technology, companies are increasingly turning to modern inventory management systems or ERP programs. With this requirement in mind, the following work was carried out. In the following paper we will implement and analyze a modern warehouse management system using all the capabilities of modern technology. By avoiding complexity, we aim to get a better result than a legacy system where it takes longer to implement and is difficult to use for the user. The Application was implemented based on the requirements that a small - medium store could have. In our case it is Mini Market Ariston.

Key Words: Android, Retrofit Client, Rest Api, Spring Boot, Βάση δεδομένων, Heroku, HTTP, GET, PUT, UPDATE, DELETE, Java, WMS, ERP, Eclipse,Plug-in, Postman, Cloud Computing, Heroku Dynos, Android Studio, Mysql, Kotlin, C++, Retrofit,

Πίνακας Περιεχομένων

Λίστα Εικόνων

Λίστα Διαγραμμάτων

Λίστα Πινάκων

Εισαγωγή

Ο Σκοπός της εργασίας είναι δημιουργία μια σύγχρονη σύστημα διαχείρισης αποθήκης για να καλύψει της ανάγκες του καταστήματος «Άριστον». Η εργασία αναπτύχθηκε σύμφωνα με τις ανάγκες του καταστήματος και ο στόχος ήταν η εφαρμογή να είναι απλή στη χρήση και γρήγορη. Η αρχική ιδέα ήταν να το αναπτύξουμε σε μια εφαρμογή των Windows. Αυτό όμως αποτελούσε ένα μεγάλο πρόβλημα, διότι ο χρήστης έπρεπε να χρησιμοποιεί κάθε φορά έναν υπολογιστή με συνδεδεμένο σαρωτή και να τον έχει πάντα δίπλα του. Η ανάπτυξη ενός τέτοιου συστήματος , μας δίνει τη δυνατότητα να συνδεθούμε από όπου κι αν βρισκόμαστε ανεξάρτητα από οποιονδήποτε υπολογιστή πάρα μόνο έχοντας πρόσβαση στο δίκτυο και το κινητό τηλέφωνο μας δίπλα.

Η εργασία αποτελείτε από τέσσερα βασικά κομμάτια.

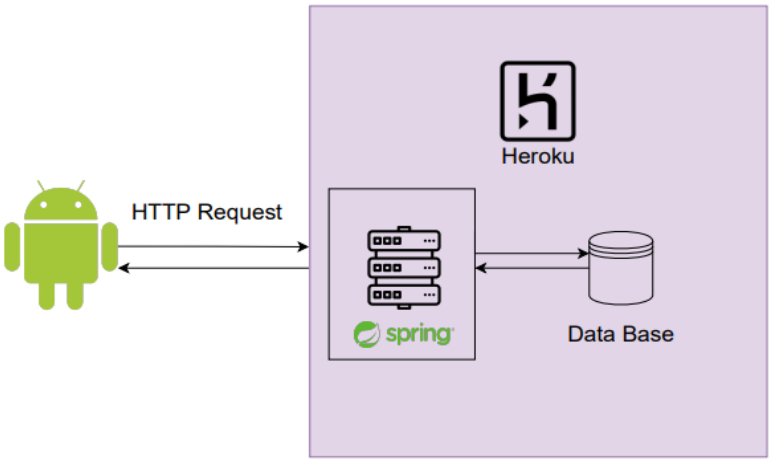
1) Η **Εφαρμογή Android** – Retrofit Client

2) **Rest Api** εξυπηρετής με επέκταση **Spring Boot**,

3) **Βάση Δεδομένων**

4) **Heroku** όπου εκεί φιλοξενείτε και το Server μας

Η εφαρμογή Android στέλνει HTTP αιτήματα μέσω Retrofit όπως GET,PUT,UPDATE και Delete, εννοώ ο ενδιάμεσος εξυπηρετείς ανάλογα με τα αιτήματα που στέλνει η εφαρμογή μας επιστρέφει τις τιμές από την Βάση δεδομένων όπου ο εξυπηρετείς και η βάση δεδομένων φιλοξενείτε στην πλατφόρμα Heroku. Οι τιμές που επιστρέφονται αποθηκεύονται σε μια λίστα στην εφαρμογή και δεν χρειάζεται να υπάρχει κάποια άλλη βάση δεδομένων στην εφαρμογή. Κάθε φόρα ο χρήστης που στέλνει αιτήματα η λίστα ανανεώνεται. Χρησιμοποιώντας μια τέτοια αρχιτεκτονική, μας δίνει την δυνατότητα να αναπτύξουμε όλα τα παραπάνω μόνο στην γλώσσα προγραμματισμού Java.



Διάγραμμα 1. 1 Περίληψη της συνδεσμολογίας

1. Σύστημα Διαχείρισης Αποθήκης (WMS)

1.1 Σύστημα διαχείρισης αποθήκης

Ένα σύστημα διαχείρισης αποθήκης (WMS) είναι μια λύση λογισμικού που προσφέρει ορατότητα σε ολόκληρο το απόθεμα μιας επιχείρησης και διαχειρίζεται τις λειτουργίες εκπλήρωσης της αλυσίδας εφοδιασμού από το κέντρο διανομής έως το ράφι του καταστήματος. Οι λύσεις διαχείρισης αποθήκης (WMS) επιτρέπουν επιπλέον στις εταιρείες να μεγιστοποιήσουν τη χρήση του εργατικού δυναμικού και του χώρου και τις επενδύσεις τους σε εξοπλισμό, συντονίζοντας και βελτιστοποιώντας τη χρήση των πόρων και τις ροές υλικών. Συγκεκριμένα, τα συστήματα WMS έχουν σχεδιαστεί για να υποστηρίζουν τις ανάγκες ολόκληρης της παγκόσμιας αλυσίδας εφοδιασμού, συμπεριλαμβανομένων των επιχειρήσεων διανομής, παραγωγής, έντασης περιουσιακών στοιχείων και παροχής υπηρεσιών

Εικόνα 1.1 WMS WORK FLOW

Στη σημερινή δυναμική, πολυκαναλική οικονομία της εκπλήρωσης, οι συνδεδεμένοι καταναλωτές θέλουν να αγοράζουν οπουδήποτε, να εκπληρώνουν οπουδήποτε και να επιστρέφουν οπουδήποτε. Προκειμένου να είναι σε θέση να ανταποκριθούν σε αυτή την ανάγκη, οι επιχειρήσεις χρειάζονται τη δυνατότητα να ανταποκρίνονται γρήγορα με λογισμικό διαχείρισης αποθήκης που βελτιστοποιεί τις δυνατότητες εκπλήρωσης. Το κορυφαίο, βασισμένο στο cloud, σύστημα διαχείρισης αποθηκών που διαθέτουμε, σας προετοιμάζει για την αλυσίδα εφοδιασμού του αύριο, σήμερα. Το WMS Cloud επεκτείνει τις αλυσίδες εφοδιασμού για να ευθυγραμμίσει τις υπηρεσίες διαχείρισης αποθεμάτων και εκπλήρωσης με τις σύγχρονες μεθόδους αγορών και προσφέρει ορατότητα σε

πραγματικό χρόνο σε ολόκληρο το απόθεμα - διαθέσιμη μέσω έξυπνου τηλεφώνου και προγράμματος περιήγησης - με μόνη προϋπόθεση την πρόσβαση στο Διαδίκτυο.

1.2 Πώς ένα σύστημα διαχείρισης αποθήκης βελτιστοποιεί την παραγωγικότητα

Όταν ένας διανομέας αποφασίζει να εφαρμόσει ένα σύστημα διαχείρισης αποθήκης, βελτιστοποιεί την παραγωγικότητα της αποθήκης και τα λογιστικά. Οι εργαζόμενοι στην αποθήκη επωφελούνται από τα λεπτομερή αρχεία καταγραφής δραστηριοτήτων, επειδή γνωρίζουν ακριβώς ποιο είναι το επόμενο βήμα στη διαδικασία παραγγελίας και μπορούν να παραλαμβάνουν, να συσκευάζουν και να αποστέλλουν τα αποθέματα με ευκολία. Ο έλεγχος των αποθεμάτων είναι απλός επειδή κάθε στοιχείο στην αποθήκη καταγράφεται σε πραγματικό χρόνο, ανά πάσα στιγμή. Οι πελάτες μπορούν να εμπιστεύονται ότι οι παραγγελίες τους θα εκτελούνται με ακρίβεια σε σύντομο χρονικό διάστημα και ότι οι πελάτες τους θα είναι ικανοποιημένοι με αυτό που λαμβάνουν.

1.3 Πλεονεκτήματα ενός σύγχρονου, εφαρμογής διαχείρισης αποθήκης

1.3.1 Ακριβής αρχεία απογραφής

Όταν ένας πελάτης σας εμπιστεύεται να εκτελείτε παραγγελίες για λογαριασμό του, η ακρίβεια της απογραφής είναι απαραίτητη. Ένα WMS παρακολουθεί ακριβώς πού είναι αποθηκευμένα τα αποθέματα, πόσες μονάδες είναι διαθέσιμες και ποια είδη πρέπει να αποσταλούν. Ο κύκλος ζωής μιας παραγγελίας συντομεύεται σημαντικά όταν αυτές οι λεπτομέρειες είναι σαφείς. Η χρήση ενός WMS μειώνει τα ανθρώπινα λάθη και επιτρέπει στους διανομείς να διαχειρίζονται προληπτικά τα αποθεμάτά τους με απόλυτη ακρίβεια.

1.3.2 Βελτιστοποίηση Λογιστικής

Από την παραλαβή των αποθεμάτων έως την έξοδό τους από την αλυσίδα εφοδιασμού, υπάρχουν αμέτρητα μικρά πράγματα που μπορούν να πάνε στραβά. Η χρήση ενός WMS μειώνει τα σφάλματα στη διαχείριση της εφοδιαστικής, καθώς διατηρεί αρχείο με κάθε βήμα στον κύκλο ζωής των παραγγελθέντων αποθεμάτων. Η διαχείριση της αλυσίδας εφοδιασμού γίνεται σημαντικά λιγότερο περίπλοκη όταν υπάρχει ένα WMS για την τήρηση λεπτομερών αρχείων μιας παραγγελίας. Αυτό οδηγεί σε μεγαλύτερη ακρίβεια και έλεγχο των αποθεμάτων, γεγονός που με τη σειρά του οδηγεί σε μεγαλύτερη ικανοποίηση των πελατών όταν παραλαμβάνεται μια παραγγελία.

1.3.3 Ενημερώσεις σε πραγματικό χρόνο

Ένα από τα βασικά στοιχεία για τη διασφάλιση της ακρίβειας των προϊόντων είναι η ενημέρωση σε πραγματικό χρόνο. Ένα WMS επιτρέπει στους διανομείς και τους πελάτες να γνωρίζουν τι υπάρχει σε απόθεμα εκείνη ακριβώς τη στιγμή, ώστε να γνωρίζουν ακριβώς τι είναι διαθέσιμο προς πώληση ή αποστολή χωρίς καθυστέρηση. Με τον τρόπο αυτό αποφεύγεται η παραγγελία ειδών χωρίς απόθεμα λόγω καθυστέρησης στον εντοπισμό των αποθεμάτων και τα δημοφιλή είδη μπορούν να ανακατασκευαστούν πριν

μειωθούν τα επίπεδα αποθεμάτων. Επιπλέον, το WMS διατηρεί πλήρη αρχεία κάθε δραστηριότητας στη διαδικασία παραγγελίας, ώστε οι διανομείς και οι εργαζόμενοι στην αποθήκη να έχουν μια τρέχουσα εικόνα για το πού βρίσκονται οι παραγγελίες.

1.3.4 Εύκολη πρόσβαση σε δεδομένα

Στο σημερινό διαρκώς μεταβαλλόμενο εργατικό δυναμικό, οι διαχειριστές αποθηκών μπορεί να μην είναι σε θέση να ελέγχουν αυτοπροσώπως το απόθεμα ενός προϊόντος ή να ελέγχουν χειροκίνητα για να δουν τι χρειάζεται αναπλήρωση. Λόγω των άφθονων διαθέσιμων επιλογών λογισμικού WMS που βασίζονται στο cloud, οι διανομείς και οι πελάτες μπορούν να παρακολουθούν τα αποθηκευμένα αποθέματα οπουδήποτε και ανά πάσα στιγμή. Η πρόσβαση σε εικονικά δεδομένα είναι βολική και εξοικονομεί χρόνο, τόσο για την εταιρεία αποθήκευσης όσο και για τους πελάτες που εξυπηρετεί.

1.3.5 Καλύτερη εξυπηρέτηση πελατών

Όταν ένας πελάτης παραγγέλνει εμπορεύματα, περιμένει να λάβει ακριβώς αυτό που παρήγγειλε σε συγκεκριμένο χρονικό διάστημα. Όταν ένα WMS χρησιμοποιείται για τη βελτιστοποίηση της εκτέλεσης των παραγγελιών, οι προσδοκίες αυτές ικανοποιούνται πολύ συχνότερα. Όταν η κατάσταση μιας παραγγελίας παρακολουθείται συνεχώς και με συνέπεια, οι εργαζόμενοι μπορούν να διασφαλίσουν ότι τηρούν τις προθεσμίες και ότι κάθε παραγγελία αποστέλλεται εγκαίρως. Οι διαδικασίες αποθήκευσης είναι περίπλοκες, επομένως η χρήση ενός WMS για τη βελτιστοποίηση της εφοδιαστικής είναι απαραίτητη για να διασφαλιστεί ότι οι απαιτήσεις των πελατών ικανοποιούνται και ότι είναι ικανοποιημένοι με το τελικό προϊόν.

1.4 Τύποι συστημάτων διαχείρισης αποθήκης

1.4.1 Ολοκληρωμένη WMS

Υπάρχουν δύο τύποι λογισμικού συστήματος διαχείρισης αποθήκης που μπορείτε να χρησιμοποιήσετε για να παρακολουθείτε όλες τις εισερχόμενες και εξερχόμενες λειτουργίες των προϊόντων σας. Ένα ολοκληρωμένο WMS είναι συνήθως ένα πρόσθετο από τον υπάρχοντα πάροχο υπηρεσιών διαχείρισης επιχειρησιακών πόρων (ERP). Τα συστήματα ERP διαχειρίζονται την τιμολόγηση, τη λογιστική και την παρακολούθηση των αποθεμάτων. Το σύστημα διαχείρισης αποθήκης λαμβάνει τις παραγγελίες και κατευθύνει τη διαδικασία συλλογής παραγγελιών, την απογραφή, την παραλαβή και την αποστολή των προϊόντων. Όταν όλα μπορούν να ενσωματωθούν σε ένα σύστημα, είναι πολύ πιο εύκολο να παρακολουθείτε σε ποιες παραγγελίες είναι καλύτερο να επενδύσετε χρήματα. Εάν έχετε ένα προϊόν που πουλάει πολύ καλά αλλά έχει χαμηλό περιθώριο κέρδους, μπορείτε να επιλέξετε να επαναφέρετε ένα προϊόν με υψηλότερο κέρδος και ελαφρώς χαμηλότερες πωλήσεις. Μπορείτε να παρακολουθείτε όλες αυτές τις οικονομικές αναλύσεις με ένα ολοκληρωμένο σύστημα διαχείρισης αποθήκης.

1.4.2 Αυτόνομο WMS

Ένα αυτόνομο σύστημα διαχείρισης αποθήκης είναι ένα λογισμικό πλούσιο σε χαρακτηριστικά που εξυπηρετεί κυρίως τη λειτουργία της διαχείρισης αποθήκης. Ως εκ τούτου, ενδέχεται να έχει περιορισμένη λειτουργικότητα για άλλες πτυχές της επιχείρησής σας, όπως η απογραφή ή η λογιστική. Δεδομένου ότι είναι προσαρμοσμένο στη διαχείριση αποθηκών, αυτός ο τύπος WMS μπορεί να διαθέτει προηγμένες λειτουργίες αναφοράς που θα σας βοηθήσουν να βελτιώσετε την αποθήκη σας.

1.4.3 On-Premises και. Cloud WMS

Με βασικούς όρους, ένα on-premises WMS είναι ένα σύστημα στο οποίο είστε υπεύθυνοι για τη φιλοξενία και τη συντήρηση τόσο του υλικού όσο και του λογισμικού που σχετίζονται με το σύστημά σας. Ενώ αυτό σας δίνει πλήρη έλεγχο σε πράγματα όπως ο χρόνος διαθεσιμότητας και η ασφάλεια, συνοδεύεται επίσης από ένα μεγάλο αρχικό κόστος, επειδή είστε υπεύθυνοι για όλα τα εξαρτήματα. Θα πρέπει επίσης να συντηρείτε τακτικά το WMS σας. Παρόλο που οι πολύ μικρές επιχειρήσεις μπορούν να χρησιμοποιούν άνετα το WMS στις εγκαταστάσεις τους, η διαχείριση των πάντων από εσάς μπορεί να είναι πονοκέφαλος. Εναλλακτικά, τα συστήματα WMS που βασίζονται στο cloud χρεώνονται συνήθως με συνδρομή, αλλά φιλοξενούνται σε απομακρυσμένο διακομιστή. Πράγματα όπως οι διορθώσεις σφαλμάτων και οι ενημερώσεις λογισμικού διεκπεραιώνονται από τον προμηθευτή, και συνήθως λαμβάνετε ένα εγγυημένο επίπεδο διαθεσιμότητας της υπηρεσίας όταν εγγράφεστε. Όσο περισσότερο αναπτύσσεται μια επιχείρηση διαδικτυακής λιανικής πώλησης, τόσο περισσότερο πρέπει να αξιολογείτε τον πιο αποτελεσματικό τρόπο διαχείρισης όλων των προϊόντων. Οι πελάτες έχουν υψηλές προσδοκίες όσον αφορά την αποστολή και ακόμη και τη συσκευασία των παραγγελιών στις μέρες μας, οπότε είναι σημαντικό να εξαλειφθεί η σύγχυση στην αποθήκη. Η εξοικονόμηση όλου του χρόνου και των πόρων που σχετίζονται με τα συστήματα που βρίσκονται στις εγκαταστάσεις μπορεί να είναι ένας καλός τρόπος για να γίνει ακριβώς αυτό.

2. Java

2.1 Java

Η Java είναι μια γλώσσα προγραμματισμού και μια υπολογιστική πλατφόρμα που κυκλοφόρησε για πρώτη φορά από τη Sun Microsystems το 1995. Έχει εξελιχθεί από τις ταπεινές απαρχές της και τροφοδοτεί ένα μεγάλο μέρος του σημερινού ψηφιακού κόσμου, παρέχοντας την αξιόπιστη πλατφόρμα πάνω στην οποία βασίζονται πολλές υπηρεσίες και εφαρμογές. Τα νέα, καινοτόμα προϊόντα και οι ψηφιακές υπηρεσίες που έχουν σχεδιαστεί για το μέλλον συνεχίζουν να βασίζονται επίσης στη Java. Αν και οι περισσότερες σύγχρονες εφαρμογές Java συνδυάζουν το χρόνο εκτέλεσης της Java και την εφαρμογή μαζί, υπάρχουν ακόμα πολλές εφαρμογές, ακόμα και ορισμένοι ιστότοποι, που δεν θα λειτουργούν αν δεν έχετε εγκαταστήσει μια Java για επιφάνεια εργασίας.

Εικόνα 2 Λογότυπο της Javas

Η Java σχεδιάστηκε αρχικά για χρήση σε ψηφιακές κινητές συσκευές όπως κινητά τηλέφωνα. Ωστόσο, όταν η Java 1.0 κυκλοφόρησε στο κοινό το 1996, η κύρια εστίασή της μετατοπίστηκε στη χρήση στο Διαδίκτυο, επιτρέποντας στους προγραμματιστές να αλληλεπιδρούν με τους χρήστες δίνοντάς τους. [1]

2.2 Πού χρησιμοποιείται η Java

Ο αριθμός των υπολογιστών στους οποίους εκτελείται η Java είναι στα δισεκατομμύρια. Η Java, η οποία έχει γίνει το αναπόφευκτο πρόγραμμα κάθε υπολογιστή, επιτρέπει σε πολλούς ιστότοπους να λειτουργούν ομαλά στον υπολογιστή σας με τους απλούστερους όρους. Η Java χρησιμοποιείται σε εφαρμογές όπως προγράμματα προστασίας από ιούς, Adobe Reader κ.λπ., καθώς και σε ενσωματωμένα συστήματα που δεν είναι ορατά στους κανονικούς χρήστες υπολογιστών.

Οι ιστότοποι που χρησιμοποιούν συστήματα που έχουν κατασκευαστεί με τη γλώσσα προγραμματισμού Java είναι από τους πιο δημοφιλείς τομείς χρήσης Java. Εάν δεν έχετε εγκαταστήσει Java στον υπολογιστή σας, πιθανότατα έχετε δει ένα σφάλμα κατά το

άνοιγμα ακόμη και του απλούστερου ιστότοπου. Βεβαιωθείτε ότι έχετε κάνει ενημερώσεις έτσι ώστε η γλώσσα προγραμματισμού Java που χρησιμοποιείται σχεδόν σε κάθε ιστότοπο να λειτουργεί άψογα στον υπολογιστή σας.

Ομοίως, πολλές επιχειρηματικές εταιρείες προτιμούν την Java στις εφαρμογές και τις πλατφόρμες τους. Έχουμε ήδη πει ότι είναι μια γλώσσα προγραμματισμού με υψηλή αξιοπιστία. Δεν μας εκπλήσσει το γεγονός ότι προτιμάται επίσης από εταιρικές εταιρείες.

Μια άλλη κοινή χρήση είναι τα κινητά τηλέφωνα. Πολλές εφαρμογές και συστήματα σε περίπου 3 δισεκατομμύρια τηλέφωνα είναι γραμμένα σε Java. Τα παιχνίδια σε κινητά τηλέφωνα, τα παιχνίδια 3D που ανοίγουν από ιστότοπους κ.λπ., ή τα ηλεκτρονικά παιχνίδια που αναπτύσσονται από μόνα τους επωφελούνται επίσης από τις δυνατότητες και τα πρακτικά χαρακτηριστικά που προσφέρει η Java.

Η Java, η οποία συμπεριλήφθηκε επίσης στις τηλεοράσεις, η οποία ήταν το πρώτο σημείο ανάπτυξης, χρησιμοποιείται σε 125 εκατομμύρια τηλεοράσεις.

