***ΠΡΟΗΓΜΕΝΑ ΘΕΜΑΤΑ ΑΛΛΗΛΕΠΙΔΡΑΣΗΣ (ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΚΙΝΗΤΩΝ ΣΥΣΚΕΥΩΝ)***

***Η Εργασία Εξαμήνου:***

*Δημιουργία εφαρμογής εκδρομών για κινητές συσκευές*

***Link Εργασίας:***

*https://drive.google.com/file/d/16zi8Dc8Q4mvja3AdisnrUigvOq0y73\_R/view?usp=sharing*

***Μέλη Ομάδας:***

*Μπαρμπαγιάννη Ελένη 185239 ([lena.mparmpa@gmail.com](mailto:lena.mparmpa@gmail.com))*

*Ηρακλής Βασιλόπουλος 2019016(iraklis.vas01@gmail.com)*

***~~~~~~~~~~~~~~~ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ~~~~~~~~~~~~~~***

[**Εισαγωγή**](#_gtw8gjgkzrod)2

[**Περιγραφή Βάσεων**](#_qewt07t0odua)3

[Τοπική Βάση](#_hzepuuocsl5x) 3

[Απομακρυσμένη Βάση(Firestore)](#_l7et0m6pcywn) 4

[**Περιγραφή εφαρμογής**](#_j9hbhsbhzj7a)6

[Menu](#_he13t6ojqojq) 6

[Home](#_qr0jz0wmrkky) 7

[Διαχείριση Πελατών](#_8zh71hlukc9y) 8

[Ανάκτηση των στοιχείων από τη βάση και εμφάνιση σε RecyclerView](#_vkw1lo8wwean) 8

[Προσθήκη/Επεξεργασία/Διαγραφή πελατών](#_u7wqphttw9gc) 10

[Εμφάνιση των queries](#_ivdfs13fwove) 16

[Διαχείριση Εκδρομών](#_vrkymdarf4fp) 17

[Ταξιδιωτικά Γραφεία/Εκδρομές/Πακέτα Εκδρομής](#_psdhlpshuikm) 18

[Εμφάνιση Πίνακα](#_msr37qdg09bt) 23

[**Επίλογος**](#_9qwi6ciz74oi)25

# 

# ***Εισαγωγή***

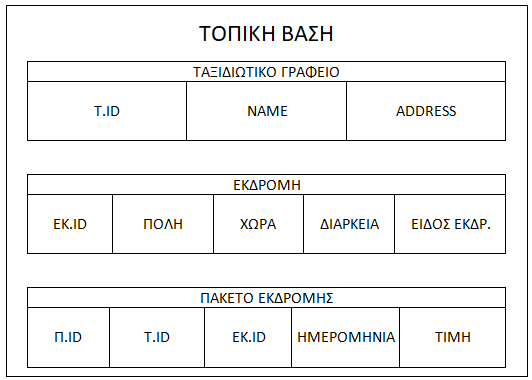
*Η εφαρμογή που υλοποιήσαμε είναι μία android εφαρμογή διαχείρισης δεδομένων ταξιδιωτικών εταιρειών. Απευθύνεται στα άτομα που θέλουν να διαχειριστούν και να επεξεργαστούν τις καταχωρήσεις που γίνονται στο σύστημα των ταξιδιωτικών εταιρειών.*

*Αρχικά, η εφαρμογή χωρίζεται σε δύο κατηγορίες, με γνώμονα τα δεδομένα που καταχωρούνται στην τοπική βάσή και αυτά που καταχωρούνται σε μία απομακρυσμένη βάση.Στην τοπική βάση, αποθηκεύονται όλα τα δεδομένα που αφορούν την ταξιδιωτική εταιρία, την εκδρομή και το πακέτο εκδρομής, ενώ στην απομακρυσμένη βάση υπάρχουν όλα τα δεδομένα που αφορούν τους πελάτες που έχουν κάνει μία κράτηση σε ένα ταξιδιωτικό γραφείο.Επιπρόσθετα, εκτός από την δυνατότητα προσθήκης δεδομένων για πελάτες και εκδρομές, ο χρήστης της εφαρμογής μπορεί να διαγράψει και να τροποποιήσει τα δεδομένα που έχει καταχωρήσει.*

*Τέλος, στον χρήστη δίνεται και η επιλογή να δει τις καταχωρήσεις του, αφού αυτές προσθέτονται σε έναν πίνακα, ταξινομημένο σύμφωνα με αυτό που ο χρήστης ψάχνει κάθε φορά.*

# ***Περιγραφή Βάσεων***

## ***Τοπική Βάση***



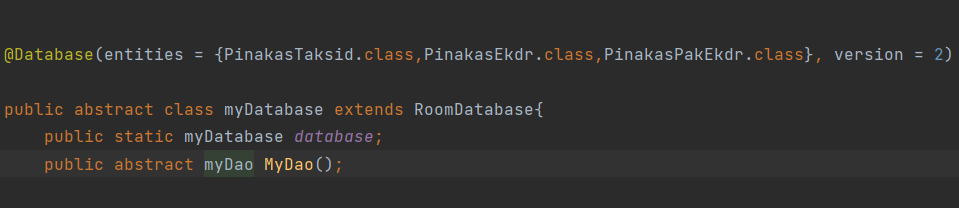
Η τοπική βάση της εφαρμογής αποτελείται από τρεις πίνακες με ονόματα: **PinakasTaksid(Ταξιδιωτικό Γραφείο)**, **PinakasEkdr(Εκδρομή)**, **PinakasPakEkdr(Πακέτο Εκδρομής)**.

Αρχικά, στον πίνακα **PinakasTaksid** έχουμε τρία πεδία ,με το ID του Ταξιδιωτικού Γραφείου(T.ID), το όνομα του Ταξιδιωτικού(Name) και την διεύθυνση του Ταξιδιωτικού(Address). Το πεδίο **Τ.ID** είναι το κύριο κλειδί του πίνακα.

Στον πίνακα **PinakasEkdr** υπάρχουν πέντε πεδία. Το ID Εκδρομής (EK.ID), η πόλη της εκδρομής , η χώρα, η διάρκεια και τέλος το είδος της εκδρομής. Το **EK.ID** είναι το κύριο κλειδί αυτού του πίνακα.

Ομοίως και στον πίνακα **PinakasPakEkdr** έχουμε πέντε πεδία, με όνομα Π.ID(ID πακέτου), T.ID(ID Ταξιδιωτικού) ,EK.ID(ID Εκδρομής), την ημερομηνία και την τιμή του πακέτου εκδρομής. Στον συγκεκριμένο πίνακα, το κύριο κλειδί είναι το Π.ID. Παρόλα αυτά, ο πίνακας αυτός συνδέεται και με τους υπόλοιπους μέσω των πεδίων T.ID και EK.ID, καθώς αποτελούν τα ξένα κλειδιά του πίνακα.

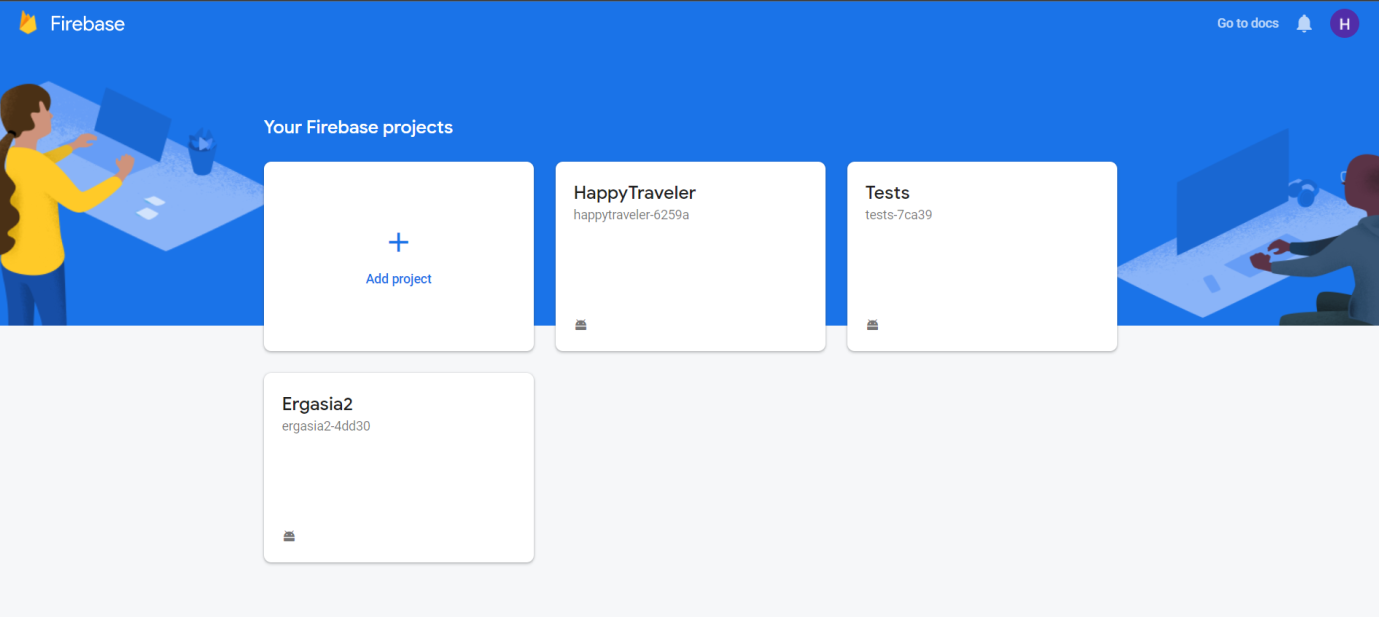
Για τον κώδικα της τοπικής βάσης (myDatabase), έχουμε:



