ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

«Εφαρμογή κινητής συσκευής και σχετικού πληροφοριακού συστήματος για το νομό Κοζάνης»

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΛΙΤΣΑΡΔΟΠΟΥΛΟΥ ΝΑΠΟΛΕΩΝ

(ΑΕΜ: 1320)

Επιβλέπων : Δημόκας Νικόλαος

Επίκουρος καθηγητής

Καστοριά Οκτώβριος 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

«Εφαρμογή κινητής συσκευής και σχετικού πληροφοριακού συστήματος για το νομό Κοζάνης»

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΛΙΤΣΑΡΔΟΠΟΥΛΟΥ ΝΑΠΟΛΕΩΝ

(ΑΕΜ: 1320)

Επιβλέπων : Δημόκας Νικόλαος

Επίκουρος καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 18/10/2024

.....	.....	.....
Δημόκας Νικόλαος Επίκουρος Καθηγητής (Επιβλέπων)	Ιωάννης Σινάτκας, Καθηγητής	Ιωάννης Τουλόπουλος, Επίκουρος Καθηγητής

Καστοριά Οκτώβριος 2024

Ευχαριστίες

Θα ήθελα να εκφράσω την ευγνωμοσύνη μου προς τον επιβλέποντα καθηγητή μου κύριο Δημόκα Νικολάο για την υποστήριξή του, τις συμβουλές και την καθοδήγηση του κατά τη διάρκεια αυτής της πορείας. Χωρίς τη βοήθειά σας, δεν θα ήταν δυνατή η ολοκλήρωση αυτής της σημαντικής φάσης της ακαδημαϊκής μου πορείας.

Επιπλέον, θέλω να ευχαριστήσω την οικογένειά μου και τους φίλους μου για την υπομονή τους και τη στήριξή τους καθ' όλη τη διάρκεια αυτής της περιπέτειας και ιδιαιτέρως τη γυναίκα μου η οποία τόσα χρόνια έχει σταθεί δίπλα μου σε ότι έχω κάνει.

Περίληψη

Στο πλαίσιο αυτής της πτυχιακής εργασίας δημιουργήθηκε μια εφαρμογή Android για smartphones συσκευές με όνομα Travel Kozani που αφορά τον νομό Κοζάνης. Η εφαρμογή έχει σχεδιαστεί με στόχο την παροχή πληροφοριών στους χρήστες σχετικά με αξιοθέατα, δραστηριότητες, εστιατόρια, καφετέριες, μπαρ και ξενοδοχεία της πόλης.

Μέσω της εφαρμογής, οι χρήστες μπορούν να αναζητήσουν πληροφορίες για τα δημοφιλή αξιοθέατα της περιοχής, να ενημερώνονται για πιθανές δραστηριότητες που μπορούν να πραγματοποιήσουν κατά την επίσκεψη τους στη πόλη της Κοζάνης, να ανακαλύψουν τα καλύτερα εστιατόρια, καφετέριες και μπαρ της περιοχής για να διασκεδάσουν καθώς και να εντοπίσουν και να κάνουν κράτηση σε ξενοδοχεία για τη διαμονή τους. Μπορούν επίσης να πλοηγηθούν μέσω χάρτη σε διάφορα σημεία ενδιαφέροντος της πόλης.

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκαν σύγχρονες τεχνολογίες ανάπτυξης λογισμικού. Συγκεκριμένα για το android app χρησιμοποιήθηκε το Android Studio, η γλώσσα προγραμματισμού JAVA καθώς και η βιβλιοθήκη Retrofit. Η Βάση δεδομένων έγινε στο MysQL Workbench και το εργαλείο Hibernate και τέλος για τον server χρησιμοποιήθηκε το IntelliJ IDEA Ultimate. Τέλος για τη διασύνδεση και τη δημιουργία web services χρησιμοποιήθηκε το αρχιτεκτονικό στυλ REST API του οποίου η σωστή λειτουργία δοκιμάστηκε στο POSTMAN.

Στη πράξη η εφαρμογή επικοινωνεί μέσω του REST API και λαμβάνει όλες τις πληροφορίες από τη βάση δεδομένων η οποία διαθέτει όλους τους κατάλληλους πίνακες για κάθε δεδομένο της εφαρμογής. Κάθε φορά που ενημερώνεται η αλλάζει κάποιο στοιχείο της βάσης, αυτόματα ενημερώνεται και η εφαρμογή.