2.3 Τα πλεόνεκτήματα της Java

1. Η Java είναι απλή:

   Μια γλώσσα προγραμματισμού είναι "απλή" αν είναι εύκολο να την μάθει και να την κατανοήσει κανείς. Η Java δεν είναι μόνο αυτό, αλλά ο κώδικάς της είναι επίσης εύκολο να αποσφαλματωθεί και να συντηρηθεί με την πάροδο του χρόνου. Είναι επίσης λιγότερο πολύπλοκη από άλλες παρόμοιες γλώσσες όπως η C και η C++. Αυτές οι γλώσσες έχουν πολλά πολύπλοκα χαρακτηριστικά, όπως κλάσεις αποθήκευσης, υπερφόρτωση τελεστών και έννοιες ρητών δεικτών, που μερικές φορές είναι συντριπτικά και δεν υπάρχουν στη Java.
2. Διαθέτει αντικειμενοστραφή προγραμματισμό:

   Η Java υιοθετεί τον αντικειμενοστραφή προγραμματισμό, ο οποίος είναι μια έννοια κωδικοποίησης που ορίζει όχι μόνο τον τύπο των δεδομένων και τη δομή τους, αλλά και το σύνολο των λειτουργιών που εφαρμόζονται σε αυτά. Αυτή η μέθοδος προγραμματισμού καθιστά τη δομή δεδομένων ένα αντικείμενο που μπορεί να χειριστεί για να επιτρέψει τη δημιουργία σχέσεων μεταξύ άλλων αντικειμένων.
3. Η Java είναι μια γλώσσα υψηλού επιπέδου με ήπια μόνο καμπύλη εκμάθησης:

   Είτε ένας προγραμματιστής μόλις ξεκινάει είτε έχει πολυετή εμπειρία, είναι εύκολο να μάθει Java. Ως γλώσσα υψηλού επιπέδου, η Java μοιάζει πολύ με την ανθρώπινη γλώσσα. Αυτό σημαίνει ότι εκεί που άλλες γλώσσες μοιάζουν με κώδικα μηχανής, η Java και άλλες επιλογές υψηλού επιπέδου απαιτούν μετατροπή με τη χρήση μεταγλωττιστών και διερμηνέων. Αυτό καθιστά την ανάπτυξη γενικά πολύ πιο απλή.
4. Είναι μια ασφαλής γλώσσα προγραμματισμού:

   Η Java μειώνει στην πραγματικότητα τις απειλές και τους κινδύνους ασφαλείας αποφεύγοντας τη χρήση ρητών δεικτών. Οι δείκτες μερικές φορές προκαλούν μη εξουσιοδοτημένη πρόσβαση στη μνήμη, καθώς αποθηκεύουν τη διεύθυνση μνήμης άλλων τιμών. Καθώς η έννοια των δεικτών δεν υπάρχει στη

Java, αυτό δεν αποτελεί πρόβλημα. Η γλώσσα προσφέρει επίσης έναν διαχειριστή ασφαλείας για κάθε εφαρμογή που επιτρέπει στους προγραμματιστές να ορίζουν κανόνες για την πρόσβαση σε κάθε κλάση.

5. Η Java είναι μια κατανεμημένη γλώσσα:

   Η Java παρέχει έναν μηχανισμό για την κοινή χρήση δεδομένων και προγραμμάτων σε πολλούς υπολογιστές, ώστε να βελτιωθεί η αποδοτικότητα και η απόδοση του συστήματος. Αυτό επιτρέπει την κατανεμημένη υπολογιστική ή τη συνεργασία πολλών υπολογιστών σε ένα δίκτυο. Αυτό αποτελεί μεγάλη βοήθεια στην ανάπτυξη εφαρμογών σε δίκτυα που μπορούν να συμβάλουν τόσο στη λειτουργικότητα της εφαρμογής όσο και στη λειτουργικότητα των δεδομένων.

6. Η Java προσφέρει διάφορα API για την ανάπτυξη εφαρμογών:

   Τα API της Java (ή Διεπαφές προγραμματισμού εφαρμογών) είναι σύνολα εντολών ή μεθόδων επικοινωνίας μεταξύ διαφόρων δραστηριοτήτων. Αυτό περιλαμβάνει πράγματα όπως σύνδεση βάσης δεδομένων, δικτύωση, βοηθητικά προγράμματα και άλλα.

7. Υποστηρίζει πολυνηματικότητα:

   Αυτό σημαίνει ότι διαθέτει τη δυνατότητα ενός προγράμματος να εκτελεί ταυτόχρονα πολλές εργασίες ή νήματα μέσα σε ένα πρόγραμμα. [2]

3. Eclipse και Spring boot

3.1 Eclipse

3.1.1 Τι είναι Eclipse

Το Eclipse είναι μια δωρεάν πλατφόρμα ανάπτυξης βασισμένη στη Java, γνωστή για τα πρόσθετα που επιτρέπει στους προγραμματιστές να αναπτύσσουν και να δοκιμάζουν κώδικα γραμμένο σε άλλες γλώσσες προγραμματισμού. Το Eclipse κυκλοφορεί υπό τους όρους της Eclipse Public License.Το Ίδρυμα Eclipse είναι μια ανεξάρτητη, μη κερδοσκοπική εταιρεία με έδρα τον Καναδά, η οποία διαχειρίζεται την κοινότητα ανάπτυξης λογισμικού ανοικτού κώδικα Eclipse και περιλαμβάνει τη νομική δικαιοδοσία της Ευρωπαϊκής Ένωσης.Το Eclipse υποστηρίζεται από πάνω από 320 μέλη, 1.750 committers και περισσότερες από 332 εκατομμύρια γραμμές κώδικα. Στόχος του ιδρύματος είναι η δημιουργία μιας κοινότητας και ενός οικοσυστήματος συμπληρωματικών προϊόντων και υπηρεσιών. [3]

3.1.2 Μια σύντομη ιστορία των εμπορικών IDE

Το Eclipse ξεκίνησε το 2001, όταν η IBM δώρισε τρία εκατομμύρια γραμμές κώδικα από τα εργαλεία της Java για την ανάπτυξη ενός ολοκληρωμένου περιβάλλοντος ανάπτυξης (IDE) ανοικτού κώδικα. Το Eclipse IDE επιβλέφθηκε αρχικά από μια κοινοπραξία προμηθευτών λογισμικού που επεδίωκε να δημιουργήσει και να προωθήσει μια νέα κοινότητα που θα συμπλήρωνε την κοινότητα ανοικτού κώδικα του Apache. Έχει ειπωθεί, αν και δεν έχει επιβεβαιωθεί, ότι το όνομα της πλατφόρμας προήλθε από έναν

δευτερεύοντα στόχο, ο οποίος ήταν να επισκιάσει το δημοφιλές IDE της Microsoft, το Visual Studio. Το 2011, η Oracle έγινε πάροχος του Eclipse, δωρίζοντας τον διακομιστή συνεχούς ολοκλήρωσης Hudson που κληρονόμησε από τη Sun Microsystems και την πλατφόρμα Java 2, Enterprise Edition (Java EE), το 2017.Το 2016, η Microsoft ανακοίνωσε ότι θα ενταχθεί στο Eclipse Foundation και θα υποστηρίξει την ενσωμάτωση του Visual Studio δίνοντας στους προγραμματιστές του Eclipse πλήρη πρόσβαση στις υπηρεσίες Visual Studio Team Services.Το διοικητικό συμβούλιο της Eclipse περιλαμβάνει τον εκτελεστικό διευθυντή Mike Milinkovich και στρατηγικά μέλη από τις CA Technologies, IBM, Oracle και SAP.

3.1.3 Η αρχιτεκτονική Eclipse

Η αρχιτεκτονική του Eclipse βασίζεται στις ακόλουθες κλασικές έννοιες: βασικό περιβάλλον εκτέλεσης της πλατφόρμας(Core platform run-time), πρόσθετα(plug-ins) και σημεία επέκτασης(Extension Points) [4]

3.1.4 Χρόνος εκτέλεσης πλατφόρμας (Core platform run-time)

Ο χρόνος εκτέλεσης της πλατφόρμας είναι υπεύθυνος για τη διαμόρφωση της πλατφόρμας, τον εντοπισμό των διαθέσιμωνπρόσθετων προγραμμάτων, διαβάζοντας τα αρχεία δηλωτικών τους και δημιουργώντας δυναμικά ένα μητρώο πρόσθετων προγραμμάτων. ToTo Platform Runtime εκτελεί επίσης ορισμένες εργασίες επικύρωσης, όπως η ανίχνευση επεκτάσεωνσε σημεία επέκτασης που δεν υπάρχουν. Αφού ολοκληρωθούν όλες αυτές οι εργασίες, ο χρόνος εκτέλεσης καθιστά το μητρώο και τη διαμόρφωση της πλατφόρμας διαθέσιμα στα πρόσθετα προγράμματα. [4]

3.1.5 Πρόσθετα (plug-ins)

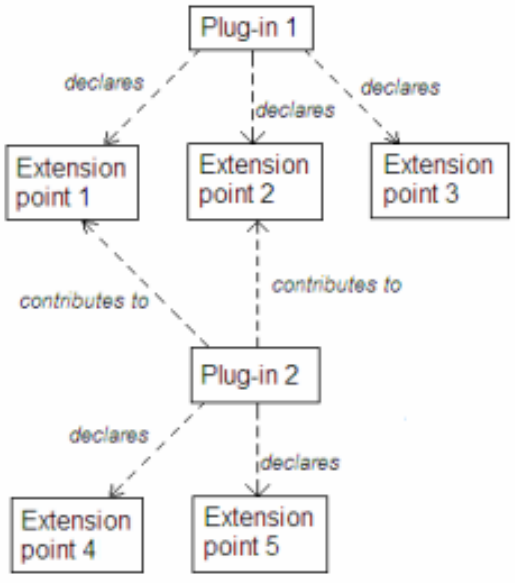
Ένα plug-in (ή ένα component) είναι η μικρότερη μονάδα συμπεριφοράς στην πλατφόρμα Eclipse. Οποιοδήποτε εργαλείο μπορεί να αναπτυχθεί ως πρόσθετο πρόγραμμα ή ως σύνολο πρόσθετων προγραμμάτων. Στην πραγματικότητα, όλα τα προγράμματα του Eclipse η λειτουργικότητα βρίσκεται σε πρόσθετα, με εξαίρεση τη λειτουργικότητα στο πρόγραμμα core Platform Runtime. Ένα πρόσθετο πρόγραμμα διαθέτει επίσης ένα αρχείο manifest το οποίο περιέχει πληροφορίες σχετικά με το πρόσθετο πρόγραμμα το ίδιο, όπως τα σημεία επέκτασης (εξηγείται στη συνέχεια) που παρέχει το πρόσθετο και τι επεκτάσεις που παρέχει το πρόσθετο σε σημεία επέκτασης που ορίζονται από άλλα πρόσθετα. Ο λόγος για τον οποίο αυτό το αρχείο δηλωτικού βρίσκεται ξεχωριστά από το ίδιο το πρόσθετο είναι ότι το Runtime μπορεί να διαβάσει τις πληροφορίες σχετικά με το πρόσθετο πρόγραμμα χωρίς να φορτώσει το πρόσθετο πρόγραμμα στο μνήμη. Με αυτόν τον τρόπο, το Runtime μπορεί να δημιουργήσει το μητρώο χωρίς να φορτώσει κάθε πρόσθετο, εξοικονόμηση πόρων.

Τα πρόσθετα μπορούν επίσης να εξαρτώνται το ένα από το άλλο. Αυτές οι εξαρτήσεις εκφράζονται στο το αρχείο manifest του plug-in. Με αυτόν τον τρόπο, ένα πρόσθετο θα

χρησιμοποιηθεί (και, φυσικά, θα φορτωθεί επιτυχώς) μόνο εάν όλα τα άλλα πρόσθετα από τα οποία εξαρτάται έχουν επίσης φορτωθεί επιτυχώς

3.1.6 Σημεία επέκτασης (Extension Points)

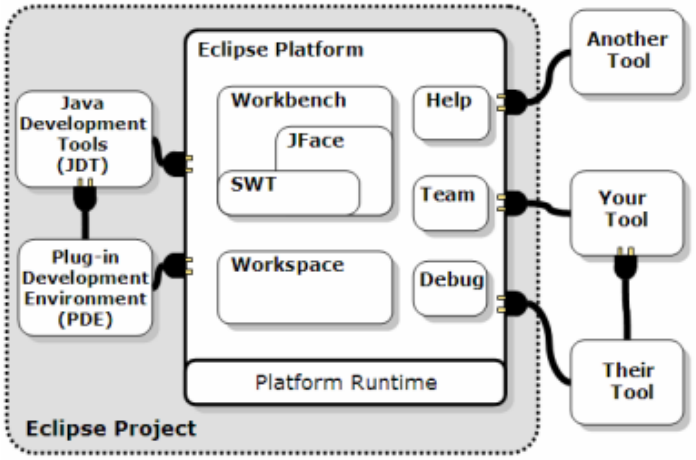
Ένα σημείο επέκτασης είναι ένας τρόπος με τον οποίο τα πρόσθετα μπορούν να συνεργάζονται μεταξύ τους. Αυτό το μοντέλο διασύνδεσης επιτυγχάνεται μέσω της κλασικής διαδικασίας: (1) κάθε plug-in μπορεί να δηλώσει οποιονδήποτε αριθμό σημείων επέκτασης- (2) από την άλλη πλευρά, κάθε plug-in μπορεί να παρέχειοποιονδήποτε αριθμό επεκτάσεων (που αποκαλούνται επίσης "συνεισφορές" στην κοινότητα του Eclipse) γιαένα ή περισσότερα σημεία επέκτασης που έχουν δηλωθεί από άλλα πρόσθετα. Για παράδειγμα, το Workbench το πρόσθετο δηλώνει ένα σημείο επέκτασης για τις προτιμήσεις του χρήστη. Οποιοδήποτε πρόσθετο μπορεί να συνεισφέρει τις δικές του προτιμήσεις χρήστη, καθορίζοντας επεκτάσεις σε αυτό το σημείο. Ένα σημείο επέκτασης μπορεί να έχει μια αντίστοιχη διασύνδεση API, που ορίζεται χρησιμοποιώντας ένα άλλο προηγούμενο API ή χρησιμοποιώντας ένα νέο API. Άλλα πρόσθετα μπορούν να παρέχουν επεκτάσεις σε ένα σημείο επέκτασης υλοποιώντας τη διεπαφή. Το Σχήμα παρακάτω απεικονίζει τον τρόπο με τον οποίο συνδέονται τα πρόσθετα και τα σημεία επέκτασης. Το πρόσθετο 1 δηλώνει τρία σημεία επέκτασης (σημείο επέκτασης 1, σημείο επέκτασης 2 και σημείο επέκτασης σημείο 3). Το σημείο επέκτασης 1 και το σημείο επέκτασης 2 λαμβάνουν την επέκταση (ή τη συνεισφορά) από το πρόσθετο 2. Από την άλλη πλευρά, το Plug-in 2 δηλώνει επίσης δύο άλλες επεκτάσεις σημεία (σημείο επέκτασης 4 και σημείο επέκτασης 5). Αυτά τα δύο τελευταία σημεία επέκτασης μπορούν το πρόσθετο 1 μπορεί επίσης να συνεισφέρει σε άλλα υπάρχοντα σημεία επέκτασης, τα οποία δεν απεικονίζονται στο σχήμα.



Διάγραμμα 1. 2 Extension Points

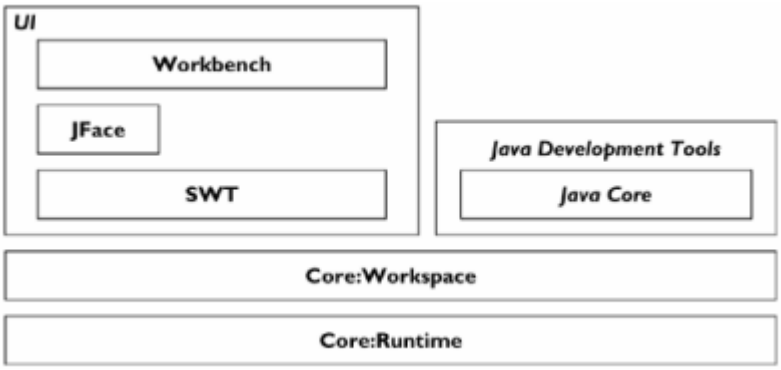
3.1.7 Οργάνωση plug-in της πλατφόρμας Eclipse

Η πλατφόρμα Eclipse διαθέτει διάφορα components (δηλ. plug-ins), εκτός από αυτά που που μπορούν να δημιουργήσουν οι προγραμματιστές. Στην Εικόνα 4 παρουσιάζονται τα κύρια στοιχεία του Eclipse. Ως παράδειγμα χρήσης συστατικών, όλες οι εφαρμογές Eclipse που διαθέτουν γραφικό περιβάλλον χρήστη θα έχουν εξαρτήσεις από το πρόσθετο WorkBench, το οποίο θα εξαρτάται επίσης από το SWT (Standard Widget Toolkit) και τα πρόσθετα JFace. Ωστόσο, ένα plug-in με διεπαφή χρήστη μόνο με κείμενο δεν θα τις χρειάζονται .



Εικόνα 3 Οργάνωση plug-in της πλατφόρμας Eclipse

Στην εικόνα 1.3 παρουσιάζεται μια άλλη άποψη της οργάνωσης της πλατφόρμας Eclipse, αυτή τη φορά από μια πιο πολυεπίπεδη οπτική γωνία, καθώς και των πρόσθετων προγραμμάτων που χρησιμοποιούνται για την παροχή των πιοκοινές χρήσεις. Σε αυτό το σχήμα, παρατηρήστε το πρόσθετο Java Development Tools, το οποίο παρέχει στην πλατφόρμα Eclipse τη δυνατότητα να λειτουργεί ως Java IDE. [5]



Εικόνα 1. 1 Οργάνωση plug-in της πλατφόρμας Eclipse 2

3.2 Spring Boot

3.2.1 Τι είναι το Spring boot

Το Springboot είναι ένα Framework που αναπτύχθηκε στη γλώσσα προγραμματισμού Java και επιτρέπει την εγγραφή ισχυρών MicroService σε Java. Χάρη στο Springboot, μπορούμε να γράψουμε τις λειτουργίες της βάσης δεδομένων μας ως Java και Language Integrated κατά την ανάπτυξη εφαρμογών backend και αυτές οι λειτουργίες εκτελούνται από το Springboot. Ταυτόχρονα, μπορούμε να δημιουργήσουμε δεδομένα εισόδου-εξόδου παρέχοντας υπηρεσίες σε εφαρμογές από την πλευρά του πελάτη. [6]

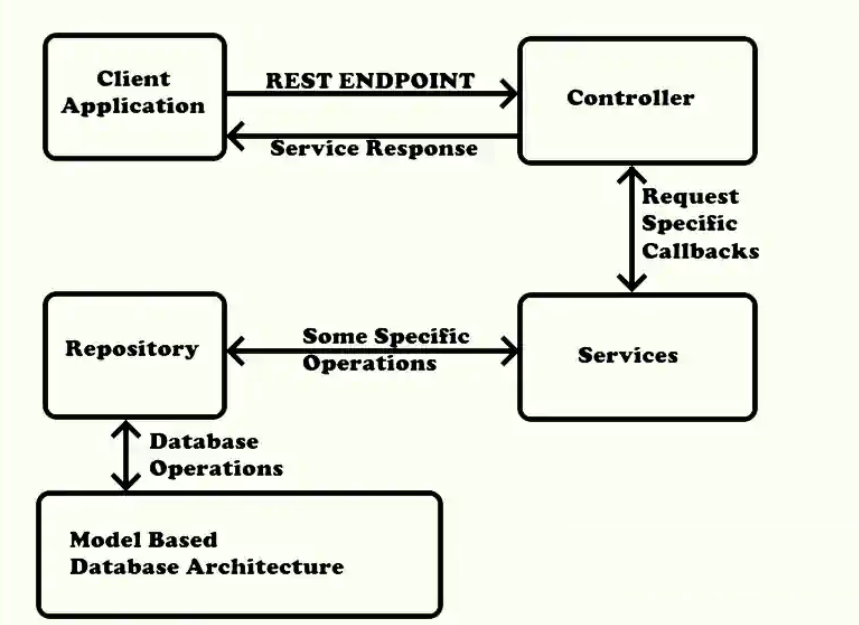
Εικόνα 4 Λογότυπο της Spring Boot

Με το Springboot:

1- Γράφοντας λιγότερο κώδικα βάσης δεδομένων, μπορούμε να αναπτυχθούμε γρηγορότερα με λιγότερη απώλεια χρόνου.
2- Η αρχιτεκτονική springboot είναι πιο κατανοητή. Με αυτόν τον τρόπο, μπορούν να αναπτυχθούν πιο αποτελεσματικές εφαρμογές.
3- Με τη γλώσσα προγραμματισμού Java, μπορούμε να γράψουμε εντολές βάσης δεδομένων ενσωματωμένης γλώσσας.
4- Προσφέρει χρήση REST για τους σκοπούς της υπηρεσίας και διαθέτει ισχυρή διαμόρφωση REST.
5- Πολλές διαμορφώσεις γίνονται αυτόματα. Μπορείτε να το αλλάξετε αν θέλετε.
6- Χρησιμοποιώντας σχολιασμούς, μπορούν να γίνουν αλλαγές στην αρχιτεκτονική εφαρμογών και εντός εφαρμογής.
7- Έχει το δικό του κοντέινερ servlet. Με αυτόν τον τρόπο, λαμβάνει τις αιτήσεις που φτάνουν στο τέλος της υπηρεσίας, τις επεξεργάζεται και τις κατευθύνει στις σχετικές εντολές.
8- Το Java Bean είναι εύκολο στη διαχείριση. Λειτουργεί επίσης σύμφωνα με την επιθυμητή διαχείριση βάσεων δεδομένων.

3.2.2 Ανάπτυξη Rest Api Server

Σε γενικές γραμμές, μια βασική υλοποίηση springboot έχει τα ακόλουθα επίπεδα επεξεργασίας. Προκειμένου να γίνει τακτική ανάπτυξη, είναι σημαντικό οι συναλλαγές να γίνονται από το σχετικό επίπεδο. Διαφορετικά, είναι πιθανό να προκύψουν πολυπλοκότητες σε μεγάλα έργα. [6]



Διάγραμμα 1. 3 Λειτουργία Spring Boot

1- Επίπεδο Model: Στο επίπεδο μοντέλου, προετοιμάζονται τα μοντέλα των πινάκων της βάσης δεδομένων μας και προσδιορίζονται οι διαμορφώσεις. Για να δημιουργήσετε μοντέλα σε εφαρμογές springboot, δημιουργείται μια κανονική κλάση Java και προστίθενται τα ονόματα των στηλών σε αυτήν. Εν τω μεταξύ, προσαρμογές όπως οι τύποι δεδομένων και τα μήκη των στηλών καθορίζονται χρησιμοποιώντας σχολιασμούς. Επειδή αυτή η κλάση αντιπροσωπεύει μια Οντότητα, η ίδια η κλάση δηλώνεται από @Entity ανοδίωση. Ταυτόχρονα, οι κατασκευαστές θα πρέπει να καθορίζονται μαζί με τους συλλέκτες και τους ρυθμιστές. Με αυτόν τον τρόπο, μπορούν να γίνουν πρόσθετοι έλεγχοι μοντέλων.

2- Επίπεδο Repository: Η επικοινωνία που θα δημιουργήσει η εφαρμογή με τη βάση δεδομένων παρέχεται μέσω του επιπέδου Repository. Το επίπεδο Repository είναι γραμμένο ως διεπαφή που κληρονομείται από την κλάση JPARepository. Το επίπεδο Repository μας επιτρέπει να καθορίσουμε τις λειτουργίες της βάσης δεδομένων απευθείας σε αυτό. Με αυτόν τον τρόπο, γράφουμε μόνο τα ονόματα των μεθόδων χωρίς να γράφουμε κώδικα SQL και κατά τη διάρκεια της φάσης μεταγλώττισης δημιουργούνται αυτόματα τα σχετικά ερωτήματα. Εάν οι εντολές που θέλουμε να γράψουμε είναι πολύ

περίπλοκες για να τις κατανοήσει η κλάση JPA, η εντολή αναφέρεται στη σχετική εντολή με το απαραίτητο ερώτημα SQL @Query ανωμαλία και το καθορισμένο ερώτημα εκτελείται όταν καλείται η μέθοδος. Αυτή η λειτουργικότητα του επιπέδου αποθετηρίου μας εξοικονομεί πολύ χρόνο κατά τη διάρκεια της ανάπτυξης.