## ***Απομακρυσμένη Βάση(Firestore)***

Η βάση δεδομένων Firestore είναι μια απομακρυσμένη βάση δεδομένων, η οποία είναι μέρος της πλατφόρμας Google Database. Μέσα σε αυτή τη πλατφόρμα προσφέρονται διάφορες λειτουργίας δημιουργίας εφαρμογών για τα κινητά και το διαδίκτυο. Η Firestore ειναι μια NoSql βάση η οποία διαφέρει απο τις σχεσιακές βάσεις. Χρησιμοποιεί διάφορα μοντέλα διαχείρισης και πρόσβασης δεδομένων όπως έγγραφα, γραφήματα, τιμές κλειδιών, μνήμη και αναζήτηση. Χαρακτηριστικό της είναι ότι χρησιμοποιείται κυρίως για την αποθήκευση μεγάλου όγκου δεδομένων. Μερικές απο τις πιο γνωστές εταιρείες που χρησιμοποιούν τη συγκεκριμένη βάση είναι οι The New York Times, Bank of America.

Η σύνδεση της βάσης με την εφαρμογή είναι αρκετά προσιτή προς τους προγραμματιστές, καθώς μπορεί να επιτευχθεί μέσα από το μενού του Android Studio, όπου το επιλέγουμε το πεδίο Tools->Firebase και στη συνέχεια στο πεδίο Assistant εμφανίζονται τα βήματα που χρειάζεται να ακολουθήσει κάποιος για να δημιουργήσει και να συνδέσει τη βάση του. Για να υπάρξει η επικοινωνία ανάμεσα στο project και στη βάση, δημιουργούμε ένα instance τύπου FirebaseFirestore. Μέσω του Firebase Console δημιουργήσαμε τη βάση HappyTraveler η οποία περιέχει τα απαραίτητα rules ώστε με τον κατάλληλο κώδικα να πραγματοποιούνται οι επιθυμητές ενέργειες από τον χρήστη. Παρακάτω βλέπουμε το μέρος όπου μπορούμε να δημιουργήσουμε ή να επιλέξουμε μια βάση, ενώ επίσης φαίνεται και η βάση HappyTraveler όπου είναι συνδεδεμένη η εφαρμογή μας.



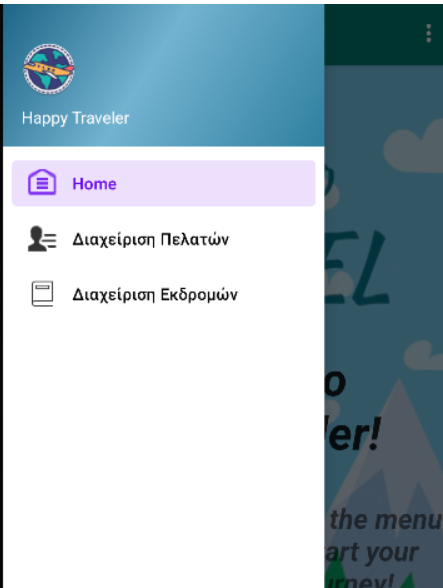
# ***Περιγραφή εφαρμογής***

## ***Menu***

Κατά τη σχεδίαση της εφαρμογής μας, αποφασίσαμε να χωρίσουμε την εφαρμογή σε δύο κύρια μέρη. Ένα μέρος αναλαμβάνει αποκλειστικά την επεξεργασία και την καταχώρηση δεδομένων με ό,τι αφορά τους πελάτες και το άλλο μέρος αντίστοιχα επεξεργάζεται τα δεδομένα που αφορούν τις εκδρομές.

*Κάτι τέτοιο επιτυγχάνεται εύκολα με το Μενού το οποίο περιλαμβάνει τρία κομμάτια:*

*Ένα για το home, ένα για τη διαχείριση πελατών και ένα για τη διαχείριση εκδρομών, όπως φαίνεται και με την παρακάτω εικόνα.*

**

*Το μενού έχει προσαρμοστεί εμφανισιακά, ώστε να εμφανίζει το εικονίδιο και το όνομα της εφαρμογής και από κάτω τα επιμέρους στοιχεία του μενού με τα αντίστοιχα εικονίδια.*

*Τα αντίστοιχα xml αρχεία βρίσκονται στον υποφάκελο layout και menu του resource φακέλου.*

*Χωρίζοντας την εφαρμογή σε 3 κομμάτια, η πλοήγηση γίνεται πιο εύκολη για τον χρήστη, διότι όλα τα επιμέρους στοιχεία βρίσκονται είτε στην διαχείριση πελατών, είτε στην διαχείριση εκδρομών, ανάλογα με τι θα ήθελε να επεξεργαστεί ο χρήστης κάθε φορά.*

## 

## ***Home***

Όπως φαίνεται στην προηγούμενη εικόνα, το πρώτο κουμπί που έχει τοποθετηθεί στο μενού είναι αυτό του home. Στο fragment αυτό δεν υπάρχει τίποτα ιδιαίτερο, πέρα από την αρχική της εφαρμογής, η οποία παραπέμπει τον χρήστη να δει το μενού για να ξεκινήσει την πλοήγησή του.

**

## 

## 

## ***Διαχείριση Πελατών***

*Στο επόμενο κουμπί του μενού έχει τοποθετηθεί η διαχείριση πελατών.*

**

Κάνοντας πλοήγηση μέσα από το μενού, με την επιλογή του του πεδίου Διαχείριση Πελατών βρίσκεται στο activity GalleryFragment, όπου εκεί έχει τη δυνατότητα ο χρήστης να κάνεις τις εξης ενέργειες. Να προσθέσει πελάτες μέσα από το κουμπί, να τους επεξεργαστεί ή να τους διαγράψει μέσα από το κουμπί επεξεργασία ενώ στη συνέχεια μπορεί να λάβει διάφορες πληροφορίες για τη βάση επιλέγοντας ένα από τα κουμπιά ταξινομημένος πίνακας, πελάτες στο ξενοδοχείο imperial και πελάτες που συμμετέχουν στην εκδρομή με ID 575.

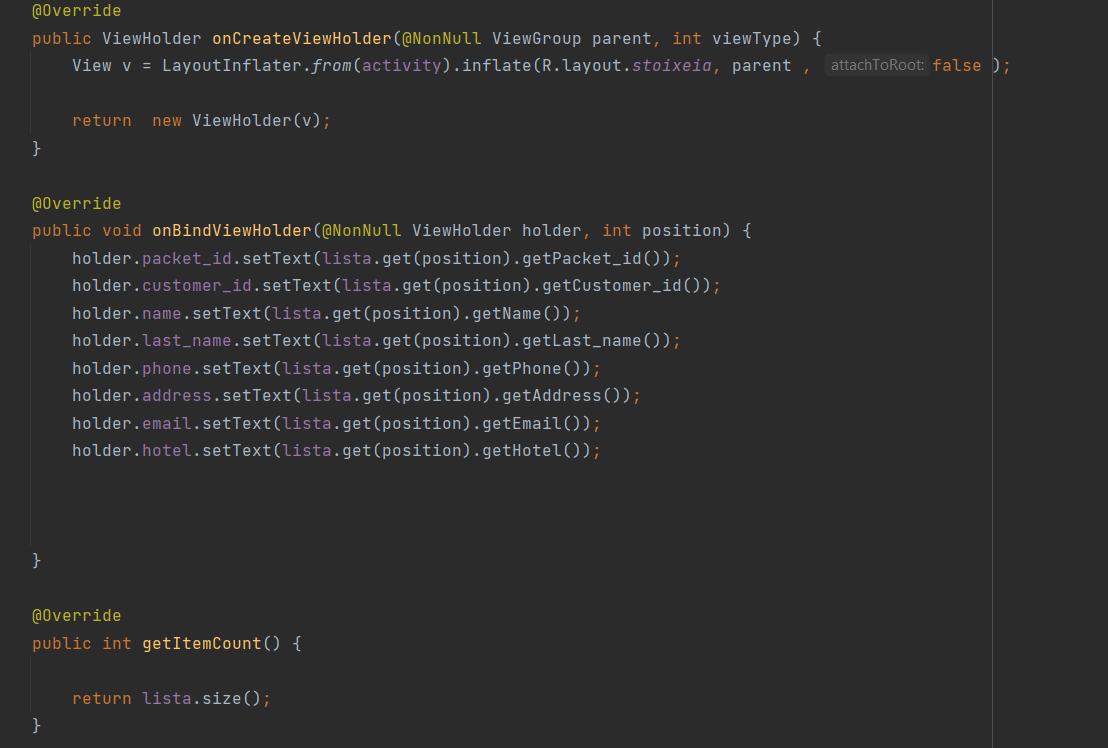
### 

### ***Ανάκτηση των στοιχείων από τη βάση και εμφάνιση σε RecyclerView***

Αρχικά δημιουργούμε ένα νέο empty activity το οποίο θα μας εμφανίσει τα στοιχεία προσθέτοντας τον κατάλληλο κώδικα στο xml αρχείο και στη συνέχεια στις κλάσεις Activity\_show, Activity\_show\_query1, Activity\_show\_query2, Activity\_show\_query3 χρησιμοποιούμε τον ίδιο κώδικα για την εμφάνιση του RecyclerView με τη διαφορά ότι κάθε φορά εμφανίζουμε διαφορετικό αποτέλεσμα.