Λέξεις Κλειδιά: KOZANI TRAVEL ,KOZANI,ANDROID APP

Abstract

In the context of this paper, an Android application for smartphone was created called Travel Kozani, which concerns the prefecture of Kozani. The application is designed to provide information to users about attractions, activities, restaurants, cafes, bars and hotels in the city.

Through the application, users can search for information about the popular attractions of the area, be informed about possible activities they can carry out during their visit to the city of Kozani, discover the best restaurants, cafes and bars in the area to have fun as well as locate and book hotels for their stay. They can also be navigated via map to various points of interest in the city.

Modern software development technologies were used to develop the application. Specifically, Android Studio, the JAVA programming language and the Retrofit library were used for the android app. The Database was made in MysQL Workbench and the Hibernate tool and finally for the server IntelliJ IDEA Ultimate was used. Finally, the REST API architectural style was used for the interface and the creation of web services, the correct operation of which was tested in POSTMAN.

In practice the application communicates through the REST API and receives all the information from the database which has all the appropriate tables for each data of the application. Every time an element of the base is updated or changed, the application is automatically updated as well.

Key Words: KOZANI TRAVEL ,KOZANI,ANDROID APP

Πίνακας Περιεχομένων

Λίστα Εικόνων

Λίστα Πινάκων

1. Εισαγωγή

Στόχος της πτυχιακής ήταν να δημιουργήσουμε μια εφαρμογή Android τουριστικού οδηγού για την πόλη της Κοζάνης. Ο χρήστης να έχει τη δυνατότητα χρησιμοποιώντας την εφαρμογή μας να μπορεί να λάβει όλες τις χρήσιμες πληροφορίες για σημεία ενδιαφέροντος της πόλης καθώς επίσης να προηγηθεί σε αυτά.

Σε πρώτο στάδιο δημιουργήσαμε μέσω του android studio και της JAVA, το κυρίως σώμα της εφαρμογής. Έγινε ερευνά για τη πόλη της Κοζάνης ώστε να υπάρχουν πάρα πολλά δεδομένα και στοιχεία για την εφαρμογή.

Σε δεύτερο στάδιο φτιάξαμε μια βάση δεδομένων ώστε να περάσουμε εκεί όλα τα δεδομένα και τις πληροφορίες και να είναι μελλοντικά πολύ πιο απλό και γρήγορο να τροποποιήσεις και να προσθέσεις διαφορά στοιχεία.

Σε τρίτο και τελευταίο στάδιο έπρεπε να πραγματοποιήσουμε τη διασύνδεση μεταξύ της βάσης δεδομένων και της εφαρμογής Android μέσω Web Services ώστε κάθε αλλαγή στη βάση δεδομένων να ενημερώνεται αμέσως στην εφαρμογή που θα έχει ο χρήστης χωρίς να χρειάζεται κάποια ενημέρωση αυτής.

2. Βιβλιογραφική Ανασκόπηση

2.1 Προγραμματισμός λογισμικού

Ο προγραμματισμός λογισμικού είναι η διαδικασία εκείνη κατά την οποία οι μηχανικοί λογισμικού χρησιμοποιώντας διάφορα εργαλεία αλλά και γλώσσες προγραμματισμού σχεδιάζουν, δημιουργούν, αναπτύσσουν και διαχειρίζονται εφαρμογές υπολογιστών. Η υλοποίηση αυτών των εφαρμογών έχει ως στόχο την επίλυση προβλημάτων εκτελώντας συγκεκριμένες λειτουργίες.

Εν έτη 2024 ο προγραμματισμός λογισμικού έχει γίνει πιο αναγκαίος από ποτέ. Η εξέλιξη της τεχνολογίας έχει βάλει για τα καλά στη ζωή μας τον προγραμματισμό είτε αυτό αφορά τη διασκέδαση και την ψυχαγωγία είτε την καθημερινότητα και την επιχειρηματικότητα. Όλα γίνονται πιο αυτοματοποιημένα και πιο αποδοτικά. Αυτό από την άλλη έχει φέρει και μεγαλύτερη ανάγκη για εξειδικευμένες γνώσεις και δεξιότητες από τους προγραμματιστές.