3- Επίπεδο Controller: Τα παραγόμενα άκρα υπηρεσίας REST καθορίζονται από το επίπεδο του ελεγκτή και η επικοινωνία με την υπηρεσία πραγματοποιείται μέσω των ελεγκτών. Το επίπεδο ελεγκτή είναι η πύλη προς το εξωτερικό της εφαρμογής μας. Η εφαρμογή μας καθορίζει τα REST API εδώ και επιστρέφει μια απάντηση στον πελάτη μεταφέροντας τα δεδομένα που προέρχονται από το εξωτερικό στα καθορισμένα API μέσω των σχετικών λειτουργιών. Οι καθορισμένοι τερματισμοί υπηρεσίας REST αντιστοιχίζονται από το Springboot. Η μικροεφαρμογή εξυπηρετητή δρομολογεί τα εισερχόμενα αιτήματα στις σχετικές μεθόδους μέσω αυτής της χαρτογράφησης. Με αυτόν τον τρόπο, η εφαρμογή Springboot επικοινωνεί με το εξωτερικό.

Μετά την ανάπτυξη του Server μας, έχουν δημιουργηθεί οι παρακάτω Web Services που μας δίνουν την δυνατότητα να στέλνουμε ή να παίρνουμε δεδομένα, να κάνουμε αλλαγές ή και να διαγράφουμε δεδομένα από την βάση μας μέσου Server πολύ απλά χρησιμοποιώντας την διαδρομή /productnames/(Μέθοδος).

Action	HTTP METHOD	ΠΕΡΙΓΡΑΦΗ
GET	/getall	Μας φέρνει όλα τα προϊόντα από την βάση μας
GET	/{id}	Μας φέρνει τα προϊόντα που με το id που έχει ορίσει ο χρήστης
GET	/0/0	Μας φέρνει όλα τα προϊόντα με υπόλοιπο 0
GET	/1/{qty}	Μας φέρνει τα προϊόντα από έναν επιθυμητό ποσότητα και πάνω
GET	barcode2/{barcode}	Μας φέρνει τα προϊόντα με το Barcode που ορίζει ο χρήστης και είναι ίσο πάνω 0
GET	barcode/{barcode}	Μας φέρνει τα προϊόντα με το Barcode που ορίζει ο χρήστης και είναι ίσο με 0

POST	/post	Ανεβάζει το καινούριο προϊόν στο Server
DELETE	/delete/{id}	Διαγραφή το προϊόν με το id που ορίζει ο χρήστης
PUT	/{id}	Επεξεργασία το προϊόν που ορίζει ο χρήστης

Πίνακας 1. 1 Http Μέθοδοι

3.2.3 Ανάλυση POST–GET –PUT-DELETE

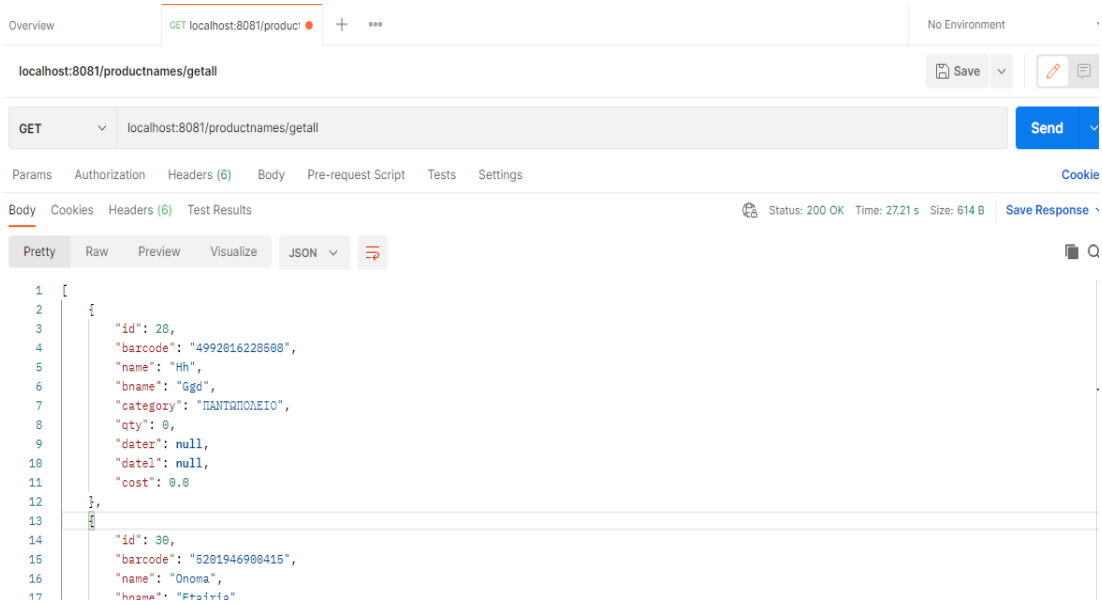
1) GET: Εάν θέλουμε να τραβήξουμε μόνο (να διαβάσουμε) δεδομένα από το Server, δηλαδή εάν δεν θα γίνουν αλλαγές (προσθήκες, διαγραφές, τροποποιήσεις) στα δεδομένα, συνιστάται να χρησιμοποιήσετε τη μέθοδο GET.Από τις λειτουργίες CRUD μπορούμε να πούμε ότι αντιστοιχεί στο Read.Π.χ.: Όταν χρησιμοποιούμε το GET /students, μας επιστρέφουν μια λίστα μαθητών.
2) POST: Μπορούμε να το χρησιμοποιήσουμε συμπληρώνοντας το μέρος του σώματος του Server Api και όταν θέλουμε να κάνουμε αλλαγές στα δεδομένα. Αυτό που σημαίνει η πραγματοποίηση αλλαγών περιλαμβάνει τα τμήματα CREATE και UPDATE των λειτουργιών CRUD. Για παράδειγμα: Εισαγωγή πληροφοριών χρήστη στην ενότητα σώματος με POST /createUser και αίτημα CREATE χρήστη στη βάση δεδομένων
3) PUT: Έχει τα χαρακτηριστικά του αιτήματος POST. Έτσι το χρησιμοποιούμε όταν θέλουμε να κάνουμε λειτουργίες δημιουργίας και ενημέρωσης από λειτουργίες CRUD. Η πλευρά που φεύγει από τη θέση είναι ότι το αίτημα Put ορίζεται ως idempotent και δεν μπορεί να αποθηκευτεί προσωρινά
4) ΔΙΑΓΡΑΦΗ: Αντιστοιχεί στο Διαγραφή από τις εργασίες CRUD. Συνιστάται να το χρησιμοποιείτε όταν θέλουμε να διαγράψουμε ένα δεδομένο

3.3 Postman

3.3.1 Τί είναι το Postman

Το Postman είναι μια εφαρμογή που μπορούμε να χρησιμοποιήσουμε ως επέκταση Chrome ή που μπορούμε να κατεβάσουμε και να εγκαταστήσουμε απευθείας στον υπολογιστή μας. Μπορεί επίσης να οριστεί ως Rest Client.Το API (Application Programming Interface) επιτρέπει σε διαφορετικές εφαρμογές να αλληλοεπιδρούν μεταξύ τους. Το λογισμικό πελάτη (Android) και backend (java) που επικοινωνεί με το Restfull Api μπορεί να δοθεί ως παράδειγμα. Με τον Postman, μπορούμε εύκολα να δοκιμάσουμε τα API μας αντί να γράφουμε μακροσκελείς κώδικες. Χάρη στις πολλές δυνατότητές του, μπορούμε εύκολα να προετοιμάσουμε αιτήματα και να χρησιμοποιήσουμε τιμές εισερχόμενης απόκρισης. Για παράδειγμα χρησιμοποιώντας την διεύθυνση

localhost:8081/productnames και το μέθοδο /getall , θα μας εμφάνισή στο Postman όλα τα προϊόντα που είναι στη βάση δεδομένων. [7]



Εικόνα 5 Αίτημα /getall προς το Server

3.3.2 Γιατί να χρησιμοποιήσετε το Postman

1- Με πάνω από 4 εκατομμύρια χρήστες σήμερα, το Postman Software έχει γίνει το εργαλείο επιλογής για τους ακόλουθους λόγους:

2- Προσβασιμότητα - Για να χρησιμοποιήσετε το εργαλείο Postman, θα πρέπει απλώς να συνδεθείτε στους δικούς σας λογαριασμούς, καθιστώντας εύκολη την πρόσβαση σε αρχεία οποτεδήποτε και οπουδήποτε, αρκεί να είναι εγκατεστημένη μια εφαρμογή Postman στον υπολογιστή.

3- Χρήση συλλογών - Το Postman επιτρέπει στους χρήστες να δημιουργούν συλλογές για τις κλήσεις API του Postman. Κάθε συλλογή μπορεί να δημιουργήσει υποφακέλους και πολλαπλές αιτήσεις. Αυτό βοηθά στην οργάνωση των συνόλων δοκιμών σας.

4- Συνεργασία - Οι συλλογές και τα περιβάλλοντα μπορούν να εισαχθούν ή να εξαχθούν, καθιστώντας εύκολη την κοινή χρήση αρχείων. Ένας άμεσος σύνδεσμος μπορεί επίσης να χρησιμοποιηθεί για την κοινή χρήση συλλογών.

5- Δημιουργία περιβαλλόντων - Η ύπαρξη πολλαπλών περιβαλλόντων βοηθάει στη λιγότερη επανάληψη των δοκιμών, καθώς μπορεί κανείς να χρησιμοποιήσει την ίδια συλλογή αλλά για διαφορετικό περιβάλλον. Εδώ θα γίνει η παραμετροποίηση, την οποία θα συζητήσουμε σε επόμενα μαθήματα.

6- Δοκιμές αυτοματοποίησης - Μέσω της χρήσης του Collection Runner ή του Newman, οι δοκιμές μπορούν να εκτελούνται σε πολλαπλές επαναλήψεις εξοικονομώντας χρόνο για επαναλαμβανόμενες δοκιμές.

7- Αποσφαλμάτωση - Η κονσόλα του Postman βοηθά στον έλεγχο των δεδομένων που έχουν ανακτηθεί, καθιστώντας εύκολη την αποσφαλμάτωση των δοκιμών.

8- Συνεχής ολοκλήρωση - Με τη δυνατότητά του να υποστηρίζει συνεχή ολοκλήρωση, διατηρούνται οι πρακτικές ανάπτυξης.

3.4 Heroku

3.4.1 Τι είναι το Heroku

Το Heroku είναι πάροχος υπηρεσιών υποδομής εφαρμογών υπολογιστικού νέφους (Cloud Computing), το οποίο μπορούμε να εκφράσουμε ως Πλατφόρμα ως Υπηρεσία (PaaS) που μας επιτρέπει να μεταφέρουμε τις διαδικτυακές εφαρμογές μας που αναπτύχθηκαν με JavaScript, Ruby, Java, PHP, Python, Golang, Scala και Clojure στο διαδίκτυο (ανάπτυξη, διαχείριση και κλιμάκωση). Για αυτές τις λειτουργίες, χρησιμοποιούνται μικρά και ελαφριά εικονικά περιβάλλοντα βασισμένα στο Linux που ονομάζονται Dyno. Το Heroku προσφέρει επίσης πρόσθετη υποστήριξη. Στη σελίδα Add-ons2 μπορείτε να δείτε τη λίστα των πρόσθετων πακέτων που έχουν δημιουργηθεί από το Heroku και άλλους προγραμματιστές/παρόχους υπηρεσιών 3rd party. Για παράδειγμα, με το mLab MongoDB3, μπορείτε να συσχετίσετε την υπηρεσία Node.js με τη βάση δεδομένων. Επιπλέον, είναι δυνατό να διατηρήσετε εφαρμογές σε υπηρεσίες όπως το git και το bitbucket και να τις ενσωματώσετε στο Heroku. Για την πρόσβαση στην πλατφόρμα Heroku πρέπει πρώτα να δημιουργηθεί ένας λογαριασμός. Μπορείτε να χρησιμοποιήσετε τη σελίδα Εγγραφή Heroku για αυτήν τη διαδικασία. Αφού δημιουργήσετε τον λογαριασμό και συνδεθείτε, πρέπει να ορίσουμε το κλειδί SSH από τη σελίδα Ρυθμίσεις λογαριασμού. [8]

Εικόνα 6 Λογότυπο του Heroku

3.4.2 Τι είναι το Cloud computing

Το cloud computing είναι ένας γενικός όρος για οτιδήποτε αφορά την παροχή υπηρεσιών που φιλοξενούνται μέσω του διαδικτύου. Οι υπηρεσίες αυτές χωρίζονται σε τρεις κύριες κατηγορίες ή τύπους υπολογιστικού νέφους: υποδομή ως υπηρεσία (IaaS), πλατφόρμα ως υπηρεσία (PaaS) και λογισμικό ως υπηρεσία (SaaS). Ένα νέφος μπορεί να είναι ιδιωτικό ή δημόσιο. Ένα δημόσιο νέφος πωλεί υπηρεσίες σε οποιονδήποτε στο διαδίκτυο. Το ιδιωτικό νέφος είναι ένα ιδιόκτητο δίκτυο ή ένα κέντρο δεδομένων που παρέχει υπηρεσίες φιλοξενίας σε περιορισμένο αριθμό ατόμων, με συγκεκριμένες ρυθμίσεις πρόσβασης και δικαιωμάτων. Ιδιωτικό ή δημόσιο, ο στόχος του υπολογιστικού νέφους είναι να παρέχει εύκολη, επεκτάσιμη πρόσβαση σε υπολογιστικούς πόρους και υπηρεσίες πληροφορικής. Η υποδομή υπολογιστικού νέφους περιλαμβάνει τα στοιχεία

υλικού και λογισμικού που απαιτούνται για τη σωστή εφαρμογή ενός μοντέλου υπολογιστικού νέφους. Το cloud computing μπορεί επίσης να θεωρηθεί ως utility computing ή on-demand computing. [9]

1- IaaS. Οι πάροχοι IaaS, όπως η Amazon Web Services (AWS), παρέχουν μια εικονική παρουσία διακομιστή και αποθηκευτικό χώρο, καθώς και διεπαφές προγραμματισμού εφαρμογών (API) που επιτρέπουν στους χρήστες να μεταφέρουν φορτία εργασίας σε μια εικονική μηχανή (VM). Οι χρήστες διαθέτουν μια χωρητικότητα αποθήκευσης που τους έχει εκχωρηθεί και μπορούν να ξεκινούν, να σταματούν, να έχουν πρόσβαση και να ρυθμίζουν το VM και την αποθήκευση όπως επιθυμούν. Οι πάροχοι IaaS προσφέρουν μικρές, μεσαίες, μεγάλες, πολύ μεγάλες και βελτιστοποιημένες ως προς τη μνήμη ή τον υπολογισμό περιπτώσεις, εκτός από τη δυνατότητα προσαρμογής των περιπτώσεων, για διάφορες ανάγκες φόρτου εργασίας. Το μοντέλο νέφους IaaS είναι πιο κοντά σε ένα απομακρυσμένο κέντρο δεδομένων για τους επιχειρηματικούς χρήστες.

2- PaaS. Στο μοντέλο PaaS, οι πάροχοι νέφους φιλοξενούν εργαλεία ανάπτυξης στις υποδομές τους. Οι χρήστες έχουν πρόσβαση σε αυτά τα εργαλεία μέσω του διαδικτύου χρησιμοποιώντας API, διαδικτυακές πύλες ή λογισμικό πύλης. Το PaaS χρησιμοποιείται για τη γενική ανάπτυξη λογισμικού και πολλοί πάροχοι PaaS φιλοξενούν το λογισμικό μετά την ανάπτυξή του. Τα κοινά προϊόντα PaaS περιλαμβάνουν το Lightning Platform της Salesforce, το AWS Elastic Beanstalk και το Google App Engine.

3- SaaS. Το SaaS είναι ένα μοντέλο διανομής που παρέχει εφαρμογές λογισμικού μέσω του διαδικτύου- οι εφαρμογές αυτές συχνά αποκαλούνται υπηρεσίες ιστού. Οι χρήστες μπορούν να έχουν πρόσβαση σε εφαρμογές και υπηρεσίες SaaS από οποιαδήποτε τοποθεσία χρησιμοποιώντας υπολογιστή ή κινητή συσκευή που έχει πρόσβαση στο διαδίκτυο. Στο μοντέλο SaaS, οι χρήστες αποκτούν πρόσβαση σε λογισμικό εφαρμογών και βάσεις δεδομένων. Ένα συνηθισμένο παράδειγμα εφαρμογής SaaS είναι το Microsoft 365 για υπηρεσίες παραγωγικότητας και ηλεκτρονικού ταχυδρομείου.

4-

3.4.3 Πού ταιριάζει το Heroku στο Cloud Computing

Η Heroku είναι μια πλατφόρμα ως υπηρεσία (PaaS) που βασίζεται σε κοντέινερ. Οι προγραμματιστές χρησιμοποιούν το Heroku για την ανάπτυξη, διαχείριση και κλιμάκωση σύγχρονων εφαρμογών. Η πλατφόρμα μας είναι κομψή, ευέλικτη και εύχρηστη, προσφέροντας στους προγραμματιστές την απλούστερη διαδρομή για την προώθηση των εφαρμογών τους στην αγορά. Η Heroku είναι πλήρως διαχειρίσιμη, δίνοντας στους προγραμματιστές την ελευθερία να επικεντρωθούν στο βασικό τους προϊόν χωρίς να τους

αποσπά την προσοχή η συντήρηση διακομιστών, υλικού ή υποδομών. Η εμπειρία της Heroku παρέχει υπηρεσίες, εργαλεία, ροές εργασίας και πολύγλωσσή υποστήριξη - όλα σχεδιασμένα για να βελτιώνουν την παραγωγικότητα των προγραμματιστών.

3.4.4 Πλεονεκτήματα Heroku

1- Το ξεκίνημα με το Heroku είναι πολύ εύκολο

Είναι αρκετά απλό για τους αρχάριους να δημιουργήσουν την πρώτη τους εφαρμογή. Χρειάζονται λιγότερο από πέντε λεπτά. Ως επί το πλείστων είναι η εκτέλεση απλών εντολών Git για την ανάπτυξη μιας εφαρμογής και την εκτέλεσή της. Η τεκμηρίωση στον επίσημο ιστότοπο της Heroku βοηθά επίσης πολύ.

2- Εξαιρετικά αρχεία καταγραφής σφαλμάτων

Ένα τυπικό αρχείο καταγραφής σφαλμάτων εμφανίζεται όταν η ανάπτυξή σας αποτυγχάνει στο Heroku. Αυτή η δυνατότητα είναι ελάχιστα διαθέσιμη σε άλλες πλατφόρμες PaaS- οι περισσότερες εναλλακτικές λύσεις δίνουν μη περιγραφικά μηνύματα που καθιστούν την αποσφαλμάτωση μια κολοσσιαία πρόκληση. Η αποσφαλμάτωση του Heroku είναι πιο απλή από άλλους παρόχους PaaS, καθιστώντας την ανάπτυξη σχετικά εύκολη.

3- Τα πρόσθετα διευκολύνουν την προσθήκη νέων λειτουργιών και τεχνολογιών

Η Heroku παρέχει τη δυνατότητα προσθήκης ενός τεράστιου καταλόγου πρόσθετων προγραμμάτων και υπηρεσιών σε μια παρουσία. Αυτά τα πρόσθετα καλύπτουν λειτουργίες που κυμαίνονται από βάσεις δεδομένων έως συστήματα ηλεκτρονικού ταχυδρομείου. Οι χρήστες δεν χρειάζεται να εγκαθιστούν χειροκίνητα υπηρεσίες και να τις ρυθμίζουν, επειδή το Heroku το κάνει ανώδυνα μέσω πρόσθετων.

4- Απλή κλιμάκωση

Η οριζόντια και κάθετη κλιμάκωση σε μια περίπτωση στο Heroku είναι τόσο απλή όσο η αύξηση ή η μείωση του αριθμού των διαθέσιμων dynos για την εν λόγω περίπτωση. Οι χρήστες μπορούν να εκτελέσουν αυτή την ενέργεια μέσω του CLI (Command Line Interface) ή του web UI (User Interface) του Heroku.

5- Αφιερωμένοι διακομιστές

Οι αποκλειστικοί διακομιστές του Heroku σας επιτρέπουν να δημιουργήσετε εξαρτήσεις εφαρμογών που διασφαλίζουν ότι δεν θα έχετε προβλήματα όπως σφάλματα "εκτός μνήμης" κατά την ανάπτυξη της εφαρμογής σας.

6- Πολλά χρόνια εμπειρίας

Η Heroku είναι ένας από τους παλαιότερους παρόχους PaaS- η πλατφόρμα έχει αναπτυχθεί και εξελιχθεί με την πάροδο των ετών. Υπάρχουν επίσης πολλά άρθρα, οδηγοί και σεμινάρια στο Heroku για αρχάριους και προχωρημένους χρήστες.

7- Δωρεάν βαθμίδα

Η Heroku προσφέρει μια δωρεάν βαθμίδα με μία μόνο περίπτωση dyno, 512MB μνήμης και δύο τύπους διεργασιών, ενώ η εφαρμογή κοιμάται μετά από 30 λεπτά αδράνειας.

8- Προσφέρει τυπική SQL

Η Heroku προσφέρει μια τυπική SQL που δεν συναντάται συνήθως σε άλλους παρόχους υπηρεσιών PaaS.

3.4.5 Μειονεκτήματα Heroku

1- Έντονο κόστος

Μόλις φύγετε από τη δωρεάν βαθμίδα, το Heroku αρχίζει να γίνεται ακριβό. Δεν είναι μόνο η βασική υπηρεσία Heroku που είναι δαπανηρή, αλλά και τα πρόσθετα είναι επίσης πολύ ακριβά.

2- Όχι τόσο μεγάλη όσο η Google και η AWS

Παρόλο που η Heroku υπάρχει εδώ και πολύ καιρό, δεν είναι τόσο μεγάλη όσο η Google και οι άλλοι ανταγωνιστές της.

3- Περιορισμός πρόσθετων

Υπάρχουν περιορισμοί στην προσαρμογή του περιβάλλοντος παραγωγής σας στο Heroku, επειδή οι βιβλιοθήκες ή οι υπηρεσίες μπορούν να εγκατασταθούν μόνο μέσω του πρόσθετου Heroku. Η χρήση μιας υπηρεσίας χωρίς πρόσθετο Heroku θα είναι σχεδόν αδύνατη.

4- Οι επακόλουθες αναπτύξεις είναι αργές

Ενώ η εκκίνηση μιας εφαρμογής στο Heroku είναι γρήγορη και εύκολη και οι πρώτες αναπτύξεις είναι εξίσου γρήγορες, οι μεγαλύτερες εφαρμογές τείνουν να αναπτύσσονται πολύ αργά. Χρειάζεται κάποιο χρονικό διάστημα για την επανεκκίνηση των δυναμόμετρων- κατά τη διάρκεια αυτής της περιόδου, η εφαρμογή τίθεται εκτός λειτουργίας. Αυτό σημαίνει ότι χάνονται μερικά δευτερόλεπτα χρόνου εκτέλεσης της εφαρμογής.