Γενικότερα, δημιουργούμε ένα αντικείμενο recyclerView(αντίστοιχα recyclerView1, recyclerView2, recyclerView3 σε κάθε κλάση) τύπου RecyclerView το οποίο σε κάθε κλάση καλούμε τις μεθόδους SetHasFixedSize(true)(δίνει σταθερό μέγεθος στο width και height κάθε TextView) και SetLayoutManager(new LinearLayoutManager(this))(είναι υπεύθυνη για την τοποθέτηση των στοιχείων για προβολή. Στη δική μας περίπτωση εμφανίζονται περίπου σαν λίστα με τη χρήση του LinearLayoutManager). Στη συνέχεια δημιουργούμε ενα xml αρχείο το οποίο ονομάζουμε stoixeia. Μέσα σε αυτό το αρχείο δημιουργούμε τα TextView που θα εμφανίζονται στο RecyclerView.

Επόμενο βήμα είναι η δημιουργία της κλάσης Pelates η οποία περιέχεις τον constructor και όλες τις getter και setter μεθόδους των αντικειμένων που θα περάσουμε στο RecyclerView. Ακόμα, δημιουργούμε την κλάση MyAdapter(αντίστοιχα για κάθε Activity\_show\_query, MyAdapter(1,2,3)) . Στη μέθοδο ViewHolder onCreateViewHolder δημιουργούμε ένα νέο αντικείμενο τύπου View και επιστρέφουμε ένα νέο αντικείμενο ViewHolder(v) το οποίο περιγράφει τη θέση του στοιχείου μέσα στο RecyclerView, ενώ η μέθοδος getItemCount μας επιστρέφει τα στοιχεία που είναι διαθέσιμα μια συγκεκριμένη στιγμη στον adapter. Αφου στην κλάση ViewHolder δηλώσουμε τα TextView στοιχεία, στη συνέχεια βρίσκουμε τη θέση τους στο activity και τέλος μέσα απο την μέθοδο onBindViewHolder, εμφανίζουμε τα δεδομένα μας μέσα στο RecyclerView. 



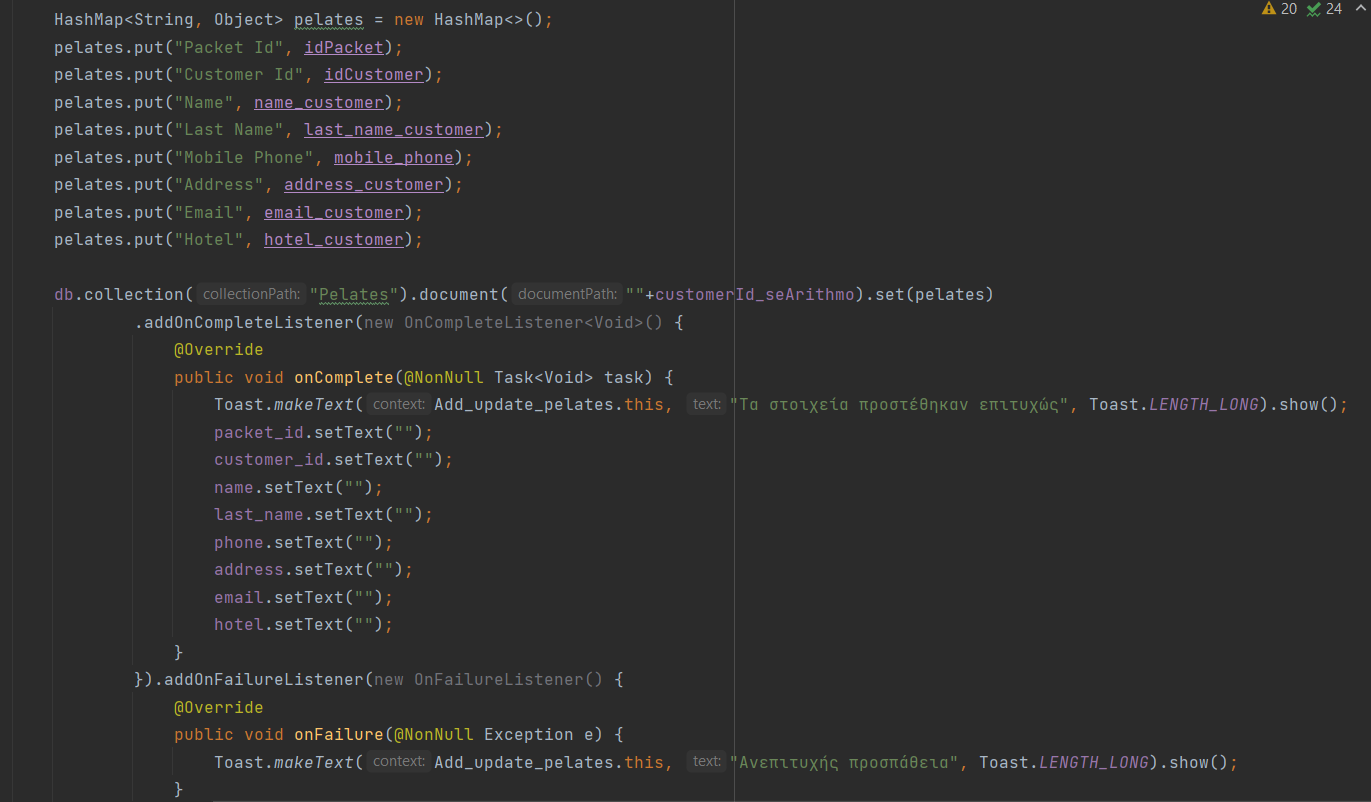
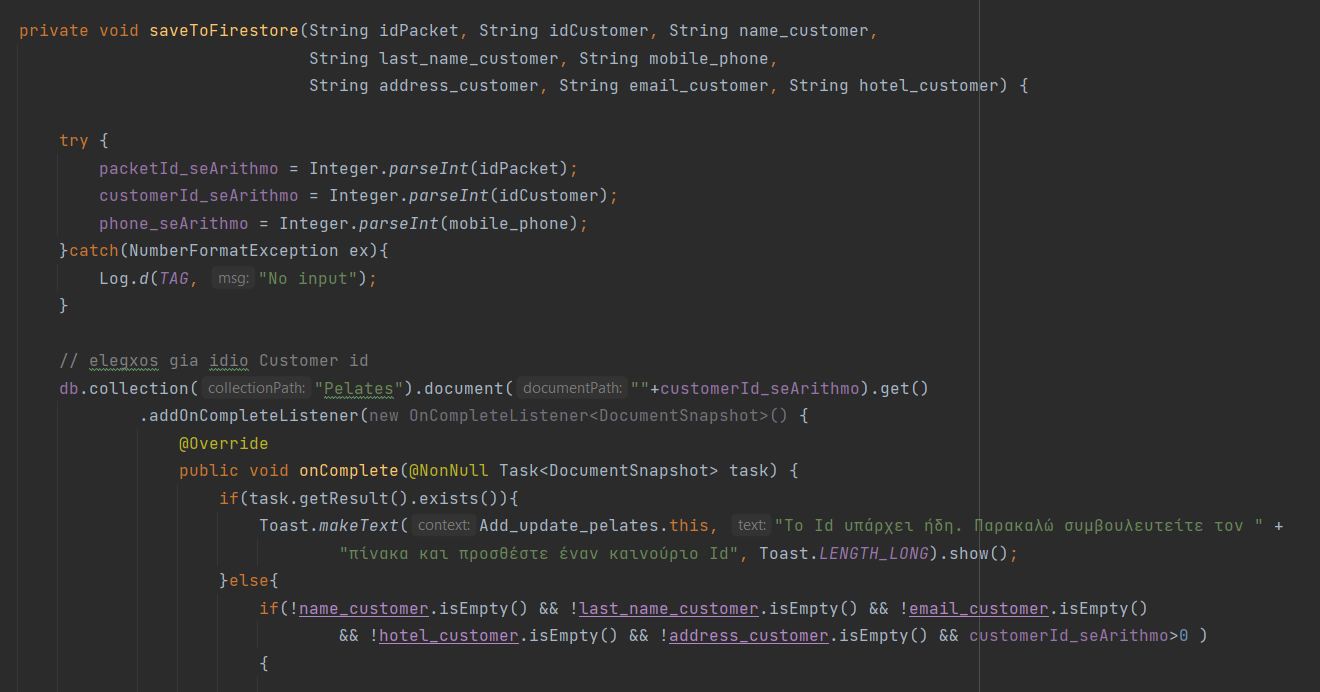
Ετσι, μέσα στο Activity Activity\_show δημιουργώντας ένα αντικείμενο adapter τύπου MyAdapter μπορούμε να πετύχουμε τη μεταφορά των δεδομένων μας από την εφαρμογή μας στο RecyclerView.