3.4.6 Γιατί το Heroku είναι κατάλληλο για επιχειρήσεις που μόλις ξεκινούν

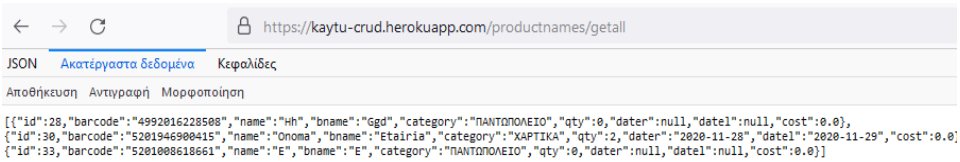
Με το Heroku, οι προγραμματιστές νεοσύστατων επιχειρήσεων έχουν πρόσβαση σε μια cloud-based πλατφόρμα ως υπηρεσία (PaaS) για να σχεδιάζουν, να αναπτύσσουν, να αναπτύσσουν και να εκτελούν εφαρμογές. Δεν χρειάζεται να ανησυχείτε για διακομιστές, υλικό ή υποδομές. Με αυτή την πλήρως διαχειρίσιμη πλατφόρμα μπορείτε να επικεντρωθείτε στην κατασκευή ενός εξαιρετικού προϊόντος. Με απρόσκοπτη επεκτασιμότητα και προηγμένα χαρακτηριστικά, η Heroku υποστηρίζει την επιτυχία των εφαρμογών σας μακροπρόθεσμα, καθώς η επιχείρησή σας αναπτύσσεται.

3.4.7 Τα Heroku Dynos διευκολύνουν την εύκολη ανάπτυξη και βελτιώνουν τη χρηστικότητα

Μια εφαρμογή Heroku συνδέεται συνήθως με ένα όνομα τομέα που χρησιμοποιείται για τη δρομολόγηση των αιτήσεων HTTP στον κατάλληλο περιέκτη. Τα κοντέινερ εφαρμογών χρησιμοποιούνται για τη φιλοξενία εφαρμογών Heroku. Ένα εμπορευματοκιβώτιο εκτελεί μια υπηρεσία συσκευασμένη σε ένα μικρό πακέτο. Οι εφαρμογές σε εμπορευματοκιβώτια εκτελούνται σε αξιόπιστα, διαχειριζόμενα περιβάλλοντα χρόνου εκτέλεσης που υποστηρίζονται από έξυπνα εμπορευματοκιβώτια. Η πλατφόρμα Heroku αναπτύσσει κοντέινερ εφαρμογών σε ένα "πλέγμα δυναμοποίησης" Πρόκειται για μια συλλογή διακομιστών και τα Dynos διαχειρίζονται και λειτουργούν από τον διαχειριστή Dyno. Λόγω της δυνατότητας του Heroku να διαχειρίζεται και να εκτελεί εφαρμογές, δεν είναι απαραίτητο να χειρίζεται λειτουργικά συστήματα ή άλλες εσωτερικές δυνατότητες διαμόρφωσης.

3.4.8 Εγκατάσταση του REST API στο Heroku

Εφόσον κάνουμε είσοδο στον λογαριασμό μας , μεταβαίνοντας στη καρτέλα Deploy μπορούμε να εγκαταστήσουμε την εργασία μας μέσω Github. Μπορούμε να επιλέξουμε την επιλογή που κάθε φορά το Heroku θα παρατηρήσει αλλαγές στο GitHub , να αναπτύξει ξανά το Project μας στην πλατφόρμα. Πλέον μπορούμε να στέλνουμε αιτήματα στο server μας με την καινούρια διεύθυνση https://kaytu-crud.herokuapp.com/productnames/getall .



Εικόνα 7 Αίτημα προς το Heroku Server.

4. Android Εφαρμογή

4.1 Android Studio

Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για την ανάπτυξη εφαρμογών Android, βασισμένο στο IntelliJ IDEA . Εκτός από τον ισχυρό επεξεργαστή κώδικα και τα εργαλεία ανάπτυξης του IntelliJ, το Android Studio προσφέρει ακόμη περισσότερες δυνατότητες που ενισχύουν την παραγωγικότητά σας κατά την κατασκευή εφαρμογών Android, όπως:

1. Ένα ευέλικτο σύστημα κατασκευής βασισμένο στο Gradle
2. Ένας γρήγορος και πλούσιος σε χαρακτηριστικά εξομοιωτής
3. Ένα ενοποιημένο περιβάλλον όπου μπορείτε να αναπτύξετε για όλες τις συσκευές Android
4. Εφαρμογή αλλαγών για να προωθήσετε αλλαγές κώδικα και πόρων στην εφαρμογή που εκτελείται χωρίς επανεκκίνηση της εφαρμογής σας
5. Πρότυπα κώδικα και ενσωμάτωση στο GitHub για να σας βοηθήσουν να δημιουργήσετε κοινά χαρακτηριστικά της εφαρμογής και να εισάγετε δείγματα κώδικα
6. Εκτεταμένα εργαλεία και πλαίσια δοκιμών
7. Εργαλεία Lint για να εντοπίζετε προβλήματα απόδοσης, ευχρηστίας, συμβατότητας εκδόσεων και άλλα προβλήματα
8. Υποστήριξη C++ και NDK
9. Ενσωματωμένη υποστήριξη για το Google Cloud Platform, που καθιστά εύκολη την ενσωμάτωση του Google Cloud Messaging και του App Engine

4.2 Γλώσσες προγραμματισμού για ανάπτυξη Android Εφαρμογή

4.2.1 Java

Η επίσημη γλώσσα για την ανάπτυξη του Android είναι η Java. Μεγάλα τμήματα του Android είναι γραμμένα σε Java και τα API του έχουν σχεδιαστεί για να καλούνται κυρίως από τη Java. Είναι δυνατή η ανάπτυξη εφαρμογών σε C και C++ χρησιμοποιώντας το Android Native Development Kit (NDK), ωστόσο δεν είναι κάτι που προωθεί η Google. Σύμφωνα με την Google, "το NDK δεν θα ωφελήσει τις περισσότερες εφαρμογές. Ως προγραμματιστής, θα πρέπει να εξισορροπήσετε τα πλεονεκτήματά του έναντι των μειονεκτημάτων του. Ειδικότερα, η χρήση εγγενούς κώδικα στο Android γενικά δεν οδηγεί σε αξιοσημείωτη βελτίωση των επιδόσεων, αλλά πάντα αυξάνει την πολυπλοκότητα της εφαρμογής σας."

4.2.2 Kotlin

Αν και σχετικά νέα στον κόσμο του Android, η Kotlin αποτελεί μια φανταστική επιλογή για τους προγραμματιστές εφαρμογών Android. Αυτή η γλώσσα ανοικτού κώδικα γενικού σκοπού σχεδιάστηκε με γνώμονα την πρακτικότητα, την αποδοτικότητα και την αποτελεσματικότητα. Η Kotlin είναι πλήρως συμβατή με τη Java και, ως εκ τούτου, έχει γίνει γρήγορα η δεύτερη επίσημη γλώσσα προγραμματισμού για το Android. Η γλώσσα σχεδιάστηκε με στόχο να αποτελέσει μια πιο σύγχρονη, αποδοτική και πρακτική εναλλακτική της Java. Ως εκ τούτου, διαθέτει πιο λογική διάταξη και μεγαλύτερη αναγνωσιμότητα σε σύγκριση με τον ανταγωνιστή του. Η Java εξακολουθεί να έχει ευρύτερη βάση χρηστών και είναι αναμφισβήτητα ταχύτερη σε σύγκριση με την Kotlin, αλλά η Kotlin έχει ευκολότερη πληκτρολόγηση, συντομότερο κώδικα και περισσότερα χαρακτηριστικά ασφαλείας.

4.2.3

4.2.4 C++

Αν και η C++ δεν είναι η πρώτη γλώσσα που έρχεται στο μυαλό των αρχάριων, είναι μια εξαιρετική επιλογή για την κατασκευή πιο σύνθετων εφαρμογών Android. Ως μία από τις πιο δημοφιλείς γλώσσες προγραμματισμού στον κόσμο, η C++ χρησιμοποιείται για οτιδήποτε, από το σχεδιασμό βιντεοπαιχνιδιών μέχρι την κατασκευή λειτουργικών συστημάτων και την παραγωγή τρισδιάστατων ταινιών. Η C++ εκτελείται εγγενώς στο τηλέφωνο - και ενώ αυτό μπορεί να κάνει πιο δύσκολη τη χρήση της, η γλώσσα είναι εξαιρετικά ευέλικτη όσον αφορά τα χαρακτηριστικά της. Παρά την μάλλον απότομη καμπύλη εκμάθησης, η C++ είναι πραγματικά εξαιρετική για το σχεδιασμό πολύπλοκων παιχνιδιών λόγω της εγγενούς πλατφόρμας της.

4.2.5 C#

Συνδυάζοντας τα καλύτερα χαρακτηριστικά της C++ και της Java, η C# μπορεί να χρησιμοποιηθεί για ένα ευρύ φάσμα εργασιών, από το σχεδιασμό παιχνιδιών με τη χρήση της Unity έως την ανάπτυξη σύνθετων εφαρμογών Windows. Η C# είναι ένα αντικειμενοστραφές και υψηλότερου επιπέδου μέλος της οικογένειας γλωσσών C και αποτελεί μια καταπληκτική επιλογή για την ανάπτυξη Android. Η C# αναπτύχθηκε από τη Microsoft το 2000 και, ενώ είναι αρκετά νεότερη από τις περισσότερες άλλες γλώσσες προγραμματισμού, έχει αποκτήσει έκτοτε μια μεγάλη κοινότητα προγραμματιστών. Η γλώσσα έχει μια λογική καμπύλη εκμάθησης, απλούστερη σύνταξη από πολλές άλλες γλώσσες και μπορεί να χρησιμοποιηθεί για διάφορους σκοπούς, γεγονός που την καθιστά μια εξαιρετική επιλογή για γλώσσα προγραμματισμού Android.

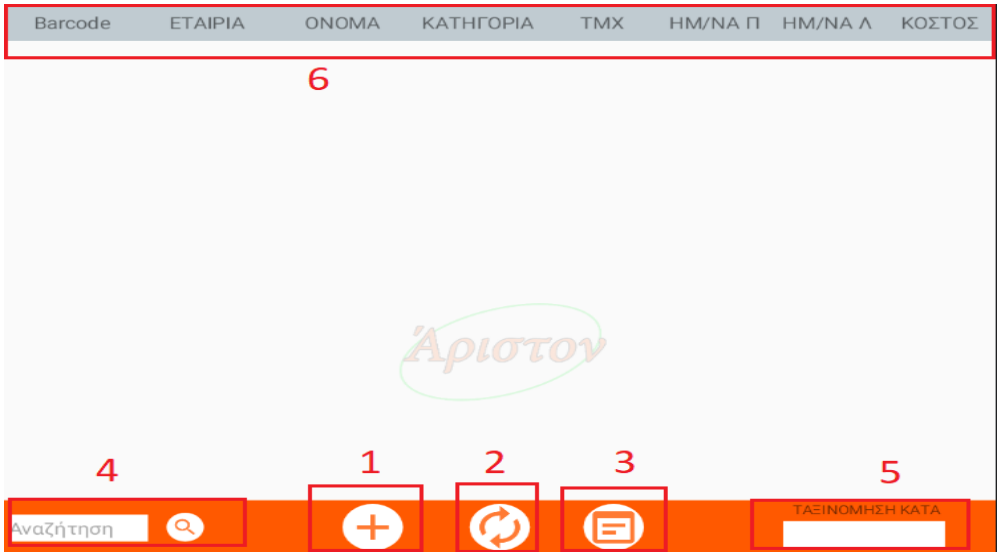
4.3 Retrofit

Το Retrofit είναι ένα REST client με ασφάλεια τύπου για Android και Java, ο οποίος έχει ως στόχο να διευκολύνει την κατανάλωση RESTful υπηρεσιών ιστού. Το Retrofit από προεπιλογή αξιοποιεί το OkHttp ως επίπεδο δικτύωσης και βασίζεται σε αυτό. Στο Android, η βιβλιοθήκη Retrofit διαφέρει από άλλες βιβλιοθήκες δικτύου επειδή μας παρέχει μια εύχρηστη πλατφόρμα μέσω της οποίας δεν χρειάζεται να αναλύουμε τις απαντήσεις JSON, καθώς αυτές γίνονται από την ίδια τη βιβλιοθήκη. Χρησιμοποίησε τη βιβλιοθήκη GSON στο παρασκήνιο για την ανάλυση των δεδομένων απόκρισης. Αυτό που πρέπει να κάνουμε είναι να ορίσουμε ένα POJO (Plain Old Java Object) για την ανάλυση της απόκρισης. [10]

4.4 Εφαρμογή Android

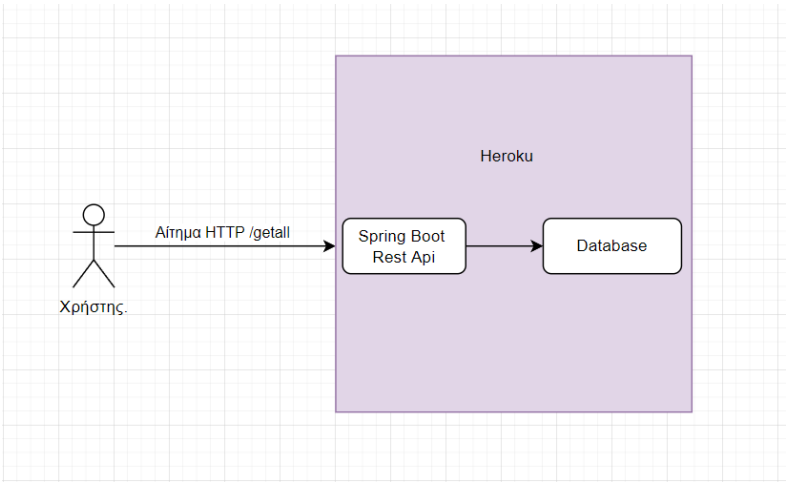
4.4.1 Βασικές λειτουργίες και ανάλυση Main Activity

Στην αρχική σελίδα, μας καλωσορίζει το κύριο μενού όπου έχει το λογότυπο και τα χρώματα του καταστήματος. Εκτός από τα χρώματα και τον λογότυπο, η εφαρμογή στέλνει αίτημα GET με HTTP μέθοδο /getall που καλεί τα προϊόντα όπου η ποσότητα τους είναι μεγαλύτερο από μηδέν. Παρακάτω θα γίνει ανάλυση τον λειτουργιών όπου εμφανίζεται στην εικόνα



Εικόνα 8 Αρχική οθόνη της εφαρμογής

1. Το συγκεκριμένο κουμπί θα μας μεταφέρει στην εισαγωγή καινούριου προϊόντος στην βάση δεδομένων
2. Ανανεώνουμε την λίστα με προϊόντα χειροκίνητα για τυχόν αλλαγές
3. Μας επιστρέφει όλα τα προϊόντα που είναι ίσων με μηδέν.
4. Κάνουμε αναζήτηση προϊόν με χρήση Barcode. Αν το πεδίο είναι άδειο ανοίγει η κάμερα για σάρωση Barcode.
5. Ταξινομεί την λίστα ανάλογα με την στήλη που θέλουμε
6. Μπάρα όπου αντιστοιχείτε κάθε στήλη.



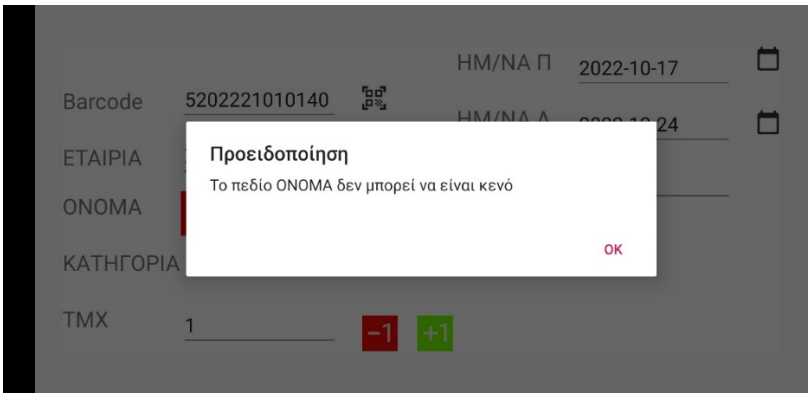
Διάγραμμα 1. 4 Αίτημα /getall από την εφαρμογή

4.4.2 Δημιουργία καινούριο προϊόντος

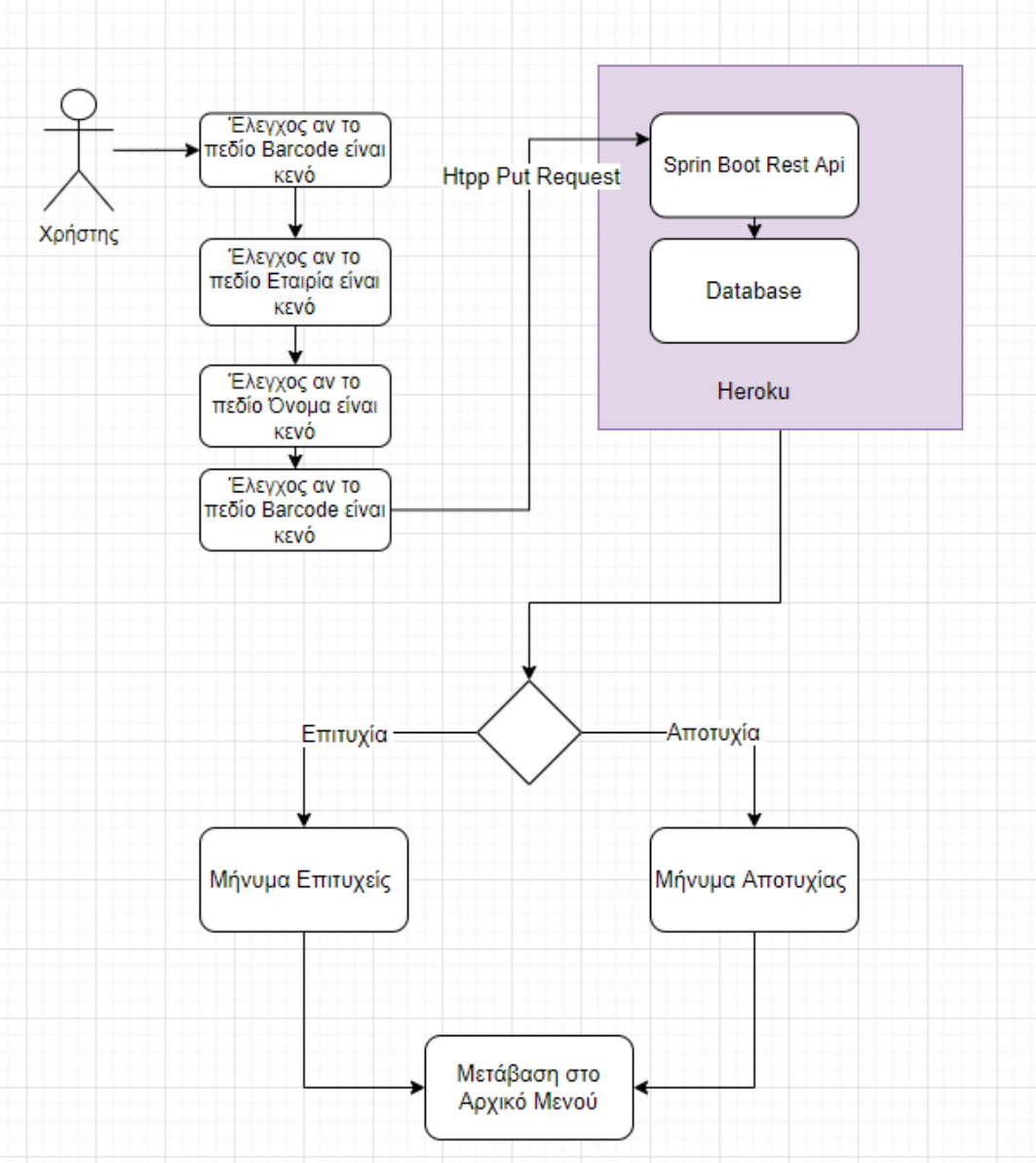
Για την δημιουργία καινούργιου προϊόν, πρέπει να πατήσουμε το κουμπί (+) που που εμφανίζεται στο μενού μας. Στη συνέχεια, μας ανοίγει το παράθυρο όπου γίνεται καταχώρηση καινούριου προϊόντος.

Εικόνα 9 Δημιουργία προϊόν

Συμπληρώνοντας τα παρακάτω πεδία , μπορούμε να καταχωρήσουμε το προϊόν στο σύστημα μας. Σε περίπτωση που έχουμε κάνει κάποιο λάθος σε κάποιο πεδίο , η εφαρμογή μας προειδοποιεί και δεν μας αφήνει να προχωρήσουμε παρακάτω. Τα πεδία όπως Barcode, Εταιρία, Όνομα δεν πρέπει να μείνουν κενά.



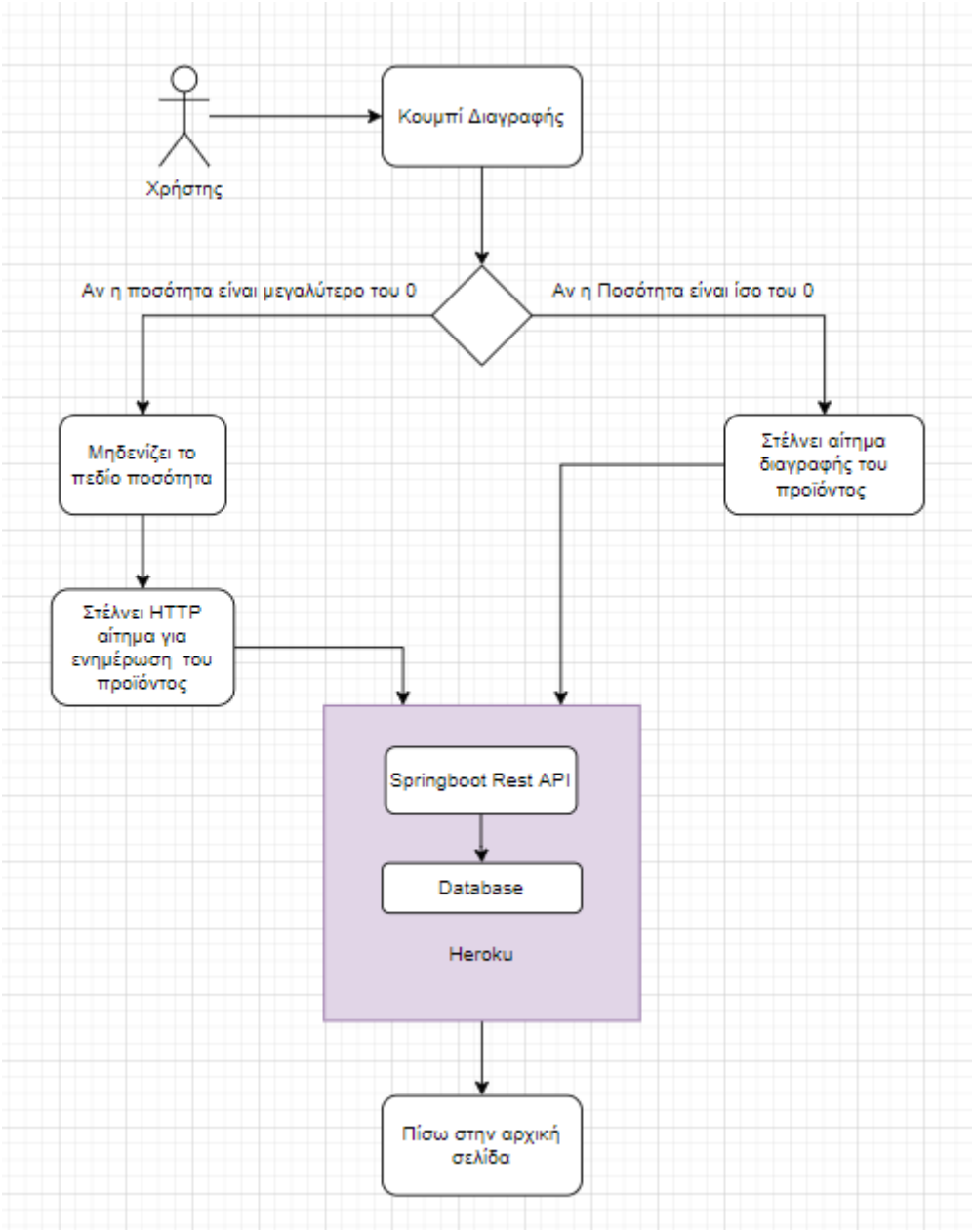
Εικόνα 10 Προειδοποίηση της εφαρμογής



Διάγραμμα 1. 5 Δημιουργία καινούργιου προϊόντος

4.4.3 Διαγραφή προϊόντος

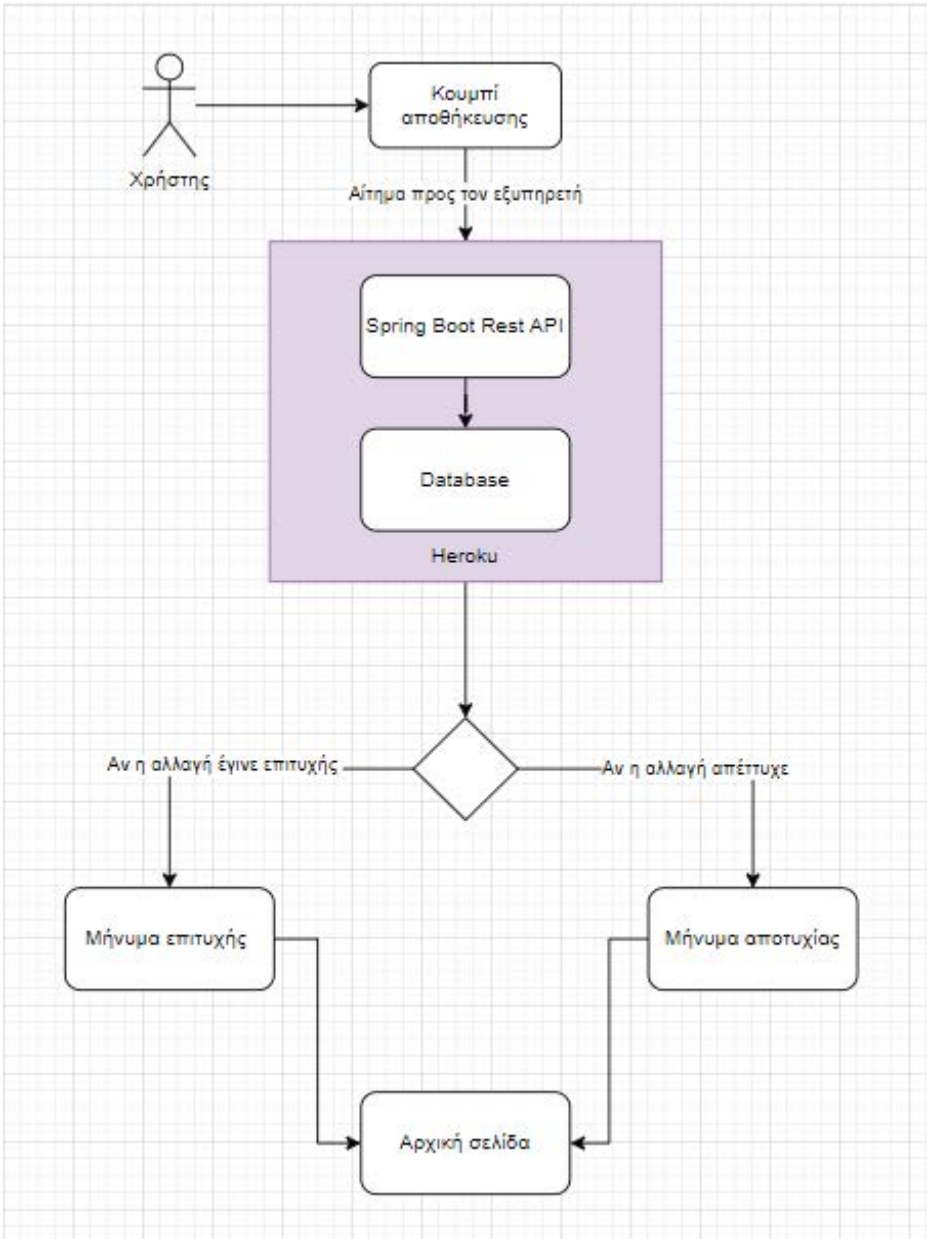
Στην διαγραφή του προϊόντος η εφαρμογή μας εξετάζει 2 ενδεχόμενα. Η πρώτη περίπτωση είναι αν το προϊόν έχει ποσότητα πάνω από 0 όπου πατώντας το κουμπί διαγραφή μηδενίζει την ποσότητα και το αποθηκεύει το προϊόν και η δεύτερη περίπτωση είναι αν η ποσότητα ισούται με το 0 , τότε διαγραφή το προϊόν. Ο στόχος είναι το προϊόν να μην διαγραφθεί με πρώτη φορά και να μπει στη λίστα με τα προϊόντα που είναι για παραγγελία. Όπου εκεί ο χρήστης αποφασίζει αν όντως θέλει να διαγράψει το προϊόν οριστικά ή να κάνει παραγγελία και όταν θα έχει απόθεμα να αυξήσει την ποσότητα του προϊόν.



Διάγραμμα 1. 6 Διαγραφή προϊόντος

4.4.4 Επεξεργασία προϊόντος

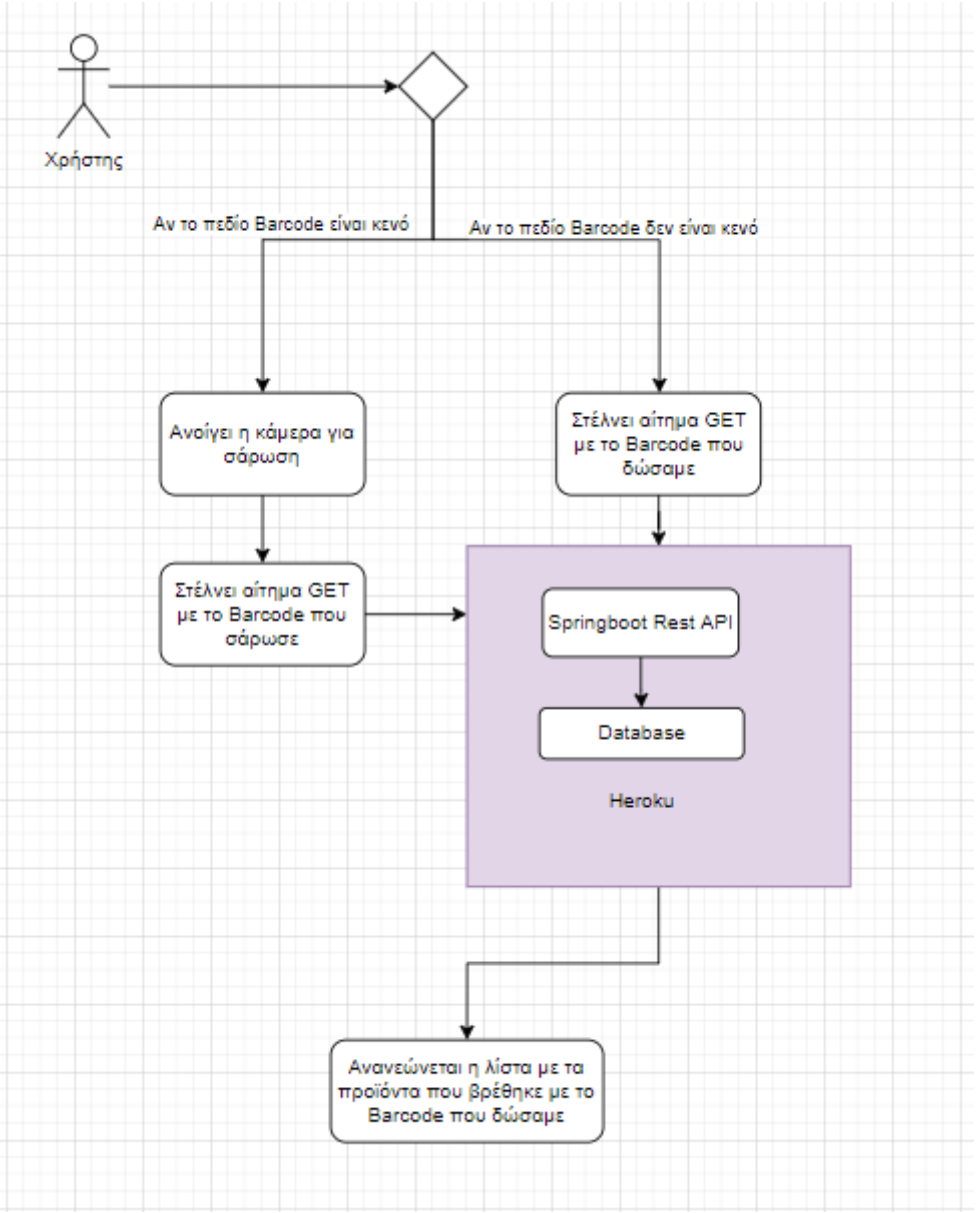
Η Επεξεργασία του προϊόντος δεν απαιτεί κάποια ιδιαίτερη διαδικασία πάρα μόνο να επιλέξουμε το προϊόν που θέλουμε να επεξεργαστούμε και να κάνουμε τις αλλαγές που επιθυμούμαστε. Πατώντας το κουμπί αποθήκευση , η εφαρμογή στέλνει αίτημα προς το server με το id του προϊόντος για αλλαγή. Ενημερωνόμαστε με το μήνυμα αν η αλλαγή έγινε επιτυχώς ή όχι και μεταφερόμαστε στην αρχική σελίδα.



Διάγραμμα 1. 7 Επεξεργασία προϊόντος

4.4.5 Αναζήτηση προϊόντος με Barcode

Η Αναζήτηση του προϊόντος χρησιμοποιώντας το Barcode γίνεται με 2 τρόπους. Είτε πληκτρολογώντας τον Barcode είτε χρησιμοποιώντας την κάμερα του κινητού μας για σάρωση. Για να χρησιμοποιήσουμε την κάμερα του κινητού μας, ο χρήστης πρέπει να δώσει δικαίωμα στην εφαρμογή. Πατώντας το κουμπί αναζήτησης αν το πεδίο Barcode είναι άδειο, αυτομάτως ανοίγει η κάμερα μας για σάρωση. Αν όμως το πεδίο έχει κάποιο αριθμό τότε η εφαρμογή στέλνει αίτημα προς τον εξυπηρετεί για να φέρει τα προϊόντα με Barcode που θέλουμε.



Διάγραμμα 1. 8 Αναζήτηση με Barcode

5. Βάση Δεδομένων

5.1 Αρχιτεκτονική της Βάσης

Η Βάση δεδομένων της εφαρμογής αποτελείτε από έναν πίνακα. Το όποιο αυτός ο πίνακας δημιουργείται αυτόματα από το Spring Boot. Ορίζοντας στην κλάση Entity /User.java το όνομα του πίνακας και της στήλες που επιθυμούμε και στο application.properties το spring.jpa.hibernate.ddl-auto= σε Create, ο Server θα δημιουργήσει αυτόματα τον πίνακα και της στήλες. Σε περίπτωση αν υπάρχει πίνακα και το γνωρίζουμε, καλό θα ήταν το spring.jpa.hibernate.ddl-auto να το ορίσουμε σε update έτσι ώστε να μην δημιουργεί κάθε φορά καινούριο πίνακα και να κάνει μόνο Update. Ο λόγος που δημιουργήθηκε μόνο ένας πίνακας και δεν έχουμε σχεσιακό μοντέλο είναι η αποφυγή από την πολυπλοκότητα και ότι η εφαρμογή δεν απαιτούσε να υπάρχει το σχεσιακό μοντέλο από πίσω. Τρέχοντας το Server με πληροφορίες που ορίσαμε θα απεκτήσουμε τον πίνακα product_names και της παρακάτω στήλες.

Column Name	Datatype
Id	INT(11) Primary Key
Barcode	Varchar(255)
Bname	Varchar(255)
Category	Varchar(255)
Cost	Double
Dateel	Date
Dater	Date
Name	Varchar(255)
Qty	INT(11)

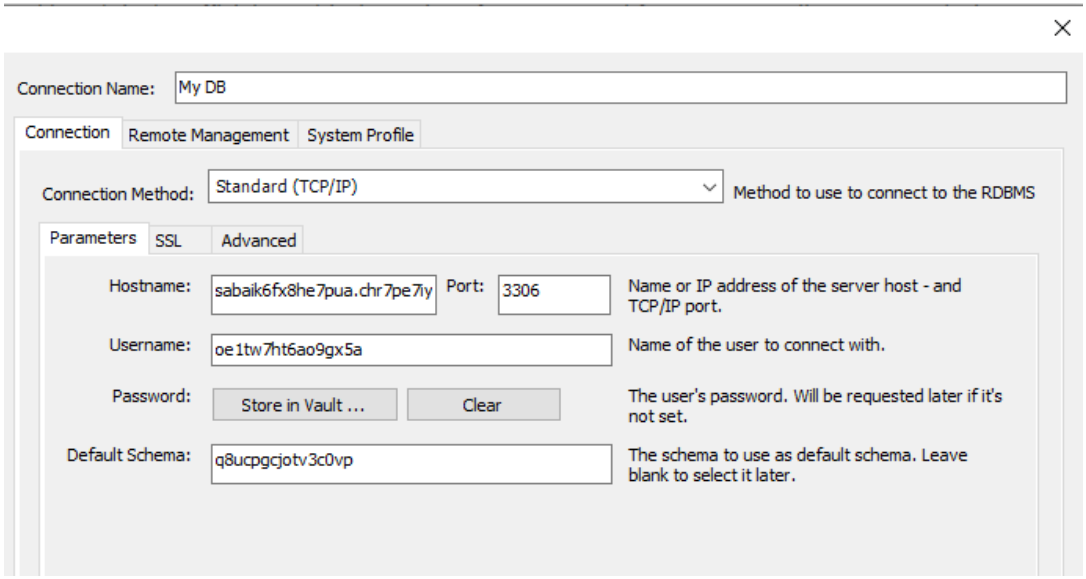
Πίνακας 1. 2 Στήλες του Πίνακα

5.2 Πρόσβαση στην Βάση δεδομένων

Η βάση μας είναι τύπου Mysql και φιλοξενείτε δωρεάν από το Heroku. Για να μπορέσουμε να συνδεθούμε θα χρειαστούμε να κατεβάσουμε την εφαρμογή Mysql Workbench και τα Credentials που μας έχει δώσει το Heroku.

Host URL	`sabaik6fx8he7pua.chr7pe7iynqr.eu-west-1.rds.amazonaws.com`
Port	`3306`
Username	`oe1tw7ht6ao9gx5a`
Password	`xk9a2c321fcrxb0b`
Db Scheme	`q8ucpgcjotv3c0vp`

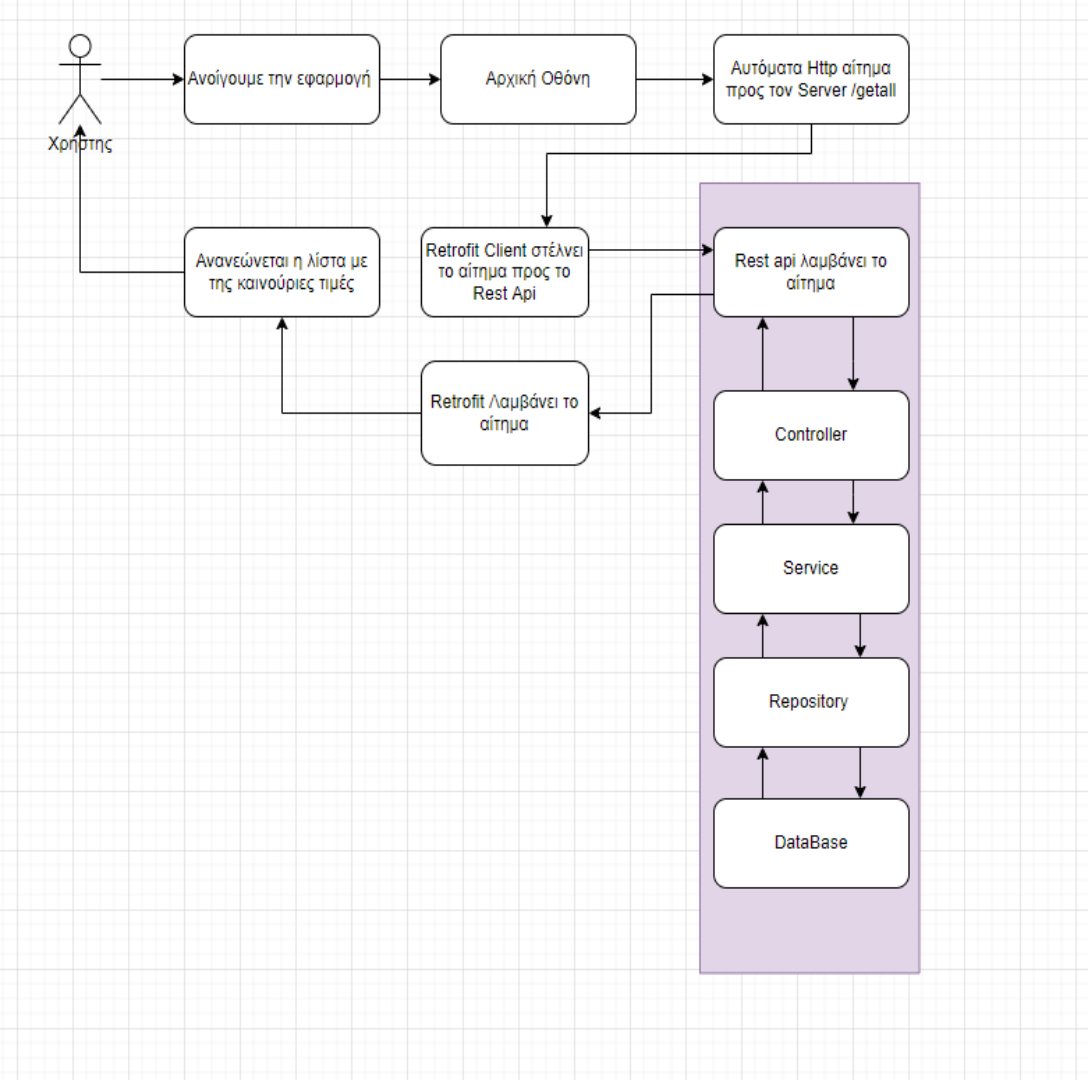
Συμπληρώνοντας τα στοιχεία που μας έχει δώσει το Heroku το Workbench θα είναι έτσι



Εικόνα 11 Είσοδος στην βάση μέσω Mysql

6. Διαγράμματα Ευρέως

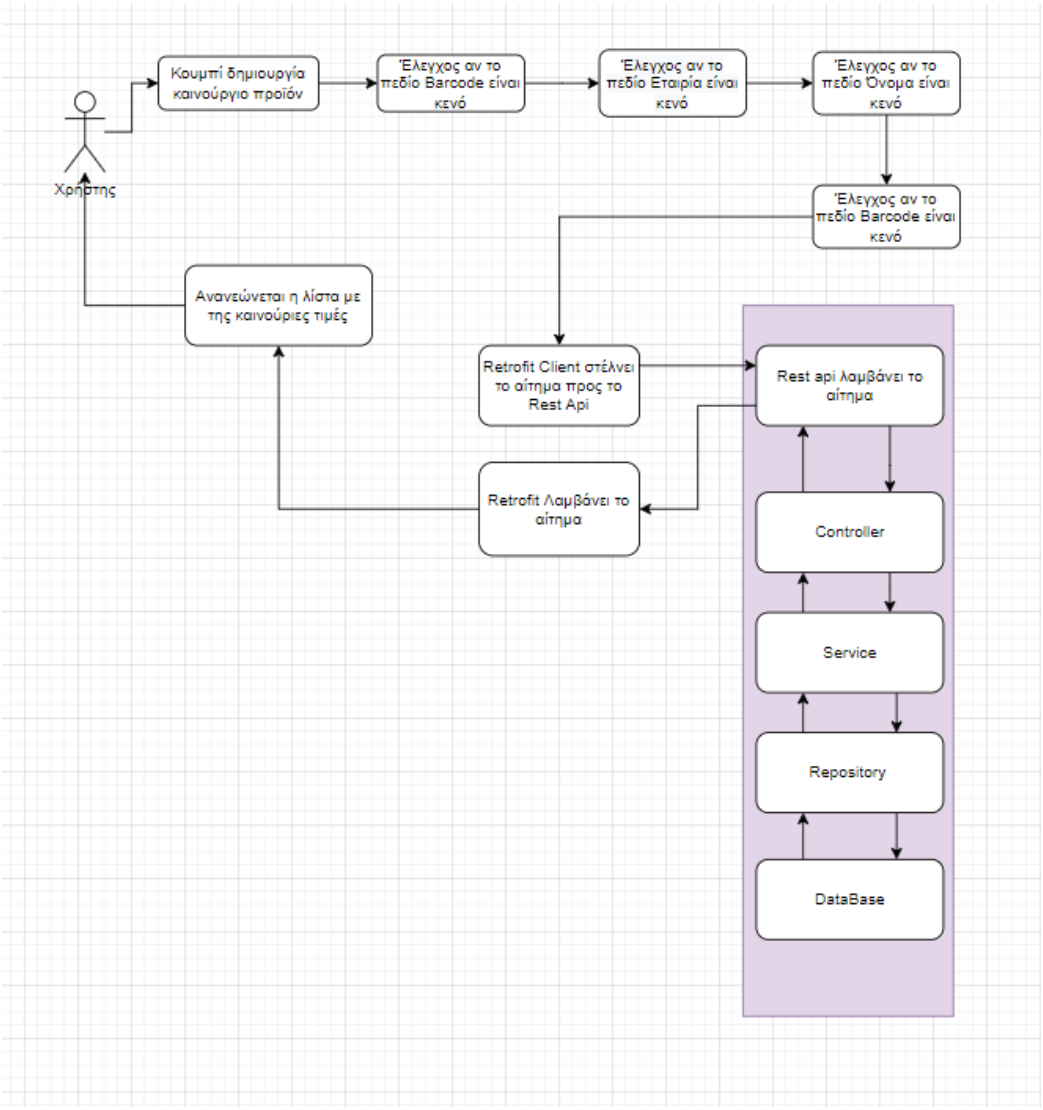
6.1 Ανάλυση Είσοδος στην εφαρμογή



Διάγραμμα 1. 9 Ανάλυση Είσοδος στην εφαρμογή

Κατά την διάρκεια που ανοίγει η εφαρμογή μας , η εφαρμογή στέλνει αυτόματα το αίτημα /getall προς τον Rest Api μέσω Retrofit έτσι ώστε στην αρχική σελίδα μας να δούμε τα προϊόντα που η ποσότητα τους είναι ίσω ή μεγαλύτερο του 1.