### ***Προσθήκη/Επεξεργασία/Διαγραφή πελατών***

Επιλέγοντας το κουμπί προσθήκη πελατών, ο χρήστης μεταφέρεται στο συγκεκριμένο activity.



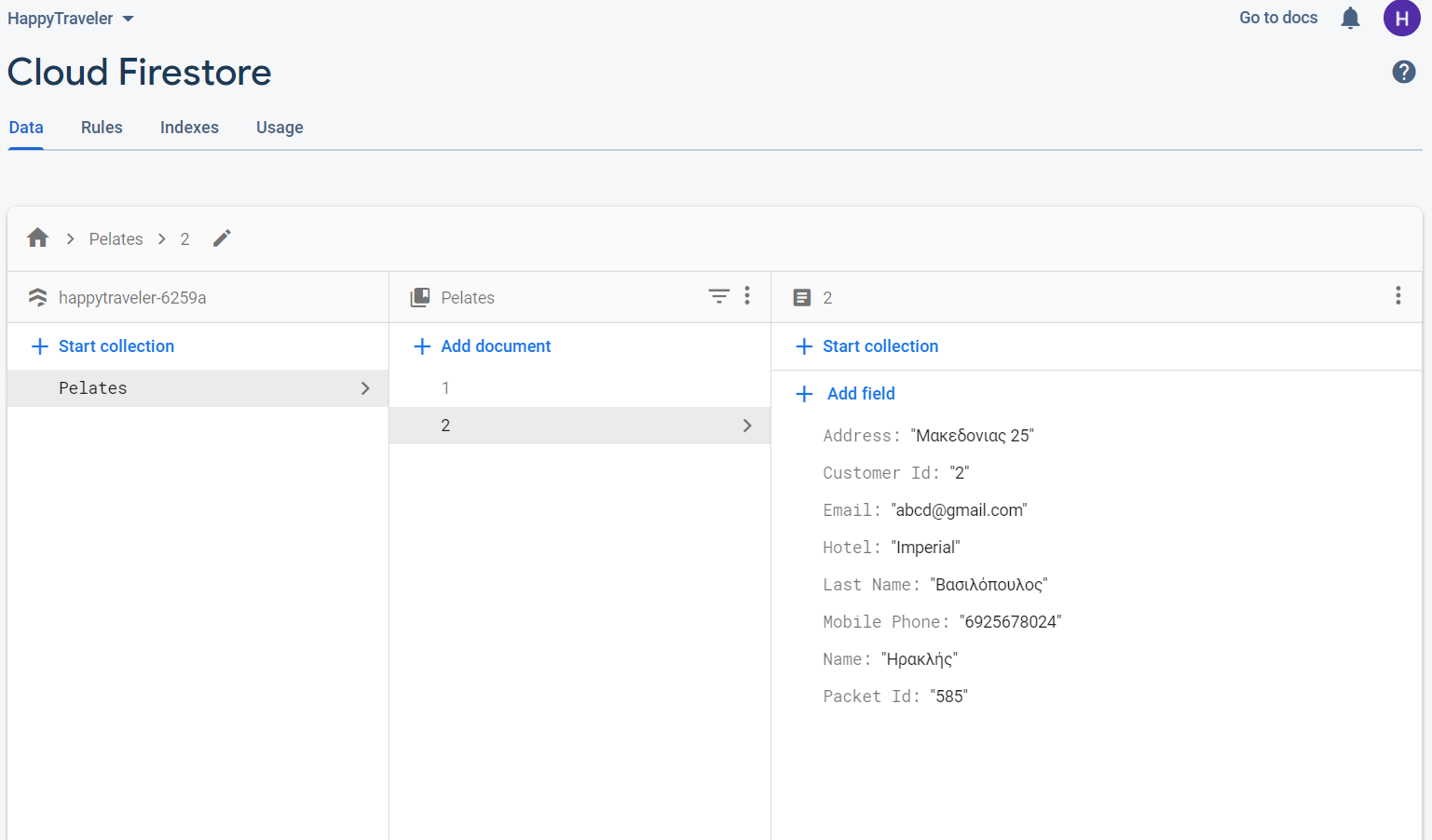
Η προσθήκη των στοιχείων των πελατών επιτυγχάνεται με τον παρακάτω κώδικα:

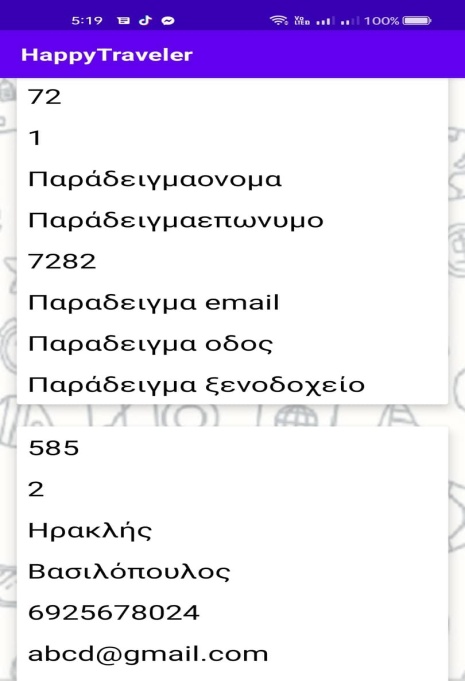


Αρχικά, χρησιμοποιούμε τις συγκεκριμένες παραμέτρους στις οποίες αποθηκεύεται το περιεχόμενο που υπάρχει μέσα στο κάθε TextView. Στη συνέχεια τα πεδία στα οποία τοποθετούνται αριθμοί, οι οποίοι γίνονται cast απο String σε int μέσα σε μια try-catch ώστε όταν δεν πληκτρολογείται κάτι από τον χρήστη μέσα στο πεδίο να μην κλείνει η εφαρμογή, αλλά να υπάρχει μια ενημέρωση. Με τη δημιουργία του Hashmap<String, Object> pelates, περνάμε μέσα στις παραμέτρους το περιεχόμενο των TextView και με την εντολή db.collection("Pelates").document(""+customerId\_seArithmo).set(pelates)

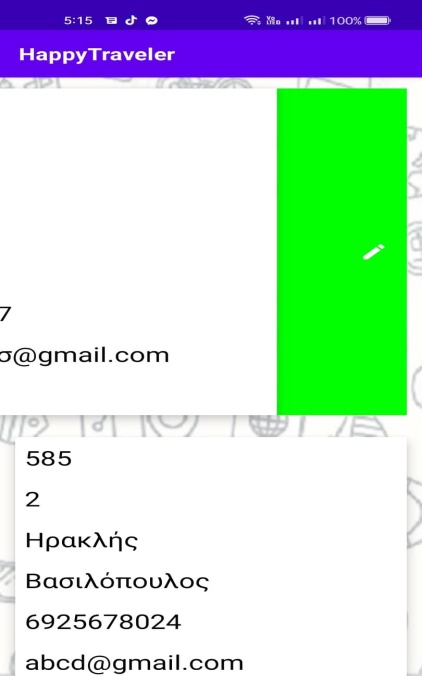
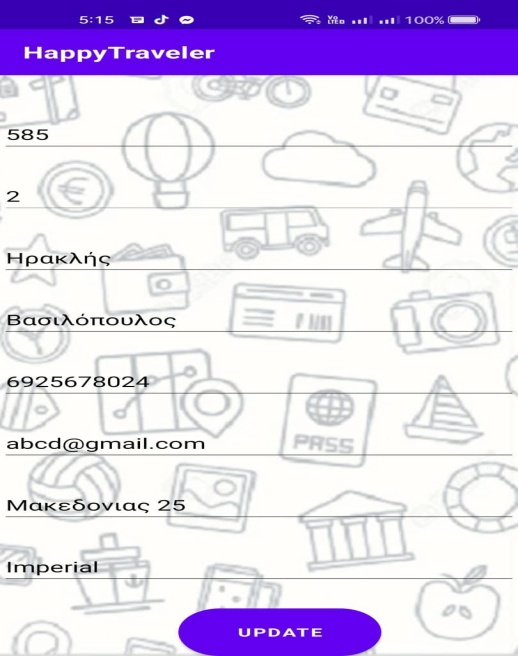
δημιουργείται η νέα συλλογή στη βάση μας, με όνομα Collection “Pelates”, με όνομα Document τη τιμή που θα δώσει ο χρήστης μέσα από το TextView και με fields Packet Id, Customer Id, Name, Last Name, Mobile Phone, Address, Email, Hotel. Αμέσως μετά την επιτυχή δημιουργία τα TextView, επανέρχονται στην αρχική τους μορφή, δηλαδή στο κενό.

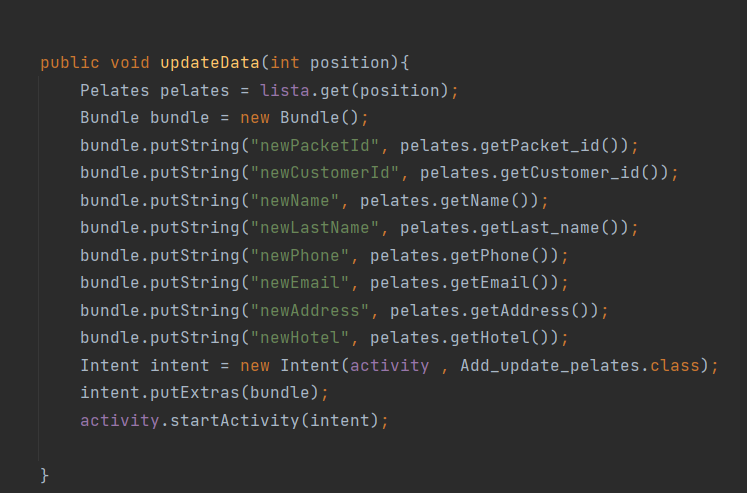
Παράδειγμα δομής της βάσης Cloud Firestore όπως αυτή εμφανίζεται στη κονσόλα της Firebase:(Αριστερά Collection, στη μέση Document, δεξιά τα πεδία με το περιεχόμενό τους)



Εάν ο χρήστης επιλέξει να κάνει ενημέρωση ή διαγραφή κάποιων στοιχείων τότε πρέπει να πατήσει το κουμπί Επεξεργασία. Πατώντας το συγκεκριμένο κουμπί, ο χρήστης μεταφέρεται σε ένα RecyclerView όπου εμφανίζονται όλα τα στοιχεία που έχουν περαστεί. 

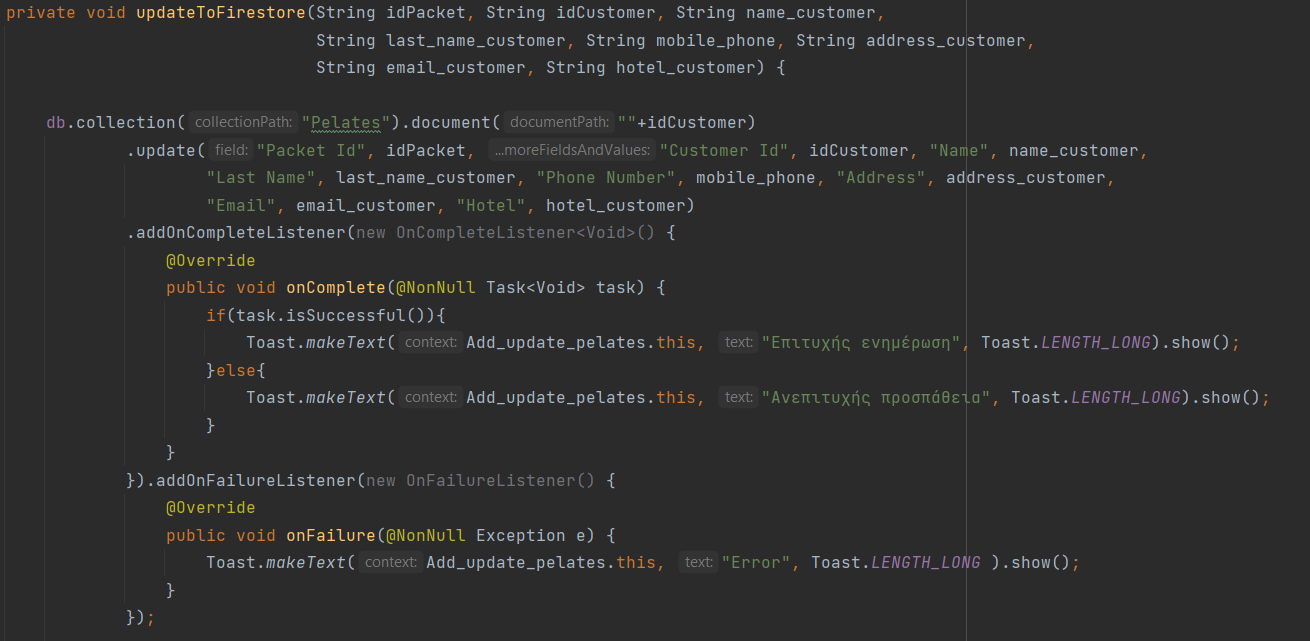
Σέρνοντας το κάθε κομμάτι στοιχείων προς τα αριστερά, ο χρήστης μπορεί να επεξεργαστεί τα στοιχεία όπου εμφανίζεται στην οθόνη την παρακάτω εικόνα:

 Στην ουσία, μεταφέρεται ξανά στο Activity Add\_update αυτή τη φορά όμως με τα στοιχεία περασμένα ήδη στο TextView ενώ στη κουμπί αναγράφεται update αντι για Add new:

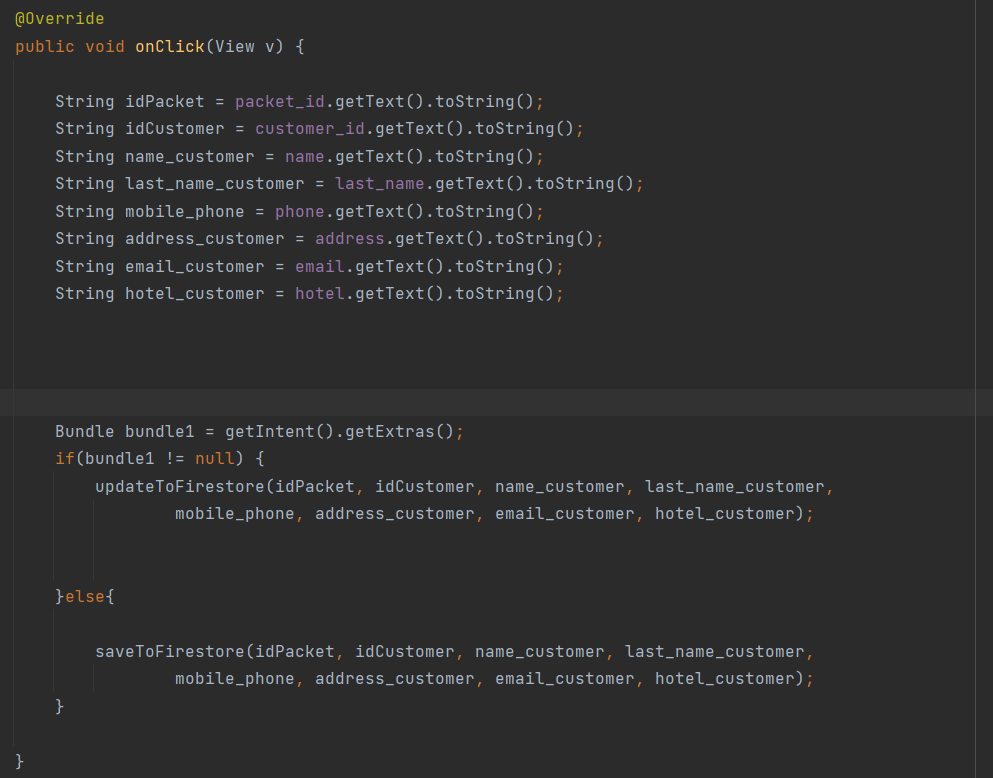
Αυτό επιτυγχάνεται με τα εξης κομμάτια κώδικα.

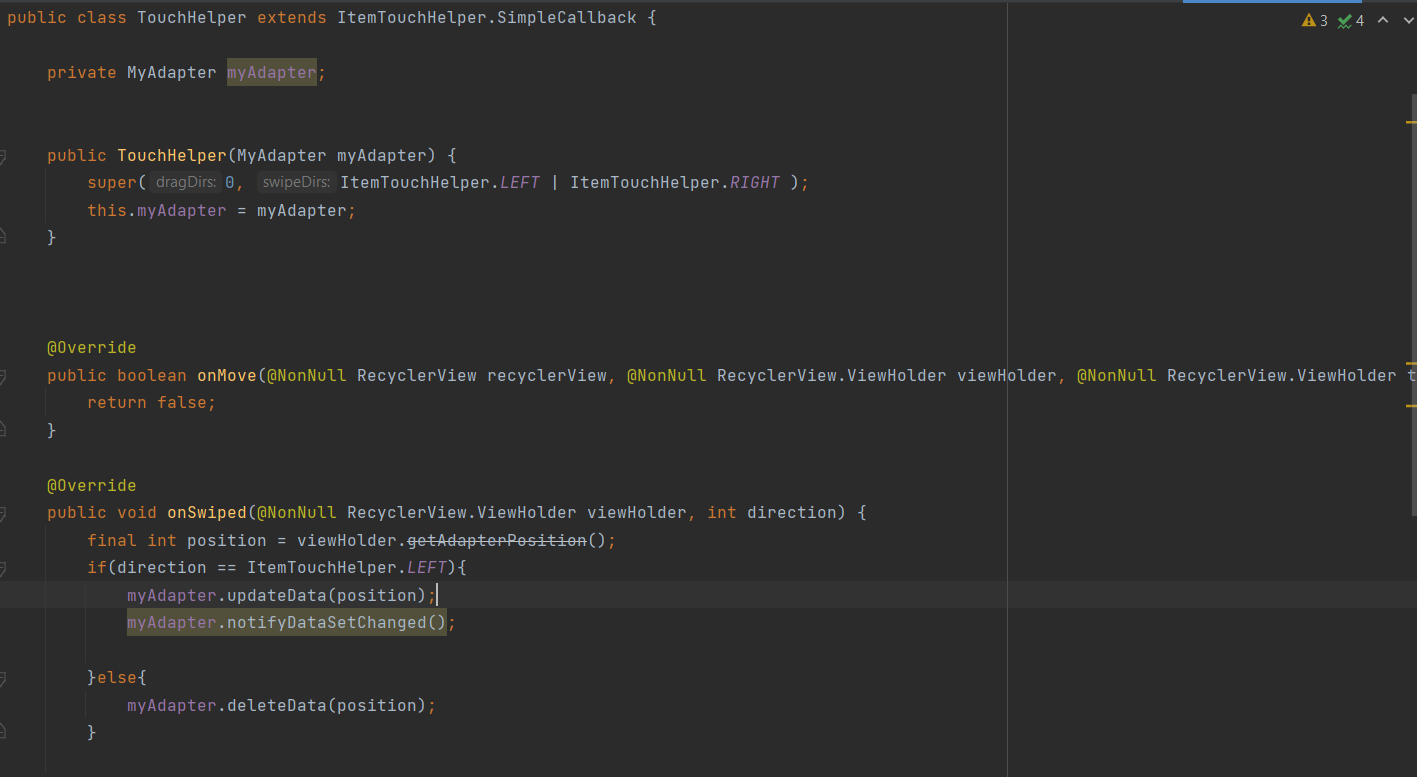
Παίρνουμε τις τιμές από το RecyclerView όταν σέρνουμε ένα μέρος του μέσα από την κλάση MyAdapter και στη συνέχεια μέσα από την κλάση Add\_update καταχωρούμε τι τιμές στις νέες μεταβλητές.