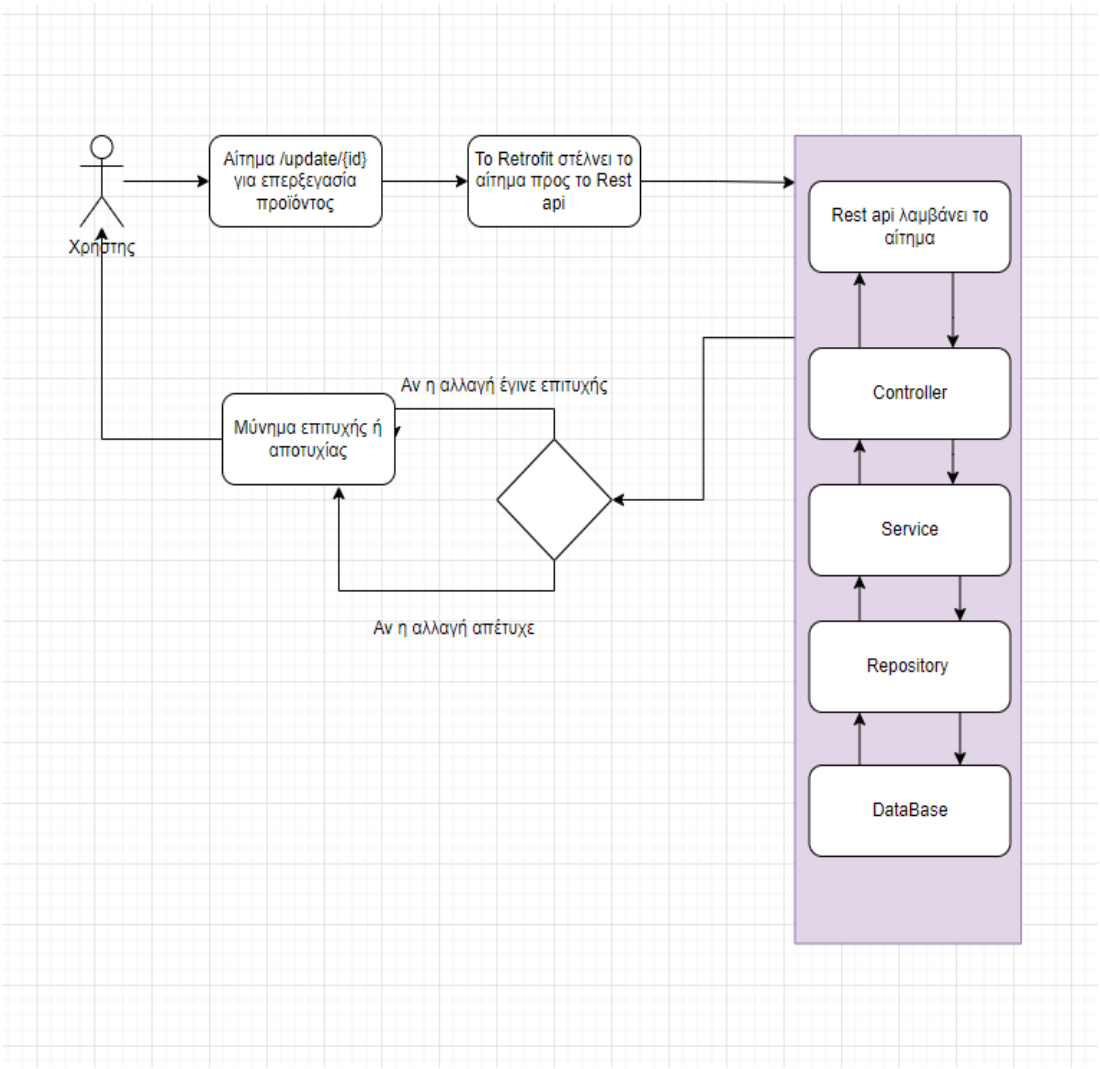
6.2 Ανάλυση Δημιουργία Προϊόντος



Διάγραμμα 1. 10 Ανάλυση Δημιουργία Προϊόντος

Παρατηρούμε ότι πριν σταλεί το αίτημα /put προς τον server μέσω retrofit, τα στοιχεία που καταχώρησε ο χρήστης περνάει από φίλτρα έτσι ώστε να εξασφαλίσουμε κάποια πεδία να μην καταχωρηθούν κενά στην βάση.

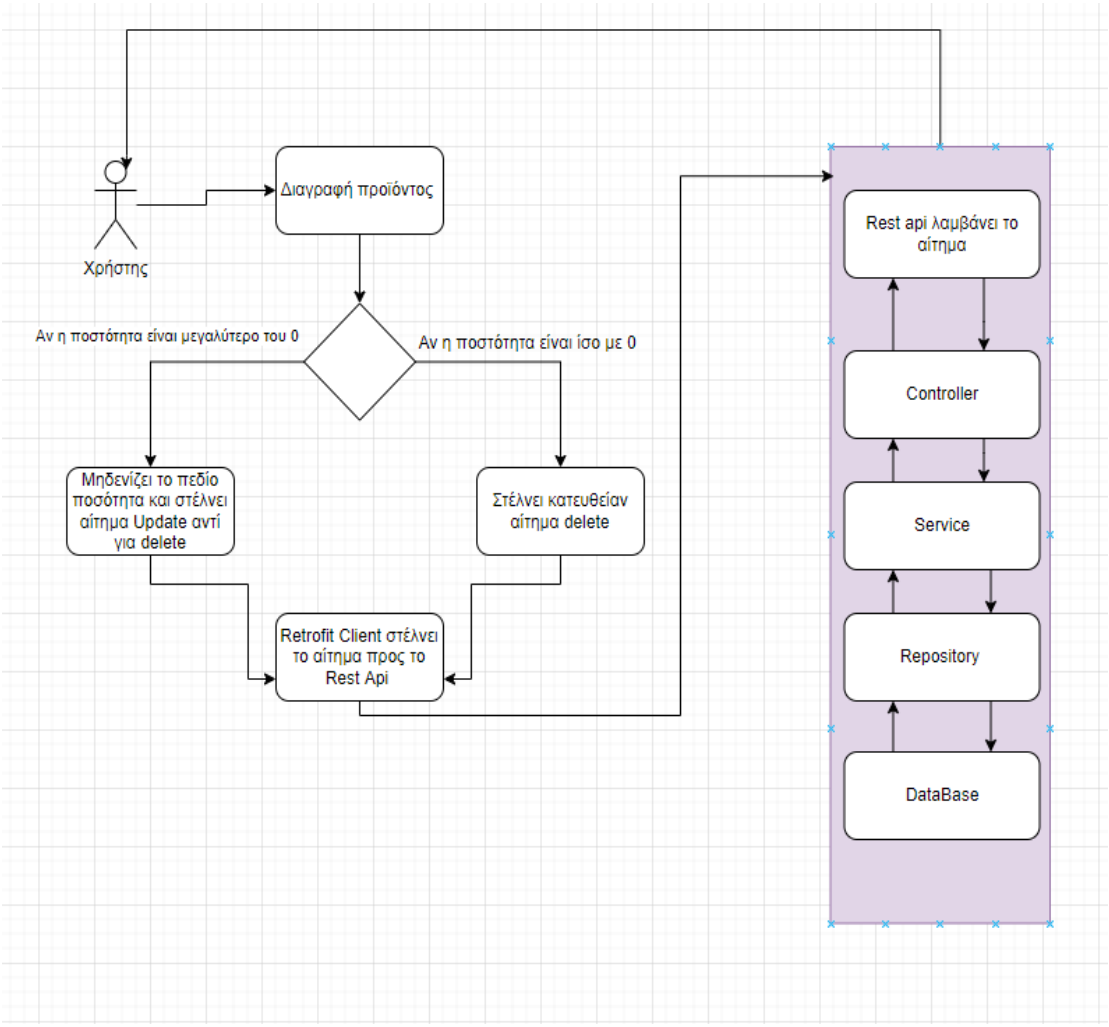
6.3 Ανάλυση Επεξεργασία Προϊόντος



Διάγραμμα 1. 11 Ανάλυση Επεξεργασία Προϊόντος

Βλέποντας το διάγραμμα καταλαβαίνουμε πως υπάρχει μια κρυφή στήλη που είναι το id του κάθε προϊόντος. Είναι μοναδικό επίσης είναι και το Primary Key στην βάση δεδομένων. Χρησιμοποιώντας το id του προϊόντος που είναι μοναδικό αποφεύγουμε τα διπλότυπα και κάθε φορά αλλάζουμε το σωστό προϊόν.

6.4 Ανάλυση Διαγραφή Προϊόντος



Διάγραμμα 1. 12 Ανάλυση Διαγραφή Προϊόντος

Ο λόγος που δεν διαγράφετε κατευθείαν το προϊόν είναι να δώσει στον χρήστη την δυνατότητα να ελέγχει ποια προϊόντα έχουν μηδενιστεί και να ξέρει ποια προϊόντα είναι προς παραγγελία. Έπειτα αν αποφασίσει να το διαγράψει επειδή δεν το θέλει για κάποιο λόγο , τότε το προϊόν διαγράφετε κατευθείαν από την βάση.

Συμπεράσματα

Η ανάγκη προς τα συστήματα διαχείρισης αποθήκης έχει αυξηθεί πάρα πολύ και οι επιχειρήσεις το χρειάζονται περισσότερο από ποτέ. Η εφαρμογή μας κατασκευάστηκε αποκλειστικά για το μαγαζί Άριστον. Για μια πιο γενική εφαρμογή θα χρειαστεί να ανακατασκευαστεί ξανά από την αρχή . Χρησιμοποιώντας βάση δεδομένων τύπου σχεσιακό μοντέλο, θα μας έδινε την δυνατότητα να δώσουμε πρόσβαση στον χρήστη να επεξεργαστή παραπάνω δεδομένα. Όπως εμπορικές κατηγορίες, Προμηθευτές, Ομάδες, Λογιστικές κατηγορίες, μέχρι και εναλλακτικούς (επιπλέον) κωδικούς προς σάρωση ειδών. Επίσης θα μπορούσαμε να έχουμε και άλλες δυνατότητες όπως ο χρήστης να αλλάζει τα χρώματα της εφαρμογής και να αλλάζει το λογότυπο που είναι στην αρχική σελίδα.

Με τα καινούρια δεδομένα στην χώρα μας (mydata), πλέον οι επιχειρήσεις απευθύνονται στα προγράμματα εμπορικής διαχείρισης(ERP) αντί για σύστημα διαχείρισης αποθεμάτων(WMS). Διότι, στα εμπορικά προγράμματα περιέχει και το σύστημα διαχείρισης αποθεμάτων και δίνει την δυνατότητα να κόβουμε παραστατικά και να τα διαβιβάζουμε στην ΑΑΔΕ. Είναι πολύ δύσκολο πλέον για έναν προγραμματιστή να φτιάξει μια πρόγραμμα εμπορικής διαχείρισης και να αντάγωνηστή με μεγάλες εταιρίες. Το Mydata αλλάζει συνεχώς και οι μεγάλες εταιρίες λαμβάνουν τις αλλαγές πιο νωρίς και τα εφαρμόζουν πιο γρήγορα στην εφαρμογή τους. Η εταιρίες όπως Epsilon Net και Softone , είναι οι δύο μεγαλύτερες εταιρείες αυτή την στιγμή στα εμπορικά προγράμματα στην Ελλάδα.

Εκτός από το σύστημα διαχείρισης αποθήκης, χρησιμοποιώντας μια τέτοια αρχιτεκτονική μπορούν να προκύψουν και άλλες εφαρμογές. Όπως σύστημα παραγγελιοληψίας για καφετέριες ή σύστημα διαχείρισης πελατών γυμναστηρίου. Ένα καλό παράδειγμα θα μπορούσε να είναι μια επειχείρειση μπορεί να έχει ανάγκει από μια εφαρμοργή που μπορεί να σαρώνει και να βλέπει αποθέματα από υπάρχον βάση δεδομένων του ERP. Κάνοντας τις κατάληλες αλλαγές, αυτή η εφαρμογή θα μπορούσε να δουλέψει άψογα και να καλύψει όλες τις ανάγκες της επειχείρισης. Με αυτό τον τρόπο η εφαρμογή θα πουληθεί πιο γρήγορα.

Βιβλιογραφία

1. H. D. PAUL DEITEL, JAVA ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ, New Jersey: Μ.Γκιούρδας, 2015.

2. S. Chakraborti, «PROS AND CONS OF JAVA DEVELOPMENT,» Coud Vandana, 2022. [Ηλεκτρονικό]. Available: https://www.bairesdev.com/java/pros-cons/.

3. P. Gupta, «What is Eclipse IDE?,» Educba, [Ηλεκτρονικό]. Available: https://www.educba.com/what-is-eclipse-ide/.

4. R. Shah, «Working the Eclipse Platform,» IBM, Μάρτιος 2003. [Ηλεκτρονικό]. Available: http://www-106.ibm.com/developerworks/opensource/library/os-eclipse.

5. A. R. D. S. Joao Saraiva, «Eclipse.NET: An Integration Platform for ProjectIT-Studio,» *ResearchGate,* p. 14, 2005.

6. T. B. Hamdi, «Java SpringBoot nedir?,» The code Program, [Ηλεκτρονικό]. Available: https://thecodeprogram.com/java-springboot-nedir-.

7. B. Egilmez, «Postman Nedir?,» Medium, 11 Δεκέμβριος 2020. [Ηλεκτρονικό]. Available: https://medium.com/huawei-developers-tr/postman-nedir-83eeaa5ed6ac.

8. «Ceyhun Enki Aksan,» Ceaksan, 14 Ιούλιος 2019. [Ηλεκτρονικό]. Available: https://ceaksan.com/tr/heroku-nedir.

9. Z. L. Y. D. &. L. G. Ling Qian, «Cloud Computing: An Overview,» *IEEE International Conference on Cloud Computing,* p. 6, 2009.

10. G. Batschinski, «What Programming Language is Used for Android App Development?,» back4app, [Ηλεκτρονικό]. Available: https://blog.back4app.com/what-programming-language-is-used-for-android-app-development/#2_Kotlin_Native.

Παράρτημα Κώδικα

6.5 Eclipe – Spring boot

Entity/User.java

```
package com.javahelps.restservice.entity;


import java.sql.Date;


import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;


import org.hibernate.annotations.Proxy;


@Entity
@Table(name = "product_names")
@Proxy(lazy = false)
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;


    @Column(name = "barcode")
    private String barcode;
```

```
    @Column(name = "name")
    private String name;

    @Column(name = "bname")
    private String bname;

    @Column(name = "category")
    private String category;

    @Column(name = "qty")
    private int qty;
    @Column(name = "dater")
    private Date dater;
    @Column(name = "datel")
    private Date datel;
    @Column(name = "cost")
    private double cost;

    public double getCost() {
        return cost;
    }



    public void setCost(double cost) {
        this.cost = cost;

    }

    public int getId() {
```

```
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }
    public String getBarcode() {
        return barcode;
    }

    public void setBarcode(String barcode) {
        this.barcode = barcode;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getBname() {
        return bname;
    }

    public void setBname(String bname) {
        this.bname = bname;
    }
    public String getCategory() {
```

```
        return category;
    }

    public void setCategory(String category) {
        this.category = category;
    }

    public int getQty() {
        return qty;
    }

    public void setQty(int qty) {
        this.qty = qty;
    }

    public Date getDater() {
        return dater;
    }

    public void setDater(Date dater) {
        this.dater = dater;

    }

    public Date getDatel() {
        return datel;
    }

    public void setDatel(Date datel) {
        this.datel = datel;
```

```
        }



        @Override
        public String toString() {
            return "User{" + "='" + ", id='" + id + '\'' + ", name='" +
barcode + '\'' + ", name='" + name + '\'' + "bname='\" + bname + '\\'' +
\"cost='\\\" + cost + '\\\\'' +category='" + category + '\'' +
                ", qty='" + qty + '\'' + ", dater='" + dater + '\'' + ",
datel='" + datel + '\'' + '}';
        }


    }
```

resources/application.properties

```
    # Automatically update the database
    spring.jpa.hibernate.ddl-auto=update


    # The database connection URL
    spring.datasource.url=jdbc:mysql://sabaik6fx8he7pua.chr7pe7iynqr.eu-
west-1.rds.amazonaws.com:3306/q8ucpgcjotv3c0vp?useSSL=false



    # Username
    spring.datasource.username=oe1tw7ht6ao9gx5a


    # Password
    spring.datasource.password=xk9a2c321fcrxb0b


    spring.jta.bitronix.connectionfactory.max-pool-size=8
```

```
   # Define the database platform

       spring.jpa.properties.hibernate.dialect                          =
org.hibernate.dialect.MySQL5InnoDBDialect



   # Define the naming strategy

   spring.jpa.hibernate.naming-strategy                                 =
org.hibernate.cfg.ImprovedNamingStrategy



   # Define the default schema

   spring.jpa.properties.hibernate.default_schema=q8ucpgcjotv3c0vp

   spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=true




   spring.datasource.dbcp2.initial-size = 8

   spring.datasource.dbcp2.max-idle = 8

   spring.datasource.dbcp2.default-query-timeout = 10000

   spring.datasource.dbcp2.default-auto-commit = true
```

Repository/UserRepository.java

```
   package com.javahelps.restservice.repository;

   import java.util.List;

   import java.util.Optional;

   import org.springframework.data.jpa.repository.JpaRepository;

   import
org.springframework.data.jpa.repository.config.EnableJpaRepositories;

   import
org.springframework.data.rest.core.annotation.RepositoryRestResource;

   import org.springframework.data.rest.core.annotation.RestResource;

   import org.springframework.stereotype.Repository;
```

```
import org.springframework.transaction.annotation.Transactional;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestParam;


import com.javahelps.restservice.entity.User;

@RestResource(exported = false)

@Repository

@Transactional

@EnableJpaRepositories

@RepositoryRestResource(exported = false)


public interface UserRepository extends JpaRepository<User,Integer> {


   List                                                    <User>
findByBarcodeAndQtyGreaterThan(@PathVariable("barcode")String
barcode,@RequestParam("qty")Integer qty);

    List<User>findByQty(int qty);

    Optional<User> findById(int qty);

    List<User>   findByBarcodeAndQty(@PathVariable("barcode")   String
barcode,@RequestParam("qty")Integer qty);


   public  List<User> findByQtyGreaterThan(Integer qty);

}
```

Controller / UserController.java

```
package com.javahelps.restservice.controller;


import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import
```

```
org.springframework.data.rest.webmvc.ResourceNotFoundException;

   import org.springframework.http.ResponseEntity;

   import org.springframework.util.CollectionUtils;

   import org.springframework.web.bind.annotation.DeleteMapping;

   import org.springframework.web.bind.annotation.GetMapping;

   import org.springframework.web.bind.annotation.PathVariable;

   import org.springframework.web.bind.annotation.PostMapping;

   import org.springframework.web.bind.annotation.PutMapping;

   import org.springframework.web.bind.annotation.RequestBody;

   import org.springframework.web.bind.annotation.RequestMapping;

   import org.springframework.web.bind.annotation.RequestParam;

   import org.springframework.web.bind.annotation.RestController;


   import com.javahelps.restservice.entity.User;


   import com.javahelps.restservice.repository.UserRepository;


   import javassist.tools.web.BadHttpRequest;


   @RestController

   @RequestMapping(path = "/productnames")

   public class UserController {




      @Autowired

      private UserRepository repository;



      @GetMapping(path = "/getall")
```

```
	public Iterable<User> findAll() {

		return repository.findAll();

	}


	@GetMapping(path = "/{id}")

	public User find(@PathVariable("id") Integer id) {

		return repository.findById(id)

		.orElseThrow(() -> new ResourceNotFoundException("User not found with id :" + id));

	}


	@GetMapping(path = "/0/0")

	public List<User> find2(Integer qty) {

		return repository.findByQty(qty=0);

	}


	@GetMapping(path = "/1/{qty}")

	public List<User> findByQthGreaterThan(@PathVariable Integer qty) {

		return repository.findByQtyGreaterThan(qty);




	}


	@GetMapping(path="barcode2/{barcode}")

	public ResponseEntity <List<User>> findByBarcode(@PathVariable String barcode,@RequestParam("qty")Integer qty) {

		List<User> users= repository.findByBarcodeAndQtyGreaterThan(barcode,qty);

		if (CollectionUtils.isEmpty(users)){

		    return ResponseEntity.noContent()

		        .build();
```

```
            }
            return ResponseEntity.ok()
                        .body(users);
            }




      @GetMapping(path = "/barcode/{barcode}")
      public ResponseEntity <List<User>> findd(@PathVariable("barcode")
String barcode,
                              @RequestParam Integer qty) {
            List<User> users = repository.findByBarcodeAndQty(barcode,
qty);
            if (CollectionUtils.isEmpty(users)){
                return ResponseEntity.noContent()
                     .build();
              }
            return ResponseEntity.ok()
                        .body(users);
            }




      @PostMapping(path = "/post")
      public User create(@RequestBody User user) {
            return repository.save(user);
      }
```

```
	@DeleteMapping(path = "/delete/{id}")
	public void delete(@PathVariable("id") Integer id) {
		repository.deleteById(id);
	}

	@PutMapping(path = "/{id}")
	public User update(@PathVariable("id") Integer id, @RequestBody
User user) throws BadHttpRequest {
		if (repository.existsById(id)) {
			user.setId(id);
			return repository.save(user);
		} else {
			throw new BadHttpRequest();
		}
	}


}
```

6.6 Android – Retrofit

Για να σειριοποιήσουμε το JSON χρειαζόμαστε πρώτα έναν μετατροπέα για να το μετατρέψουμε σε Gson. Πρέπει να προσθέσουμε τις ακόλουθες εξαρτήσεις στο αρχείο build.grade.

```
	implementation 'com.squareup.retrofit2:retrofit:2.1.0'
	implementation 'com.google.code.gson:gson:2.8.5'
	implementation 'com.squareup.retrofit2:converter-gson:2.1.0'
```

Προσθέτουμε την άδεια πρόσβασης για την κάμερα της συσκευής που θα χρησιμοποιήσουμε για αναζήτηση με Barcode και για το διαδίκτυο στο αρχείο AndroidManifest.xml.

```
<uses-permission android:name="android.permission.INTERNET" />
```

```
<uses-permission android:name="android.permission.CAMERA"/>
```

Τώρα πρέπει να δημιουργήσουμε μια διεπαφή για να ορίσουμε τις διάφορες μεθόδους που θα χρησιμοποιούνται για τις συναλλαγές στο δίκτυο.

```
package androidx.appcompat.app.Ariston.remote;


import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;


public class RetrofitClient {
    private static Retrofit retrofit=null;


    public static Retrofit getClient(String url){
        if(retrofit==null){
            retrofit=new Retrofit.Builder().baseUrl(url)

.addConverterFactory(GsonConverterFactory.create())
                    .build();
      }
        return retrofit;
    }
}
```

Πρέπει να ορίσουμε τον APIUltis για την υλοποίηση των API. Ας υποθέσουμε ότι πρέπει να υλοποιήσουμε το API σύνδεσης, τότε πρέπει να κατασκευάσουμε τον APIUltis, να ορίσουμε το root url και να δημιουργήσουμε σύνδεση μεταξύ του προσαρμογέα μας και της διεπαφής API. Για το σκοπό αυτό, είναι καλύτερο να δημιουργήσουμε μια κλάση στην οποία θα έχουμε μια μέθοδο που θα δημιουργεί τη σύνδεση και στη συνέχεια θα επιστρέφει το αντικείμενο API Interface. Μπορείτε να δημιουργήσετε αυτό το APIUltis όπου θέλετε να υλοποιήσετε το API, αλλά είναι προτιμότερο να δημιουργήσετε μια κοινή κλάση/μέθοδο και να τη χρησιμοποιήσετε όπου θέλετε.

```
package androidx.appcompat.app.Ariston.remote;


public class APIUltis {

    private  APIUltis() { };

    public     static     final     String     API_URL="https://kaytu-
crud.herokuapp.com/productnames/";

    public static UserService getUserService(){

        return
RetrofitClient.getClient(API_URL).create(UserService.class);



    }


}
```

Στη συνέχεια, θα ορίσουμε τις μεθόδους GET, POST, PUT, DELETE που εκτελούν αιτήσεις HTTP με τις επισημάνσεις @GET, @POST, @PUT, @DELETE και θα γράψουμε τον ακόλουθο κώδικα.

```
package androidx.appcompat.app.Ariston.remote;


import androidx.appcompat.app.Ariston.model.User;


import java.lang.reflect.Array;
import java.util.ArrayList;
import java.util.List;


import retrofit2.Call;
import retrofit2.http.Body;
import retrofit2.http.DELETE;
import retrofit2.http.GET;
import retrofit2.http.POST;
```

```
import retrofit2.http.PUT;
import retrofit2.http.Path;


public interface UserService {



    @GET("getall")
    Call<List<User>> getUsers();


    @GET("barcode/{barcode}?qty=0")
    Call<ArrayList<User>>checkQty(@Path("barcode") String barcode);


    @GET("1/{qty}")
    Call <List<User>>greaterThan(@Path("qty")Integer qty);


    @GET("barcode2/{barcode}?qty=0")
    Call <List<User>>findByBarcode(@Path("barcode")String barcode);
    @GET("0/0")
    Call <List<User>>find3();


    @POST("post")
    Call<User> addUser(@Body User user);


    @PUT("{id}")
    Call<User> updateUser(@Path("id") int id, @Body User user);


    @DELETE("delete/{id}")
    Call<User> deleteUser(@Path("id") int id);


}
```

Κάνουμε αντιστοίχιση την κάθε πληροφορία που θα λάβουμε μέσω Retrofit .Στη συνέχεια θα τα ορίσουμε σε μια λίστα όπου θα βλέπουμε της πληροφορίες.

```
package androidx.appcompat.app.Ariston;


import android.content.Context;
import android.content.Intent;
import android.graphics.Color;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;


import androidx.annotation.NonNull;
import androidx.appcompat.app.Ariston.model.User;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.content.ContextCompat;
import com.example.ariston.MainActivity;
import com.example.ariston.R;
import com.example.ariston.UserActivity;


import java.util.List;
```

```
   public class UserAdapter extends ArrayAdapter<User> {

       private Context context;

       private List<User> users;


       LinearLayout li;

       public  UserAdapter(@NonNull  Context  context,  int  resource,
@NonNull List<User> objects) {

           super(context, resource, objects);

           this.context=context;

           this.users=objects;

       }


       public   View getView(final int pos, View converView, ViewGroup
parent)

       {




           final      LayoutInflater      inflater=      (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

           View
rowView=inflater.inflate(R.layout.list_user,parent,false);


           TextView txtUserId= rowView.findViewById(R.id.txtId);

           TextView txtBarcode= rowView.findViewById(R.id.txtBarcode);

           TextView txtBname= rowView.findViewById(R.id.txtBname);

           TextView txtName= rowView.findViewById(R.id.txtName);

           TextView txtQty= rowView.findViewById(R.id.txtQty);

           TextView txtDr= rowView.findViewById(R.id.txtDr);

           TextView txtDl= rowView.findViewById(R.id.txtDl);

           TextView txtCat= rowView.findViewById(R.id.txtCat);

           TextView txtCost=rowView.findViewById(R.id.txtCost);
```

```
        ;

txtUserId.setText((String.format("%s",users.get(pos).getId())));

txtBname.setText(String.format("%s",users.get(pos).getBname()));

txtName.setText(String.format("%s",users.get(pos).getName()));

txtBarcode.setText(String.format("%s",users.get(pos).getBarcode()));
           txtQty.setText(String.format("%s",users.get(pos).getQty()));

txtCat.setText(String.format("%s",users.get(pos).getCategory()));
           txtDl.setText(String.format("%s",users.get(pos).getDatel()));
           txtDr.setText(String.format("%s",users.get(pos).getDater()));

txtCost.setText(String.format("%s",users.get(pos).getCost()));


           rowView.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                   //start Activity


                   Intent intent=new Intent(context, UserActivity.class);

intent.putExtra("User_id",String.valueOf((users.get(pos).getId())));

intent.putExtra("User_barcode",String.valueOf((users.get(pos).getBarcode
())));

intent.putExtra("User_bname",String.valueOf((users.get(pos).getBname()))
);

intent.putExtra("User_name",String.valueOf((users.get(pos).getName())));
```

```
intent.putExtra("User_qty",String.valueOf((users.get(pos).getQty())));

intent.putExtra("User_cat",String.valueOf((users.get(pos).getCategory())
));

intent.putExtra("User_dr",String.valueOf((users.get(pos).getDater())));

intent.putExtra("User_dl",String.valueOf((users.get(pos).getDatel())));

intent.putExtra("User_cost",String.valueOf((users.get(pos).getCost())));


                    context.startActivity(intent);

                }

            });

        return rowView;

        }

    }
```

Το Retrofit 2 θα χρησιμοποιήσει τη βιβλιοθήκη μετατροπέα που έχει επιλεγεί για τον χειρισμό της αποδιαταξικοποίησης δεδομένων από ένα αντικείμενο Java. Εάν επισημάνετε την παράμετρο με μια παράμετρο @Body, η εργασία αυτή θα γίνει αυτόματα. Εάν χρησιμοποιείτε τη βιβλιοθήκη Gson, για παράδειγμα, κάθε πεδίο που ανήκει στην κλάση θα σειριοποιηθεί για εσάς. Μπορείτε να αλλάξετε αυτό το όνομα χρησιμοποιώντας τον διακοσμητή @SerializedName.

```
    package androidx.appcompat.app.Ariston.model;

    import com.google.gson.annotations.Expose;

    import com.google.gson.annotations.SerializedName;


    public class User {

        @SerializedName("id")

        @Expose

        private int id;

        @SerializedName("barcode")
```

```
        @Expose
        private String barcode;
        @SerializedName("name")
        @Expose
        private String name;
        @SerializedName("bname")
        @Expose
        private String bname;
        @SerializedName("category")
        @Expose
        private String category;
        @SerializedName("qty")
        @Expose
        private String qty;
        @SerializedName("dater")
        @Expose
        private String dater;
        @SerializedName("datel")
        @Expose
        private String datel;
        @SerializedName("cost")
        @Expose
        private String cost;
        public User(){
        }
        public User(int id, String barcode, String bname, String name,
String category, String qty,String cost, String dater, String datel) {
            this.id=id;
            this.barcode = barcode;
            this.bname = bname;
```

```
            this.name = name;
            this.category = category;
            this.qty = qty;
            this.dater = dater;
            this.datel = datel;
            this.cost=cost;
        }
        public String getBname() {
            return bname;
        }
        public void setBname(String bname) {
            this.bname = bname;
        }
        public int getId() {
            return id;
        }
        public void setId(Integer id) {
            this.id = id;
        }
        public String getBarcode() {
            return barcode;
        }
        public void setBarcode(String barcode) {
            this.barcode = barcode;
        }
        public String getName() {
            return name;
        }
        public void setName(String name) {
            this.name = name;
```

```
    }
    public String getCategory() {
        return category;
    }
    public void setCategory(String category) {
        this.category = category;
    }
    public String getQty() {
        return qty;
    }
    public void setQty(String qty) {
        this.qty = qty;
    }
    public String getDater() {
        return dater;
    }
    public void setDater(String dater) {
        this.dater = dater;
    }
    public String getDatel() {
        return datel;
    }
    public void setDatel(String datel) {
        this.datel = datel;
    }
    public String getCost() {
        return cost;
    }
    public void setCost(String cost) {
        this.cost = cost;
```

```
        }


    }
```

6.7 Android – Εφαρμογή

Main Activity.java

```
package com.example.ariston;

import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.app.Ariston.UserAdapter;
import androidx.appcompat.app.Ariston.model.User;
import androidx.appcompat.app.Ariston.remote.APIUltis;
import androidx.appcompat.app.Ariston.remote.UserService;

import android.app.Activity;
import android.content.Intent;
import android.content.res.Configuration;
import android.os.Build;
import android.os.Bundle;
import android.util.DisplayMetrics;
import android.util.Log;
import android.view.View;
import android.view.WindowManager;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.ListView;
import android.widget.Spinner;
import android.widget.Toast;

import com.google.zxing.integration.android.IntentIntegrator;
import com.google.zxing.integration.android.IntentResult;

import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;

import at.markushi.ui.CircleButton;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class MainActivity extends AppCompatActivity {

    CircleButton btnAddUser;
    CircleButton btnGetUsersList;
    CircleButton btnOrder;
    CircleButton findBarcode;

    EditText scanText;
```

```
        ListView listView;
        Spinner spsort;
        UserService userService;
        List<User> list = new ArrayList<User>();
        loadingDialog loadingDialog=new loadingDialog(MainActivity.this);
        List<User> list2 = new ArrayList<User>();
        private ArrayAdapter<String> mListViewAdapter;
        String scanS;


        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            final Activity activity=this;

            setContentView(R.layout.activity_main);
            setTitle("Ariston");
            btnOrder=(CircleButton)findViewById(R.id.paraggelia);
            btnAddUser = (CircleButton) findViewById(R.id.btnAddUser);
            btnGetUsersList                =                (CircleButton)
findViewById(R.id.btnGetUsersList);
            listView = (ListView) findViewById(R.id.listView);
            findBarcode=(CircleButton)findViewById(R.id.scan);
            scanText=(EditText)findViewById(R.id.scanText);
            userService = APIUltis.getUserService();

            spsort = (Spinner) findViewById(R.id.spinner1);
            final       ArrayAdapter<CharSequence>       adapter       =
ArrayAdapter.createFromResource(this,
                    R.array.sort, android.R.layout.simple_spinner_item);


adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown
_item);
            spsort.setAdapter(adapter);
            greaterThan(0);
            spsort.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener()


            {


                @RequiresApi(api = Build.VERSION_CODES.N)
                @Override
                public  void  onItemSelected(AdapterView<?>  parent,  View
view, int position, long id) {


                   String item =spsort.getSelectedItem().toString();

                    if(item.equals("ID(0-9)"))
                    {
                        list.sort(Comparator.comparing(User::getId));
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }
                    else if(item.equals("ID(9-0)")){
```

```
list.sort(Comparator.comparing(User::getId).reversed());
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }
                    else if(item.equals("ΕΤΑΙΡΙΑ(Α-Ζ)")){


                        list.sort(Comparator.comparing(User::getBname));
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }

                    else if(item.equals("ΕΤΑΙΡΙΑ(Ζ-Α)")){



list.sort(Comparator.comparing(User::getBname).reversed());
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }
                    else if(item.equals("ΟΝΟΜΑ(Α-Ζ)")){


                        list.sort(Comparator.comparing(User::getName));
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }

                    else if(item.equals("ΟΝΟΜΑ(Ζ-Α)")){



list.sort(Comparator.comparing(User::getName).reversed());
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }
                    else if(item.equals("ΚΑΤΗΓΟΡΙΑ(Α-Ζ)")){



list.sort(Comparator.comparing(User::getCategory));
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }

                    else if(item.equals("ΚΑΤΗΓΟΡΙΑ(Ζ-Α)")){
```

```
list.sort(Comparator.comparing(User::getCategory).reversed());
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }


                    else if(item.equals("ΗΜ/ΝΑ Π(↓)")){


                        list.sort(Comparator.comparing(User::getDater));
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }

                    else if(item.equals("ΗΜ/ΝΑ Π(↑)")){



list.sort(Comparator.comparing(User::getDater).reversed());
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }
                    else if(item.equals("ΗΜ/ΝΑ Λ(↓)")){


                        list.sort(Comparator.comparing(User::getDatel));
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }

                    else if(item.equals("ΗΜ/ΝΑ Λ(↑)")){



list.sort(Comparator.comparing(User::getDatel).reversed());
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        Toast.makeText(MainActivity.this,           item,
Toast.LENGTH_SHORT).show();
                    }
                }

                @Override
                public void onNothingSelected(AdapterView<?> parent) {

                }
            });

            findBarcode.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
```

```
                    scanS  =  scanText.getText().toString();

                        if(scanS.equals("")) {
                            IntentIntegrator        integrator        =new
IntentIntegrator(activity);
                            integrator.setPrompt("Scan");
                            integrator.setCameraId(0);
                            integrator.setBeepEnabled(true);
                            integrator.setBarcodeImageEnabled(true);
                            integrator.initiateScan();

                        }
                     else {
                        scan(scanS);
                        scanText.setText("");
                        }

                }
            });

            btnGetUsersList.setOnClickListener(new View.OnClickListener()
{
                @Override
                public void onClick(View v) {

                    //get users list
                    greaterThan(0);
                }
            });
            btnOrder.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                   find3();
                }
            });
            btnAddUser.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Intent   intent   =   new   Intent(MainActivity.this,
UserActivity.class);
                    intent.putExtra("user_name", "");
                    startActivity(intent);
                }
            });
        }


        private View getItemText(View itemText) {
            return itemText;
        }

        public void getUsersList(){
            loadingDialog.startloadingDialog();
            Call<List<User>> call = userService.getUsers();
            call.enqueue(new Callback<List<User>>() {
                @Override
                public     void     onResponse(Call<List<User>>     call,
Response<List<User>> response) {
                    if(response.isSuccessful()){
                        loadingDialog.dismissDialog();
                        list = response.body();
```

```
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                    }
                }

                @Override
                public void onFailure(Call<List<User>> call, Throwable t)
{
                    loadingDialog.dismissDialog();
                    Log.e("ERROR: ", t.getMessage());
                }
            });
        }

        public void scan(String scan){
            loadingDialog.startloadingDialog();
            Call<List<User>> call = userService.findByBarcode(scan);
            call.enqueue(new Callback<List<User>>() {
                @Override
                public     void     onResponse(Call<List<User>>     call,
Response<List<User>> response) {
                    if(response.code()==200){
                        list = response.body();
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        loadingDialog.dismissDialog();
                    }
                        else if(response.code()==204){
                        Toast.makeText(MainActivity.this,  "Δεν  βρέθηκαν
αποτελέσματα", Toast.LENGTH_SHORT).show();
                        loadingDialog.dismissDialog();
                        }

                }

                @Override
                public void onFailure(Call<List<User>> call, Throwable t)
{
                    loadingDialog.dismissDialog();
                }
            });

        }
        public void greaterThan(int qty){
            loadingDialog.startloadingDialog();
            Call<List<User>> call = userService.greaterThan(qty);
            call.enqueue(new Callback<List<User>>() {
                @Override
                public     void     onResponse(Call<List<User>>     call,
Response<List<User>> response) {
                    if(response.isSuccessful()){

                        list = response.body();
                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        loadingDialog.dismissDialog();
                    }
                }

                @Override
                public void onFailure(Call<List<User>> call, Throwable t)
```

```
{
                    loadingDialog.dismissDialog();
                    Log.e("ERROR: ", t.getMessage());
                }
            });
        }
        public void find3(){
            loadingDialog.startloadingDialog();
            Call<List<User>> call = userService.find3();
            call.enqueue(new Callback<List<User>>() {
                @Override
                public     void     onResponse(Call<List<User>>     call,
Response<List<User>> response) {
                    if(response.isSuccessful()){


                        list = response.body();

                        listView.setAdapter(new
UserAdapter(MainActivity.this, R.layout.list_user, list));
                        loadingDialog.dismissDialog();
                    }
                }

                @Override
                public void onFailure(Call<List<User>> call, Throwable t)
{
                    loadingDialog.dismissDialog();
                    Log.e("ERROR: ", t.getMessage());
                }
            });
        }

        protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
            IntentResult                        result                        =
IntentIntegrator.parseActivityResult(requestCode, resultCode, data);
            if (result != null) {

                if (result.getContents() == null) {
                    Toast.makeText(this,    "Ακυρώσατε    την    σάρωση",
Toast.LENGTH_LONG).show();
                } else {
                    Toast.makeText(this,            result.getContents(),
Toast.LENGTH_LONG).show();
                    String barText = (String) result.getContents();
                    scan(barText);
                }

            }
            else {
                super.onActivityResult(requestCode,resultCode,data);
            }
        }
            public void hide (){
                getWindow().getDecorView().setSystemUiVisibility(
                        View.SYSTEM_UI_FLAG_LAYOUT_STABLE
                                |
View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION
                                | View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
                                | View.SYSTEM_UI_FLAG_HIDE_NAVIGATION
```

```
                                | View.SYSTEM_UI_FLAG_FULLSCREEN
                                | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY);

        }

    }
```

Activity_main.xml

```
    <?xml version="1.0" encoding="utf-8"?>
    <androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center"
        android:screenOrientation="landscape"
        tools:context=".MainActivity">


        <ImageView
            android:id="@+id/imageView"
            android:layout_width="291dp"
            android:layout_height="130dp"
            android:layout_marginBottom="164dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:srcCompat="@drawable/logo" />

        <LinearLayout
            android:id="@+id/linearLayout2"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:background="#C0CBD1"
            android:gravity="center"
            android:orientation="horizontal"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.0">

            <TextView
                android:id="@+id/txtId"
                android:layout_width="55dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:gravity="center"
                android:text="ID"
                android:textSize="18dp"
                android:visibility="gone" />

            <TextView
                android:id="@+id/txtBarcode"
                android:layout_width="95dp"
```

```
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="Barcode"
            android:textSize="18dp"
            tools:layout_editor_absoluteX="56dp" />

        <TextView
            android:id="@+id/txtBrand"
            android:layout_width="95dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="@string/ETAIRIA"
            android:textSize="18dp"
            tools:layout_editor_absoluteX="152dp" />

        <TextView
            android:id="@+id/txtONOMA"
            android:layout_width="95dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="@string/ONOMA"
            android:textSize="18dp"
            tools:layout_editor_absoluteX="248dp" />

        <TextView
            android:id="@+id/txtCat"
            android:layout_width="95dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="@string/Cat"
            android:textSize="18dp"
            tools:layout_editor_absoluteX="344dp" />

        <TextView
            android:id="@+id/txtQty"
            android:layout_width="75dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="@string/Posotita"
            android:textSize="18dp"
            tools:layout_editor_absoluteX="446dp" />

        <TextView
            android:id="@+id/txtDater"
            android:layout_width="75dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="@string/Dater"
            android:textSize="18dp"
            tools:layout_editor_absoluteX="547dp" />

        <TextView
            android:id="@+id/txtDatel"
            android:layout_width="75dp"
            android:layout_height="wrap_content"
```

```
                android:layout_weight="1"
                android:gravity="center"
                android:text="@string/Datel"
                android:textSize="18dp"
                tools:layout_editor_absoluteX="649dp" />

            <TextView
                android:id="@+id/textView3"
                android:layout_width="75dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:gravity="center"
                android:text="@string/Cost"
                android:textSize="18dp" />

        </LinearLayout>

        <ListView
            android:id="@+id/listView"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:cacheColorHint="#8C9EFF"
            app:layout_constraintBottom_toTopOf="@+id/linearLayout"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.0"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/linearLayout2"
            app:layout_constraintVertical_bias="0.0">

        </ListView>

        <androidx.constraintlayout.widget.ConstraintLayout
            android:id="@+id/linearLayout"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:background="#FF5900"
            android:orientation="horizontal"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="1.0"
            app:layout_constraintStart_toStartOf="parent">


            <at.markushi.ui.CircleButton

                android:id="@+id/btnAddUser"
                android:layout_width="80dp"
                android:layout_height="60dp"
                android:layout_marginStart="248dp"
                android:layout_marginLeft="248dp"
                android:src="@drawable/add2"
                android:text="Add User"
                app:cb_color="#FFFFFF"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toStartOf="@+id/btnGetUsersList"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent"

app:layout_constraintVertical_bias="0.531"></at.markushi.ui.CircleButton
>
```

```
            <at.markushi.ui.CircleButton
                android:id="@+id/btnGetUsersList"
                android:layout_width="80dp"
                android:layout_height="60dp"
                android:src="@drawable/renew"
                app:cb_color="#FFFFFF"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toTopOf="parent"></at.markushi.ui.CircleButton>

            <Spinner
                android:id="@+id/spinner1"
                android:layout_width="140dp"
                android:layout_height="30dp"
                android:layout_weight="1"
                android:background="#FFFFFF"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintHorizontal_bias="0.878"
                app:layout_constraintStart_toEndOf="@+id/btnAddUser"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintVertical_bias="0.76" />

            <TextView
                android:id="@+id/textView2"
                android:layout_width="142dp"
                android:layout_height="21dp"
                android:layout_marginEnd="40dp"
                android:layout_marginRight="40dp"
                android:gravity="center"
                android:text="@string/Sortfor"
                app:layout_constraintBottom_toTopOf="@+id/spinner1"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintHorizontal_bias="1.0"
                app:layout_constraintStart_toEndOf="@+id/btnGetUsersList"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintVertical_bias="0.0" />

            <at.markushi.ui.CircleButton
                android:id="@+id/paraggelia"
                android:layout_width="80dp"
                android:layout_height="60dp"
                android:src="@drawable/order"
                app:cb_color="#FFFFFF"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintHorizontal_bias="0.059"
                app:layout_constraintStart_toEndOf="@+id/btnGetUsersList"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintVertical_bias="0.0" />

            <EditText
                android:id="@+id/scanText"
                android:layout_width="120dp"
                android:layout_height="30dp"
                android:layout_marginStart="4dp"
                android:layout_marginLeft="4dp"
                android:background="#FFFFFF"
                android:hint="@string/sbarcode"
```

```
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintVertical_bias="0.531" />

            <at.markushi.ui.CircleButton
                android:id="@+id/scan"
                android:layout_width="40dp"
                android:layout_height="60dp"
                android:layout_marginStart="12dp"
                android:layout_marginLeft="12dp"
                android:src="@drawable/scan2"
                android:text="Button"
                app:cb_color="#FFFFFF"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintStart_toEndOf="@+id/scanText"
                app:layout_constraintTop_toTopOf="parent" />
        </androidx.constraintlayout.widget.ConstraintLayout>

    </androidx.constraintlayout.widget.ConstraintLayout>
```