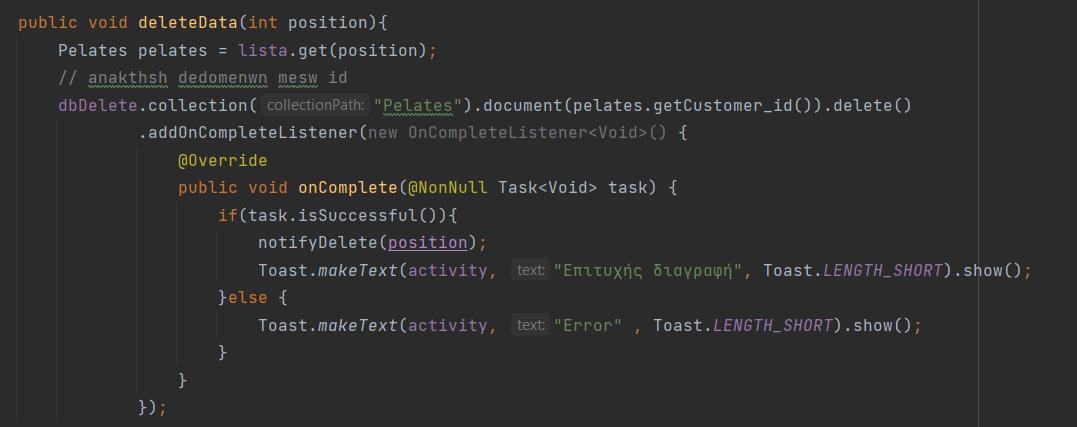
Με τον συγκεκριμένο κώδικα, εάν τα αντικείμενο bundle περιέχει τιμή, δηλαδή έχει σύρει ο χρήστης το RecyclerView προς τα αριστερά, τότε τοποθετούνται οι τιμές στα TextView και αλλάζει το κουμπί απο Add σε Update, αλλίως παραμένει όπως πριν. Παράλληλα, απενεργοποιήσαμε το κουμπί customer\_id ώστε ο χρήστης να μη μπορεί να το αλλάξει καθώς θέλουμε να είναι μοναδικό για κάθε προσθήκη. Η ενημέρωση των πεδίων γίνεται με τον εξής κώδικα:

Η μόνη διαφορά με την μέθοδο saveToFireStore ειναι οτι αντί για την μέθοδο .set() που χρησιμοποιήσαμε για την εισαγωγή των δεδομένων, τώρα χρησιμοποιήσαμε τη μέθοδο .update() για την ενημέρωση. Η επιλογή για το ποια θα είναι η εμφάνιση του Activity Add\_update γίνεται με τον παρακάτω κώδικα όπου δημιουργούμε πάλι ένα αντικείμενο Βundle και ελέγχουμε αν είναι κενό ή όχι όταν πατάμε το κουμπί προσθήκη πελατών.

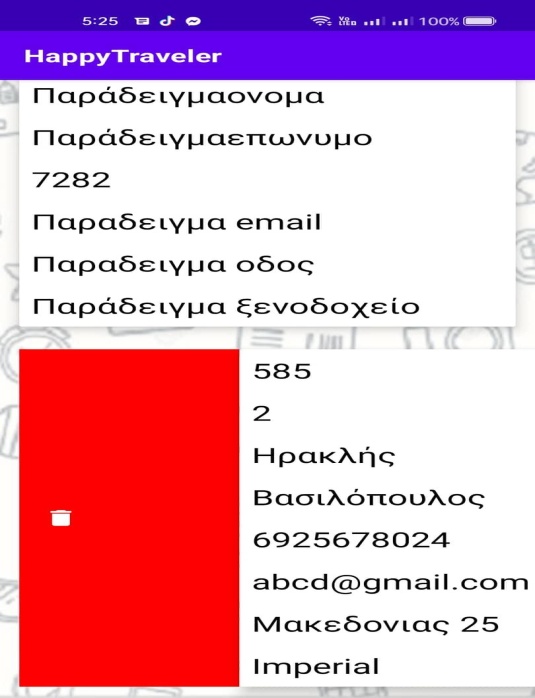


Για να διαχειριστούμε το σύρσιμο στην οθόνη προς τα δεξιά ή προς τα αριστερά δημιουργήσαμε την κλάση TouchHelper. Ο κώδικας βρίσκεται παρακάτω.

Μέσα στον constructor TouchHelper δώσαμε τις επιλογές για σύρσιμο δεξιά ή αριστερά, ενώ μέσα στη μέθοδο onSwiped, καλούμε τη μέθοδο αν σύρουμε αριστερά ώστε να ενημερωθεί η θέση που σύρουμε ενώ αν σύρουμε δεξιά γίνεται η διαγραφή ολόκληρης της θεσης. Ο κώδικας για τη διαγραφή βρίσκεται στη κλάση MyAdapter.



Με την εντολή .document(pelates.getCustomer\_id()) βρίσκουμε στη βάση το συγκεκριμένο document που έχουμε επιλέξει σέρνοντας τη θέση του RecyclerView προς τα δεξιά ο χρήστης βλέπει στην οθόνη την παρακάτω εικόνα:

και με τη μέθοδο .delete() διαγράφουμε το Document.

### ***Εμφάνιση των queries***

Επιλέγοντας το κουμπί ταξινόμηση εμφανίζεται ένα RecyclerView ταξινομημένο κατα αύξουσα σειρά. Η μέθοδος που χρησιμοποιήθηκε για το συγκεκριμένο query βρίσκεται στη κλάση activity\_show\_query1 και χρησιμοποιήθηκε η μέθοδος .orderBy(“Customer Id”, Query.Direction.ASCENDING), ώστε να ταξινομηθεί με βάση τον αριθμό των Id των πελατών.

Επιλέγοντας το κουμπί πελάτες στο ξενοδοχείο imperial εμφανίζονται οι πελάτες που έχουν κάνει κράτηση στο αντίστοιχο ξενοδοχείο. Η μέθοδος που χρησιμοποιήθηκε είναι η whereEqualTo(“Hotel”, Imperial)

Τέλος, επιλέγοντας το κουμπί πελάτες που συμμετέχουν στην εκδρομή 575 εμφανίζονται οι πελάτες που έχουν επιλέξει την εκδρομή με Packet Id 575. Η μέθοδος που χρησιμοποιήθηκε είναι η whereEqualTo(“Hotel”, Imperial). Για όλα του queries χρησιμοποιήθηκε ξεχωριστή μέθοδος MyAdapter ώστε να εμφανιστούν σε RecyclerView.

## ***Διαχείριση Εκδρομών***

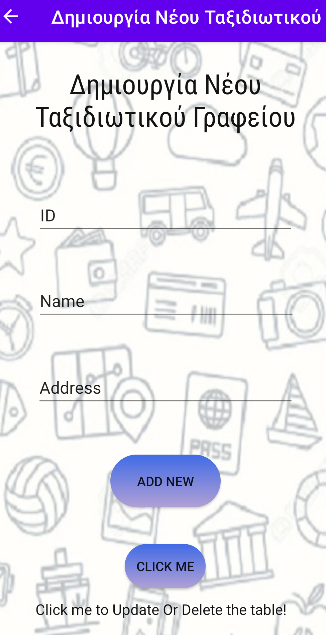
Στο τρίτο και τελευταίο κομμάτι του μενού, έχει τοποθετηθεί η διαχείριση εκδρομών. Με το πάτημά του, ο χρήστης παραπέμπεται στο κομμάτι το οποίο διαχειρίζεται οτιδήποτε αφορά τις εκδρομές και την τοπική βάση.



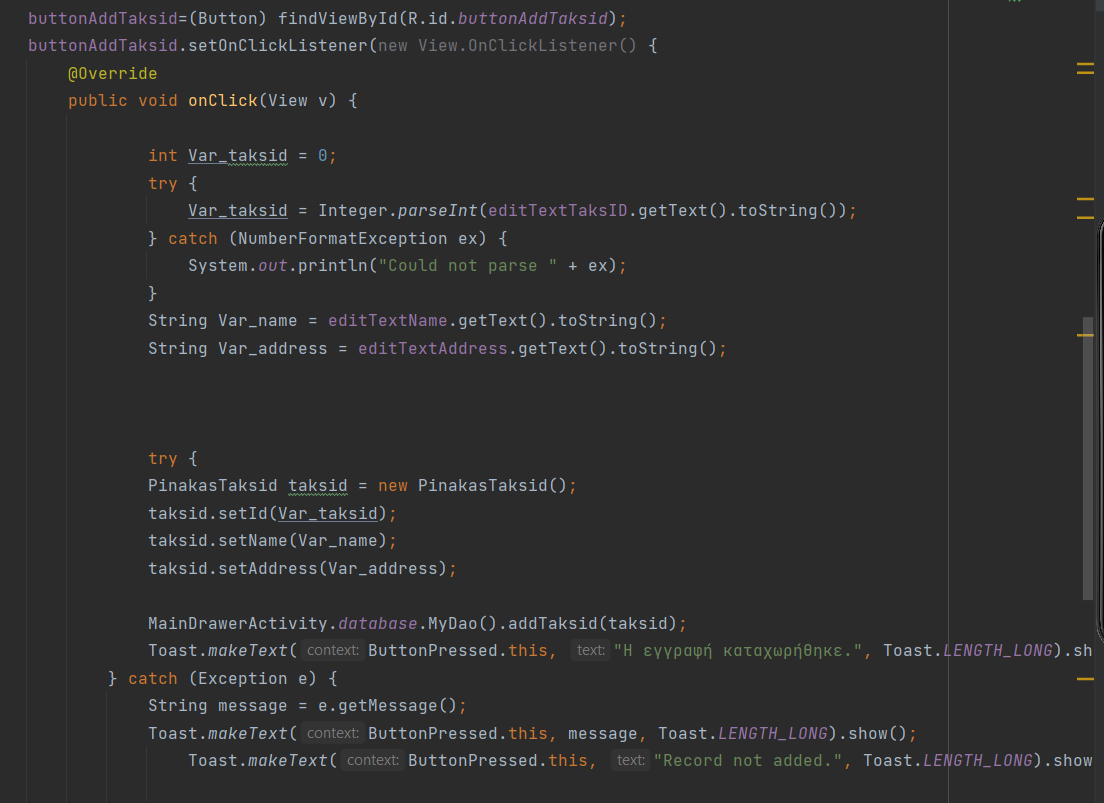
Εφόσον ο κώδικας είναι παρόμοιος στα τρία πρώτα κουμπιά, θα εξηγήσουμε την μεθοδολογία για το πρώτο κουμπί και οτιδήποτε άλλο είναι αξιοσημείωτο από τα επόμενα δύο.

### ***Ταξιδιωτικά Γραφεία/Εκδρομές/Πακέτα Εκδρομής***

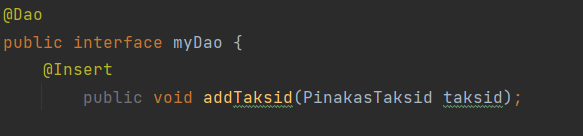
Με το πάτημα του πρώτου κουμπιού, ο χρήστης τοποθετείται στην επόμενη σελίδα (όνομα στο πρόγραμμα ButtonPressed), όπου εκεί βλέπουμε ότι υπάρχει ένα TextView με τον τίτλο και τρία ακόμη EditText που δίνουν την επιλογή στον χρήστη να τοποθετήσει το ταξιδιωτικό γραφείο που αυτός θέλει να προσθέσει, βάσει τα πεδία του πίνακα της τοπικής βάσης.



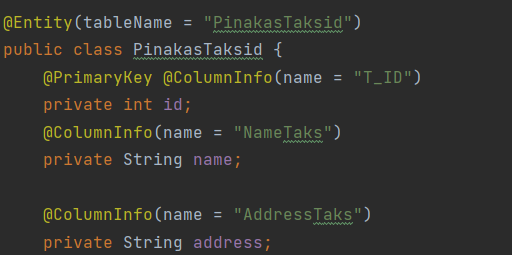
Αυτό συμβαίνει με το πάτημα του κουμπιού ADD NEW, όπου προστίθενται τα δεδομένα στη βάση, σύμφωνα με αυτόν τον κώδικα:



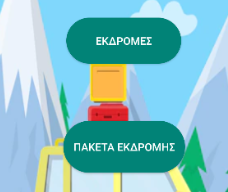
Πιο συγκεκριμένα, με το πάτημα του κουμπιού δημιουργείται ένας πίνακας PinakasTaksid, ο οποίος περιέχει τα εξής πεδία που θα συμβούν με τα interactions της βάσης με την myDao().



Ο πίνακας:



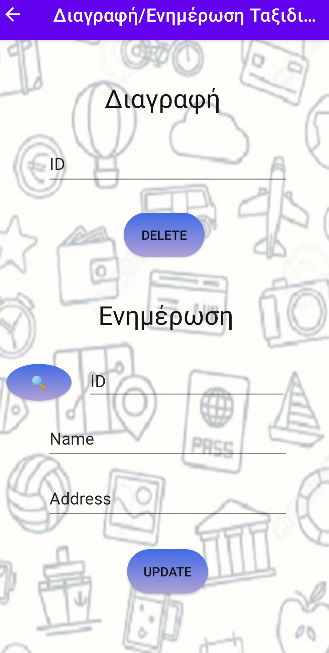
Όπου βλέπουμε ότι έχει δηλωθεί ως primary key η στήλη (T\_ID) και περιέχει τα getters και setters των πεδίων.Με αντίστοιχο τρόπο πραγματοποιείται και ο κώδικας στα επόμενα δύο κουμπιά (Εκδρομές και Πακέτα εκδρομής).



Κάτω από το κουμπί ADD NEW υπάρχει ένα ακόμη κουμπί με ένα TextView από κάτω, που παραπέμπει τον χρήστη να πατήσει το κουμπί αν θέλει να ενημερώσει ή να διαγράψει τον πίνακα.



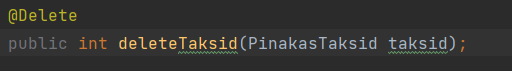
Με το πάτημα του κουμπιού CLICK ME, ο χρήστης μεταφέρεται στη σελίδα UpdateDeleteTaksid του προγράμματος, όπου βλέπουμε το εξής:



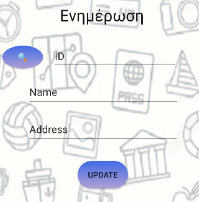
Αρχικά βλέπουμε 2 TextView , ένα για τη διαγραφή και ένα για ενημέρωση. Στη διαγραφή υπάρχει ένα EditText όπου δείχνει στον χρήστη να βάλει το ID του ταξιδιωτικού που ψάχνει, ώστε να σβήσει τον πίνακα, όταν πατήσει το κουμπί DELETE. Αυτό επιτυγχάνεται με τη χρήση αυτού του κώδικα:



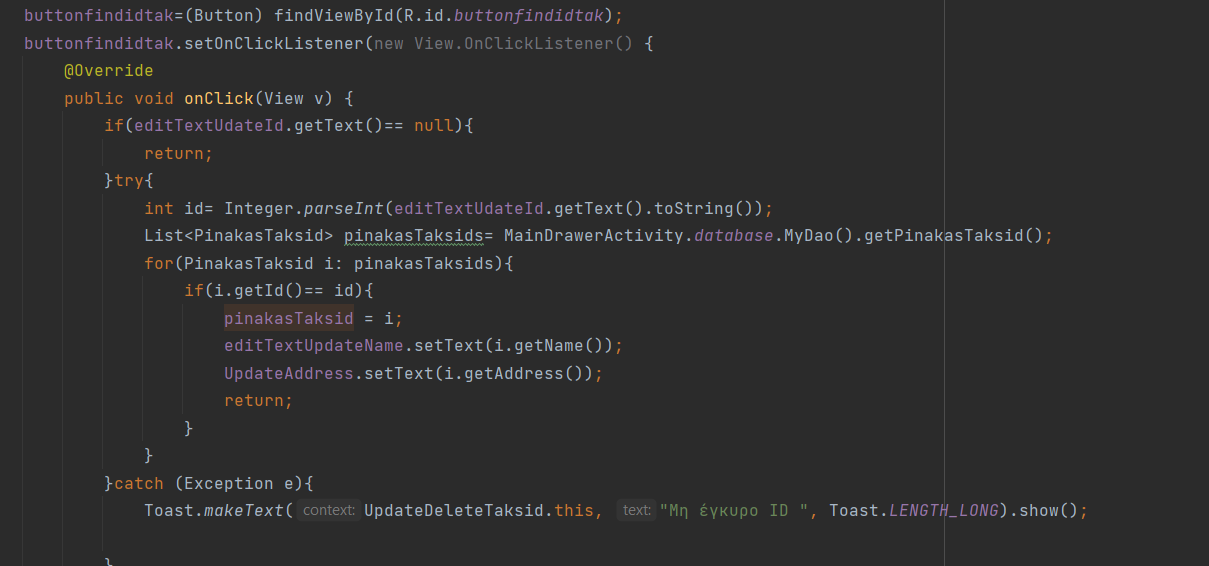
Με αντίστοιχο τρόπο με το ADD NEW κουμπί, βλέπουμε ότι για τη διαγραφή του πίνακα θα χρησιμοποιηθούν τα interactions της βάσης με την myDao() , με τον εξής τρόπο:



Στο κομμάτι της ενημέρωσης των πεδίων του πίνακα, έχουμε αρχικά 3 EditText με τα πεδία που μπορεί να ενημερώσει ο χρήστης και στη συνέχεια δύο κουμπιά.Ένα για την εύρεση και την αυτόματη συμπλήρωση των πεδίων, σύμφωνα με το ID που θα τοποθετήσει ο χρήστης και ένα για την ενημέρωση των πεδίων.