UserActivity.Java

```
package com.example.ariston;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.DatePickerDialog;
import android.content.Intent;

import android.hardware.Camera;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.util.Log;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.ListView;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.app.Ariston.model.User;
import androidx.appcompat.app.Ariston.remote.APIUltis;
import androidx.appcompat.app.Ariston.remote.UserService;
import com.google.zxing.integration.android.IntentIntegrator;
import com.google.zxing.integration.android.IntentResult;

import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
```

```
import java.util.Calendar;
import java.util.Date;
import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class UserActivity extends AppCompatActivity implements
DatePickerDialog.OnDateSetListener {

    UserService userService;
    TextView edtUId;
    EditText edtUname;
    EditText edtUBarcode;
    EditText edtUBname;
    EditText edtUQty;
    EditText edtUDater;
    EditText edtUDatel;
    EditText edtUCost;
    ImageButton btnSave;
    ImageButton btnDel;
    ImageButton btnBack;
    ImageButton dil1;
    ImageButton dil2;
    TextView txtUId;
    TextView txtUName;
    TextView txtUBarcode;
    TextView txtUBname;
    TextView txtUQty;
    TextView txtUCat;
    TextView txtUDater;
    TextView txtUDatel;
    TextView txtUCost;
    ImageButton bar;
    Spinner edtUcat;
    ArrayAdapter<CharSequence> adapter;
    String dill="";
    Integer check;
    List<User> list = new ArrayList<User>();
    ImageButton btnAnevase;
    ImageButton btnKatevase;
    ListView listView;
    loadingDialog loadingDialog=new loadingDialog(UserActivity.this);

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        getWindow().getDecorView().setSystemUiVisibility(
                View.SYSTEM_UI_FLAG_LAYOUT_STABLE
                        | View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION
                        | View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
                        | View.SYSTEM_UI_FLAG_HIDE_NAVIGATION
                        | View.SYSTEM_UI_FLAG_FULLSCREEN
                        | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY);
        setContentView(R.layout.activity_user2);
        final Activity activity=this;
        AlertDialog.Builder           builder           =           new
AlertDialog.Builder(UserActivity.this);
        listView = (ListView) findViewById(R.id.listView);
        txtUId = (TextView) findViewById(R.id.txtUId);
```

```
        edtUId = (TextView) findViewById(R.id.edtUId);

        txtUBarcode = (TextView) findViewById(R.id.txtUBarcode);
        edtUBarcode = (EditText) findViewById(R.id.edtUBarcode);

        txtUBname= (TextView) findViewById(R.id.txtUBname);
        edtUBname = (EditText) findViewById(R.id.edtUBname);


        txtUName= (TextView) findViewById(R.id.txtUName);
        edtUname = (EditText) findViewById(R.id.edtUName);

        txtUQty= (TextView) findViewById(R.id.txtUQty);
        edtUQty = (EditText) findViewById(R.id.edtUQty);

        txtUCat= (TextView) findViewById(R.id.txtUCat);
        edtUcat = (Spinner) findViewById(R.id.edtUCat);

        txtUDater= (TextView) findViewById(R.id.txtUDater);
        edtUDater = (EditText) findViewById(R.id.edtUDater);

        txtUDatel= (TextView) findViewById(R.id.txtUDatel);
        edtUDatel = (EditText) findViewById(R.id.edtUDatel);

        txtUCost= (TextView) findViewById(R.id.txtUCost);
        edtUCost = (EditText) findViewById(R.id.edtUCost);
        btnKatevase=(ImageButton)findViewById(R.id.button3);
        btnSave = (ImageButton) findViewById(R.id.btnSave);
        btnDel = (ImageButton) findViewById(R.id.btnDel);
        dil1=(ImageButton)findViewById(R.id.dialeks2);
        dil2=(ImageButton)findViewById(R.id.dialeks1);
        btnBack=(ImageButton)findViewById(R.id.btnBack);
        bar=(ImageButton)findViewById(R.id.bar);
        userService = APIUltis.getUserService();
        btnAnevase=(ImageButton)findViewById(R.id.anevase);
        Bundle extras = getIntent().getExtras();
        final String userId = extras.getString("User_id");
        final String userBarcode = extras.getString("User_barcode");
        final String userQty = extras.getString("User_qty");

        String userBname = extras.getString("User_bname");
        String userName = extras.getString("User_name");

        String userCat = extras.getString("User_cat");
        String userDatel = extras.getString("User_dl");
        String userDater = extras.getString("User_dr");
        String userCost=extras.getString("User_cost");
        String pasteData = edtUDater.getText().toString();


adapter=ArrayAdapter.createFromResource(this,R.array.katigories,android.
R.layout.simple_spinner_item);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown
_item);
        edtUcat.setAdapter(adapter);
        edtUId.setText(userId);
        edtUBarcode.setText(userBarcode);
        edtUBname.setText(userBname);
        edtUname.setText(userName);
        edtUQty.setText(userQty);
```

```
        edtUcat.setSelected(Boolean.parseBoolean(userCat));
        edtUDatel.setText(userDatel);
        edtUDater.setText(userDater);
        edtUCost.setText(userCost);

      btnAnevase.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View v) {
              int t = Integer.parseInt(edtUQty.getText().toString());
              edtUQty.setText(String.valueOf(t+1));
          }
      });

        btnKatevase.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String getid;
                getid=(edtUQty.getText().toString());
                check=Integer.parseInt(getid);

                if (check>0) {
                    int                         t                         =
Integer.parseInt(edtUQty.getText().toString());
                    edtUQty.setText(String.valueOf(t - 1)); }
                }
        });


        dil1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                showDatePickerDialog();
                dill="1";
            }
        });
        dil2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                showDatePickerDialog();
                dill="2";
            }
        });

        btnBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent    intent    =    new    Intent(UserActivity.this,
MainActivity.class);
                startActivity(intent);            }
        });

        if(userId != null && userId.trim().length() > 0 ){
            edtUId.setFocusable(false);
        } else {
            txtUId.setVisibility(View.INVISIBLE);
            edtUId.setVisibility(View.INVISIBLE);
            btnDel.setVisibility(View.INVISIBLE);
            edtUQty.setText("0");
            edtUCost.setText("0.0");
        }
```

```
        bar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                IntentIntegrator             integrator             =new
IntentIntegrator(activity);
                integrator.setPrompt("Scan");
                integrator.setCameraId(0);
                integrator.setBeepEnabled(false);
                integrator.setBarcodeImageEnabled(true);

                integrator.initiateScan();
            }
        });
        edtUcat.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            @Override
            public void onItemSelected(AdapterView<?> parent, View view,
int position, long id) {

                String itemText = (String) edtUcat.getSelectedItem();

            }

            @Override
            public void onNothingSelected(AdapterView<?> parent) {

            }
        });
        btnSave.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                User u = new User();
                String getid;
                getid=(edtUQty.getText().toString());
                check=Integer.parseInt(getid);

                String startDateString = edtUDater.getText().toString()
;  // where startDate is your TextView
                String startDateString2 = edtUDatel.getText().toString()
;

                DateFormat  dateFormat=  new  SimpleDateFormat("yyyy-MM-
dd");

                Date date1= new Date();
                Date date2=new Date();

                try {
                    date1 = dateFormat.parse(startDateString);

                } catch (ParseException e) {
                    e.printStackTrace();
                }
                try {
                    date2 = dateFormat.parse(startDateString2);

                } catch (ParseException e) {
                    e.printStackTrace();
                }
```

```
                if (edtUBarcode.getText().toString().equals("")) {

                    builder.setTitle("Προειδοποίηση");
                    builder.setMessage("Το πεδίο BARCODE δεν μπορεί να
είναι κενό");
                    // add a button
                    builder.setPositiveButton("OK", null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                    View view = new View(getApplicationContext());
                    view.setBackgroundResource(R.color.Red);
                    edtUBarcode.setBackground(view.getBackground());
                }
                else if (edtUBname.getText().toString().equals("")) {

                    builder.setTitle("Προειδοποίηση");
                    builder.setMessage("Το πεδίο ΕΤΑΙΡΙΑ δεν μπορεί να
είναι κενό");
                    // add a button
                    builder.setPositiveButton("OK", null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                    View view = new View(getApplicationContext());
                    view.setBackgroundResource(R.color.Red);
                    edtUBname.setBackground(view.getBackground());
                }
                else if (edtUname.getText().toString().equals("")) {

                    builder.setTitle("Προειδοποίηση");
                    builder.setMessage("Το πεδίο ΟΝΟΜΑ δεν μπορεί να
είναι κενό");
                    // add a button
                    builder.setPositiveButton("OK", null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                    View view = new View(getApplicationContext());
                    view.setBackgroundResource(R.color.Red);
                    edtUname.setBackground(view.getBackground());
                }
                else if (check<0) {

                    builder.setTitle("Προειδοποίηση");
                    builder.setMessage("Το πεδίο ΠΟΣΟΤΗΤΑ δεν μπορεί να
είναι αρνητικό");
                    // add a button
                    builder.setPositiveButton("OK", null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                    View view = new View(getApplicationContext());
                    view.setBackgroundResource(R.color.Red);
                    edtUQty.setBackground(view.getBackground());
                }

                else if(date1.after(date2)){

                    builder.setTitle("Προειδοποίηση");
                    builder.setMessage("ΗΜ/ΝΑ ΠΑΡΑΛΑΒΗΣ δεν μπορεί να
είναι μεγαλύτερο του ΗΜ/ΝΑ ΛΗΞΗΣ");
                    // add a button
                    builder.setPositiveButton("OK", null);
                    AlertDialog dialog = builder.create();
```

```
                    dialog.show();
                    View view = new View(getApplicationContext());
                    view.setBackgroundResource(R.color.Red);
                    edtUDater.setBackground(view.getBackground());
                }

                else if(date2.before(date1)){

                    builder.setTitle("Προειδοποίηση");
                    builder.setMessage("ΗΜ/ΝΑ ΛΗΞΗΣ δεν μπορεί να είναι μικρότερο του ΗΜ/ΝΑ ΠΑΡΑΛΑΒΗΣ");
                    // add a button
                    builder.setPositiveButton("OK", null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                    View view = new View(getApplicationContext());
                    view.setBackgroundResource(R.color.Red);
                    edtUDatel.setBackground(view.getBackground());
                }
                else   if(edtUCost.getText().toString().trim().isEmpty()
){

                    builder.setTitle("Προειδοποίηση");
                    builder.setMessage("Το πεδίο ΚΩΣΤΟΣ πρέπει να μην είναι κενό και πρέπει να περιέχει μόνο αριθμούς");
                    // add a button
                    builder.setPositiveButton("OK", null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                    View view = new View(getApplicationContext());
                    view.setBackgroundResource(R.color.Red);
                    edtUCost.setBackground(view.getBackground());
                }

                else {

                    u.setBarcode(edtUBarcode.getText().toString());
                    u.setName(edtUname.getText().toString());
                    u.setBname(edtUBname.getText().toString());
                    u.setCategory(edtUcat.getSelectedItem().toString());
                    u.setQty(edtUQty.getText().toString());
                    u.setDater(edtUDater.getText().toString());
                    u.setDatel(edtUDatel.getText().toString());
                    u.setCost(edtUCost.getText().toString());


                    if (userId != null && userId.trim().length() > 0) {
                        //update user
                        updateUser(Integer.parseInt(userId), u);
                    } else {
                        //add user
                        addUser(u);
                    }
                }
            }
        });

        btnDel.setOnClickListener(new View.OnClickListener() {


            @Override
```

```
            public void onClick(View v) {
             String getid;
             getid=(edtUId.getText().toString());
                check=Integer.parseInt(getid);


                checkQty(userBarcode);

            }
        });

    }

    public void deletemethod(int id){

        loadingDialog.startloadingDialog();
        Call<User> call = userService.deleteUser(id);
        call.enqueue(new Callback<User>() {
            @Override
            public  void  onResponse(Call<User>  call,  Response<User>
response) {
                if(response.isSuccessful()){
                    Toast.makeText(UserActivity.this,       "Διαγράφθηκε
επιτυχώς ", Toast.LENGTH_SHORT).show();
                    Intent  intent  =  new  Intent(UserActivity.this,
MainActivity.class);
                    startActivity(intent);
                    loadingDialog.dismissDialog();

                }
            }

            @Override
            public void onFailure(Call<User> call, Throwable t) {
                Toast.makeText(UserActivity.this, " Διαγράφθηκε επιτυχώς
", Toast.LENGTH_SHORT).show();
                loadingDialog.dismissDialog();
                Intent    intent    =    new    Intent(UserActivity.this,
MainActivity.class);
                startActivity(intent);
            }
        });


    }

    public void updatemethod(int id ){
        User u = new User();
        edtUQty.setText("0");
        edtUDatel.setText(null);
        edtUDater.setText(null);
        edtUCost.setText("0.0");

        u.setBarcode(edtUBarcode.getText().toString());
        u.setName(edtUname.getText().toString());
        u.setBname(edtUBname.getText().toString());
        u.setCategory(edtUcat.getSelectedItem().toString());
        u.setQty(edtUQty.getText().toString());
        u.setDater(edtUDater.getText().toString());
        u.setDatel(edtUDatel.getText().toString());
        u.setCost(edtUCost.getText().toString());
```

```
        loadingDialog.startloadingDialog();
        Call<User> call = userService.updateUser(id, u);
        call.enqueue(new Callback<User>() {
            @Override
            public void onResponse(Call<User> call, Response<User>
response) {
                if(response.isSuccessful()){

                    Toast.makeText(UserActivity.this, "Οι Αλλαγές
Ολοκληρώθηκαν", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(UserActivity.this,
MainActivity.class);
                    startActivity(intent);
                    loadingDialog.dismissDialog();
                }
            }

            @Override
            public void onFailure(Call<User> call, Throwable t) {
                Log.e("ERROR: ", t.getMessage());
                loadingDialog.dismissDialog();
            }
        });
    }



    public void addUser(User u){
        loadingDialog.startloadingDialog();
        Call<User> call = userService.addUser(u);
        call.enqueue(new Callback<User>() {
            @Override
            public void onResponse(Call<User> call, Response<User>
response) {
                if(response.isSuccessful()){
                    Toast.makeText(UserActivity.this, "Το προίον εισήχθη
επιτυχώς", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(UserActivity.this,
MainActivity.class);
                    startActivity(intent);
                loadingDialog.dismissDialog();

                }
            }

            @Override
            public void onFailure(Call<User> call, Throwable t) {
                Log.e("ERROR: ", t.getMessage());
                loadingDialog.dismissDialog();
            }
        });
    }

    public void checkQty(String barcode){

        Call<ArrayList<User>> call = userService.checkQty(barcode);
        call.enqueue(new Callback<ArrayList<User>>() {
```

```
            @Override
            public    void    onResponse(Call<ArrayList<User>>    call,
Response<ArrayList<User>> response) {

                if (response.code()==200) {

                    deletemethod(check);


                }
                else if(response.code()==204)
                {

                    updatemethod(check);


                }


            }

            @Override
            public void onFailure(Call<ArrayList<User>> call, Throwable
t) {
                Log.e("ERROR: ", t.getMessage());
            }
        });

    }

    public void updateUser(int id, User u){
        loadingDialog.startloadingDialog();
        Call<User> call = userService.updateUser(id, u);
        call.enqueue(new Callback<User>() {
            @Override
            public   void   onResponse(Call<User>   call,   Response<User>
response) {
                if(response.isSuccessful()){

                    Toast.makeText(UserActivity.this,     "Οι     Αλλαγές
Ολοκληρώθηκαν", Toast.LENGTH_SHORT).show();
                    Intent   intent   =   new   Intent(UserActivity.this,
MainActivity.class);
                    startActivity(intent);
                    loadingDialog.dismissDialog();
                }
            }

            @Override
            public void onFailure(Call<User> call, Throwable t) {
                Log.e("ERROR: ", t.getMessage());
                loadingDialog.dismissDialog();
            }
        });
    }

    public void deleteUser(int id){
        loadingDialog.startloadingDialog();
        Call<User> call = userService.deleteUser(id);
        call.enqueue(new Callback<User>() {
```

```
            @Override
            public void onResponse(Call<User> call, Response<User>
response) {
                if(response.isSuccessful()){
                    Toast.makeText(UserActivity.this, "Διαγράφθηκε
επιτυχώς ", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(UserActivity.this,
MainActivity.class);
                    startActivity(intent);
                    loadingDialog.dismissDialog();

                }
            }

            @Override
            public void onFailure(Call<User> call, Throwable t) {
                Log.e("ERROR: ", t.getMessage());
                loadingDialog.dismissDialog();
            }
        });
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case android.R.id.home:
                finish();
                return true;
        }

        return super.onOptionsItemSelected(item);
    }



    protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
        IntentResult result =
IntentIntegrator.parseActivityResult(requestCode, resultCode, data);
        if (result != null) {

            if (result.getContents() == null) {
                Toast.makeText(this, "Ακυρώσατε την σάρωση",
Toast.LENGTH_LONG).show();
            } else {
                Toast.makeText(this, result.getContents(),
Toast.LENGTH_LONG).show();
                String barText = (String) result.getContents();
                edtUBarcode.setText(barText);
            }

        }

        else {
            super.onActivityResult(requestCode,resultCode,data);
        }
    }
    public void showDatePickerDialog() {
        DatePickerDialog datePickerDialog = new DatePickerDialog(
                this,
                this,
```

```
                Calendar.getInstance().get(Calendar.YEAR),
                Calendar.getInstance().get(Calendar.MONTH),
                Calendar.getInstance().get(Calendar.DAY_OF_MONTH));
        datePickerDialog.show();

    }
    public void onDateSet(DatePicker view, int year, int month, int
dayOfMonth) {
        String date = "" + year +  "-" + (month+1)  + "-" +dayOfMonth ;

        if(dill=="1")
        {
            edtUDater.setText(date);
        }
        else if(dill=="2")

        {
            edtUDatel.setText(date);
        }

    }
}
```

Activity_user2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:background="#FFFFFF"
    android:orientation="vertical"
    android:padding="1dp"
    app:layout_scrollFlags="scroll">

    <TextView
        android:id="@+id/txtUId"
        android:layout_width="110dp"
        android:layout_height="20dp"
        android:layout_toRightOf="@id/edtUId"
        android:layout_weight="1"
        android:text="id"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/edtUId"
        android:layout_width="110dp"
        android:layout_height="20dp"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="textPersonName"
        android:textSize="16sp"
        app:layout_constraintStart_toEndOf="@+id/txtUId"
        app:layout_constraintTop_toTopOf="parent" />
```

```
    <EditText
        android:id="@+id/edtUBarcode"
        android:layout_width="180dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"
        android:textSize="20sp"
        app:layout_constraintStart_toEndOf="@+id/txtUBarcode"
        app:layout_constraintTop_toBottomOf="@+id/edtUId" />

    <EditText
        android:id="@+id/edtUName"
        android:layout_width="180dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:layout_marginTop="12dp"

        android:layout_weight="1"
        android:ems="10"
        android:gravity="center_vertical"
        android:inputType="textCapSentences|textMultiLine"
        android:textSize="20sp"
        app:layout_constraintStart_toEndOf="@+id/txtUName"
        app:layout_constraintTop_toBottomOf="@+id/edtUBname" />

    <ImageButton
        android:id="@+id/bar"
        android:layout_width="44dp"
        android:layout_height="42dp"
        android:layout_marginStart="24dp"
        android:layout_marginTop="40dp"
        android:layout_weight="1"
        android:background="#00FFFFFF"
        android:src="@drawable/scan"
        android:textCursorDrawable="@android:drawable/ic_menu_add"
        app:layout_constraintStart_toEndOf="@+id/edtUBarcode"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/edtUBname"
        android:layout_width="180dp"
        android:layout_height="60dp"

        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="textCapSentences|textMultiLine"
        android:textSize="20sp"
        app:layout_constraintStart_toEndOf="@+id/txtUBname"
        app:layout_constraintTop_toBottomOf="@+id/edtUBarcode" />

    <Spinner
        android:id="@+id/edtUCat"
        android:layout_width="180dp"
```

```
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="textPersonName"
        android:text=""
        app:layout_constraintStart_toEndOf="@+id/txtUCat"
        app:layout_constraintTop_toBottomOf="@+id/edtUName" />

    <EditText
        android:id="@+id/edtUQty"
        android:layout_width="180dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"

        android:inputType="numberDecimal"
        android:textSize="20sp"
        app:layout_constraintStart_toEndOf="@+id/txtUQty"
        app:layout_constraintTop_toBottomOf="@+id/edtUCat" />

    <ImageButton
        android:id="@+id/anevase"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="248dp"
        android:background="#76FF03"
        android:src="@drawable/up"
        android:text="up"
        app:layout_constraintStart_toEndOf="@+id/button3"
        app:layout_constraintTop_toBottomOf="@+id/bar" />

    <ImageButton
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="24dp"
        android:layout_marginTop="248dp"
        android:background="#F30A0A"
        android:src="@drawable/down"
        android:text="Button"
        app:layout_constraintStart_toEndOf="@+id/edtUQty"
        app:layout_constraintTop_toBottomOf="@+id/bar" />

    <EditText
        android:id="@+id/edtUDater"
        android:layout_width="180dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"
        android:textSize="20sp"
        app:layout_constraintStart_toEndOf="@+id/txtUDater"
        app:layout_constraintTop_toBottomOf="@+id/edtUQty" />
```

```
    <ImageButton
        android:id="@+id/dialeks2"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:layout_gravity="clip_horizontal"
        android:layout_marginStart="36dp"
        android:layout_marginTop="36dp"
        android:background="#00FFFFFF"
        android:src="@drawable/calendar"
        android:text="Button"
        app:layout_constraintStart_toEndOf="@+id/edtUDatel"
        app:layout_constraintTop_toBottomOf="@+id/button3" />

    <EditText
        android:id="@+id/edtUDatel"
        android:layout_width="180dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="date"
        android:textSize="20sp"
        app:layout_constraintStart_toEndOf="@+id/txtUDatel"
        app:layout_constraintTop_toBottomOf="@+id/edtUDater" />

    <ImageButton
        android:id="@+id/dialeks1"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:layout_marginStart="36dp"
        android:layout_marginBottom="96dp"
        android:background="#00FFFFFF"
        android:src="@drawable/calendar"
        android:text="Button"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toEndOf="@+id/edtUDatel"
        app:layout_constraintTop_toBottomOf="@+id/dialeks2"
        app:layout_constraintVertical_bias="0.447" />

    <EditText
        android:id="@+id/edtUCost"
        android:layout_width="180dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="numberDecimal"

        android:textSize="18sp"
        app:layout_constraintStart_toEndOf="@+id/txtUCost"
        app:layout_constraintTop_toBottomOf="@+id/edtUDatel" />

    <TextView
        android:id="@+id/txtUBarcode"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
```

```
        android:gravity="center_vertical"
        android:text="Barcode"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUId" />

    <TextView
        android:id="@+id/txtUBname"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:gravity="center_vertical"
        android:text="@string/ETAIRIA"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUBarcode" />

    <TextView
        android:id="@+id/txtUName"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:gravity="center|left"
        android:text="@string/ONOMA"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUBname" />

    <TextView
        android:id="@+id/txtUCat"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:gravity="center|left"
        android:text="@string/Cat"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUName" />

    <TextView
        android:id="@+id/txtUQty"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:gravity="center|left"
        android:text="@string/Posotita"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUCat" />

    <TextView
        android:id="@+id/txtUDater"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:gravity="center|left"
```

```
        android:text="@string/Dater"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/txtUQty" />

    <TextView
        android:id="@+id/txtUDatel"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:gravity="center|left"
        android:text="@string/Datel"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUDater" />

    <TextView
        android:id="@+id/txtUCost"
        android:layout_width="140dp"
        android:layout_height="60dp"
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:ems="10"
        android:gravity="center|left"
        android:text="@string/Cost"
        android:textSize="22sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUDatel" />

    <ImageButton
        android:id="@+id/btnBack"
        style="@android:style/Widget.DeviceDefault.ImageButton"
        android:layout_width="48dp"
        android:layout_height="49dp"
        android:layout_marginStart="64dp"
        android:layout_marginLeft="64dp"
        android:layout_marginTop="20dp"
        android:layout_weight="1"
        android:background="#00FFFFFF"
        android:longClickable="false"
        android:src="@drawable/back"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtUCost" />

    <ImageButton
        android:id="@+id/btnSave"
        android:layout_width="48dp"
        android:layout_height="49dp"
        android:layout_marginStart="44dp"
        android:layout_marginLeft="44dp"
        android:layout_marginTop="20dp"
        android:layout_weight="1"
        android:background="#00FFFFFF"
        android:src="@drawable/save"
        app:layout_constraintStart_toEndOf="@+id/btnBack"
        app:layout_constraintTop_toBottomOf="@+id/txtUCost" />

    <ImageButton
        android:id="@+id/btnDel"
        style="@style/Widget.AppCompat.Button"
```

```
        android:layout_width="49dp"
        android:layout_height="48dp"
        android:layout_marginStart="52dp"
        android:layout_marginLeft="52dp"
        android:layout_marginTop="20dp"
        android:layout_weight="1"
        android:background="#00FFFFFF"
        android:src="@drawable/delete"
        app:layout_constraintStart_toEndOf="@+id/btnSave"
        app:layout_constraintTop_toBottomOf="@+id/edtUCost" />
</androidx.constraintlayout.widget.ConstraintLayout>a
```