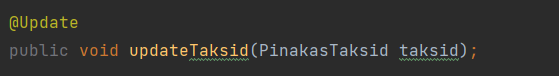


Η εύρεση του πίνακα και η αυτόματη συμπλήρωση επιτυγχάνεται με τον εξής κώδικα:

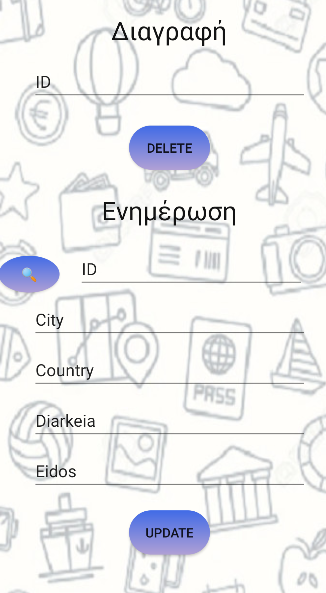
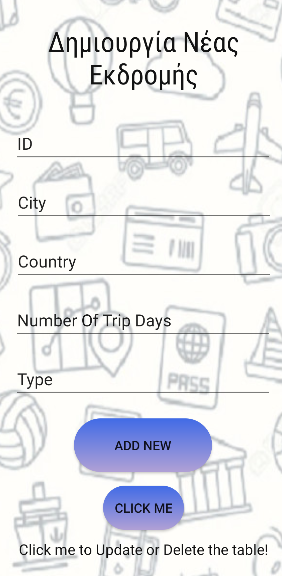
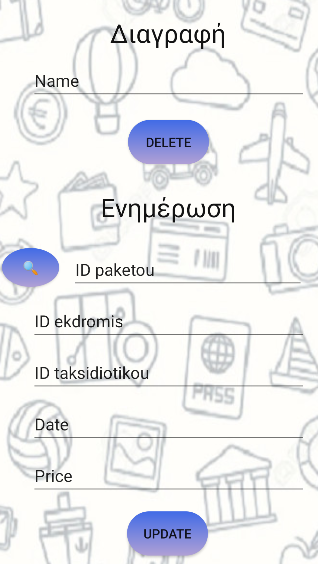
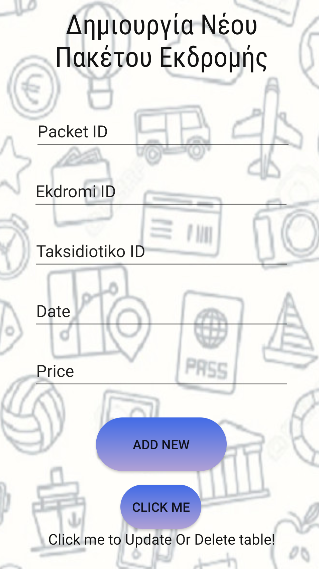


Και αντίστοιχα η ενημέρωση με αυτό το κομμάτι του κώδικα:





Όπως προαναφέρθηκε, η διαδικασία για τα επόμενα δύο κουμπιά είναι αντίστοιχη με μόνη διαφορά τα πεδία που υπάρχουν και τους πίνακες.

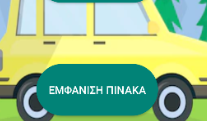


Παρόλα αυτά, κάτι που ίσως αξίζει να αναφερθεί είναι ο κώδικας του πίνακα Πακέτων Εκδρομής (PinakasPakEkdr), καθώς εκεί υπάρχουν τα ξένα κλειδιά, που συνδέουν το Πακέτο Εκδρομής με το ID της εκδρομής και του ταξιδιωτικού γραφείου. Τα ξένα κλειδιά δηλώνονται με αυτό τον τρόπο:

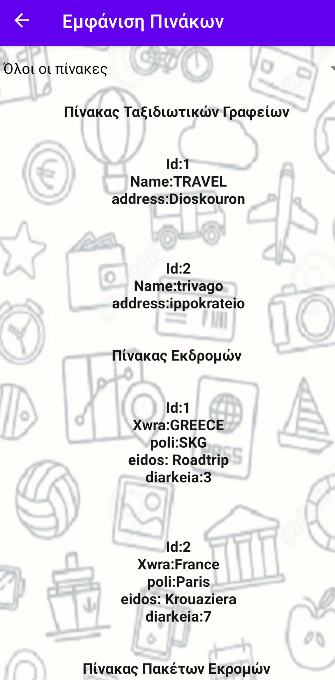


### ***Εμφάνιση Πίνακα***

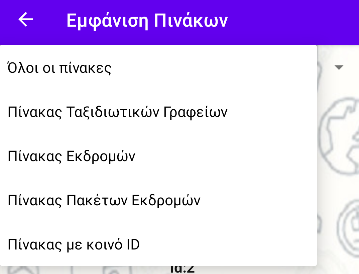
Το τελευταίο κουμπί στη Διαχείριση Εκδρομών του μενού είναι η Εμφάνιση των πινάκων.



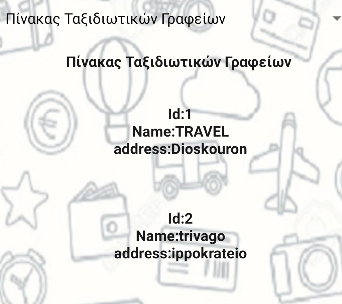
Όταν πατηθεί το κουμπί, μεταφερόμαστε στη σελίδα με όνομα ButtonPressedPinakas, όπου θα δούμε όλους τους πίνακες με ό,τι στοιχεία έχουν καταχωρηθεί στη βάση:



Παρόλα αυτά, ο χρήστης έχει τη δυνατότητα να δει και άλλα queries ,πατώντας δεξιά στο βελάκι όπου υπάρχει ένα spinner, το οποίο εμφανίζει και τα υπόλοιπα queries που έχουν καταχωρηθεί, όπως φαίνεται στην παρακάτω εικόνα:



Με το κλικ σε ένα από τα παραπάνω, το αποτέλεσμα του πίνακα θα τροποποιηθεί και θα εμφανίσει μόνο το query που αναγράφεται:

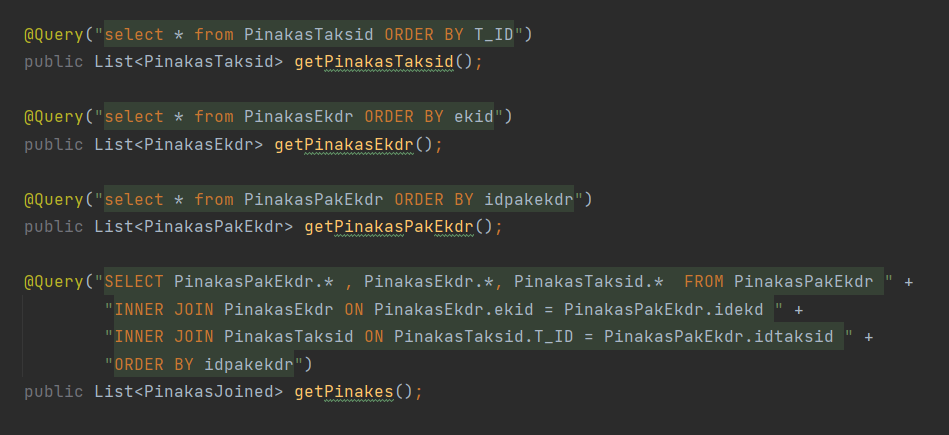


Αυτό το γεγονός επιτυγχάνεται με μία if-else συνθήκη, όπου ανάλογα το position στο οποίο βρίσκεται κάθε φορά, εκτελείται και ο αντίστοιχος κώδικας, όπως επόμενο παράδειγμα για το position1 (εμφάνιση ταξιδιωτικών γραφείων):



### 

Στη myDao(), τα queries που έχουν προστεθεί είναι τα εξής:



# ***Επίλογος***

*Με το πέρας της εργασίας μας και έπειτα από πολύωρη ενασχόληση με αυτή ,αποκομίσαμε αρκετά και σημαντικά πράγματα. Αρχικά, κατανοήσαμε πως λειτουργεί το Android Studio και πως να δημιουργούμε τοπικές και απομακρυσμένες βάσεις σε αυτό. Συνδυάσαμε τις γνώσεις που είχαμε για να δημιουργήσουμε και μια λειτουργική, αλλά και μία σχεδιαστικά όμορφη εφαρμογή. Τέλος, η εργασία αυτή μας προσέφερε πολλές γνώσεις πάνω στον προγραμματισμό κινητών συσκευών, ο οποίος είναι ένας πολύ σημαντικός τομέας του επαγγέλματος, καθώς οι κινητές συσκευές βρίσκονται πλέον παντού στην κοινωνία μας.*