2.1.1 Προγραμματισμός λογισμικού σε Android συσκευές

Ως προγραμματισμός λογισμικού σε Android συσκευές ορίζεται η διαδικασία ανάπτυξης εφαρμογών για το λειτουργικό σύστημα Android της Google. Είναι ένα από τα πιο δημοφιλή λειτουργικά συστήματα για κινητές συσκευές (smartphones,tablet κτλπ) και κατέχει μεγάλο μέρος της αγοράς των συσκευών αυτών. Η ανάπτυξη εφαρμογών για αυτό δίνει την δυνατότητα δημιουργίας μεγάλης ποικιλίας εφαρμογών από απλά εργαλεία μέχρι απαιτητικά παιχνίδια [1].

Οι δυο γλώσσες προγραμματισμού που χρησιμοποιούνται για τον προγραμματισμό λογισμικού σε Android είναι:

1) JAVA
2) Kotlin

2.1.2 Ιστορική αναδρομή

Οι Andy Rubin, Rich Miner, Nick Sears και Chris White ίδρυσαν το 2003 την Android Inc. Κύριος στόχος τους ήταν η ανάπτυξη ενός λογισμικού για κινητές συσκευές.

Δυο χρόνια αργότερα το 2005 η Google εξαγόρασε την Android Inc. Και ξεκίνησε την ανάπτυξη ενός λειτουργικού συστήματος βασισμένο στον πυρήνα του LINUX.

Το Νοέμβριο του 2007, η Google ανακοίνωσε και το λογισμικό της με όνομα Android, ένα λογισμικό ανοιχτού κώδικα. Εκτός αυτού η Google παρουσίασε την OHA (OPEN HANDSET ALLIANCE), μια κοινοπραξία 84 εταιριών. Η κοινοπραξία αυτή έπαιξε πολύ σημαντικό ρόλο στην μετέπειτα εξέλιξη του android και έφερε πολλές καινοτομίες στην κινητή τηλεφωνία. Samsung, Sony, LG, HTC, NVIDIA, Intel, Qualcomm, eBay, Vodafone είναι επιγραμματικά κάποιες από τις 84 εταιρείες που συνεργάστηκαν με την Google μέσω τις κοινοπραξίας.

Το 2008 κυκλοφόρησε η πρώτη έκδοση του Android (Android 1.0) καθώς και του πρώτου Android smartphone HTC Dream.

2.1.3 Εκδόσεις Android

ΟΝΟΜΑ	ΕΚΔΟΣΗ	ΗΜΕΡΟΜΗΝΙΑ ΚΥΚΛΟΦΟΡΙΑΣ
Alpha	1.0	23 Σεπτεμβρίου 2008
Beta	1.1	9 Φεβρουάριου 2009
Cupcake	1.5	27 Απριλίου 2009
Donut	1.6	15 Σεπτεμβρίου 2009
Eclair	2.0 - 2.1	26 Οκτωβρίου 2009
Froyo	2.2 – 2.2.3	20 Μαΐου 2010
Gingerbread	2.3 – 2.3.7	6 Δεκεμβρίου 2010
Honeycomb	3.0 – 3.2.6	22 Φεβρουάριου 2011
Ice Cream Sandwich	4.0 – 4.0.4	18 Οκτωβρίου 2011
Jelly Bean	4.1 – 4.3.1	9 Ιουλίου 2012
Kit Kat	4.4 – 4.4.4	31 Οκτωβρίου 2013
Lollipop	5.0 – 5.1.1	12 Νοεμβρίου 2014
Marshmallow	6.0 – 6.0.1	5 Οκτωβρίου 2015

Nougat	7.0 – 7.1.2	22 Αυγούστου 2016
Oreo	8.0 – 8.1	21 Αυγούστου 2017
Pie	9.0	6 Αυγούστου 2018
Q	10.0	3 Σεπτεμβρίου 2019
R	11.0	19 Φεβρουαρίου 2020
Android 12	12.0	4 Οκτωβρίου 2021
Android 12L	12.1(Pixel)	7 Μαρτίου 2022
Android 13	13.0	15 Αυγούστου 2022

Πίνακας 1 : Εκδόσεις Android

2.2 Εφαρμογές Τουριστικού Οδηγού

Οι εφαρμογές τουριστικού οδηγού είναι ψηφιακά εργαλεία που παρέχουν πληροφορίες και υπηρεσίες για την οργάνωση και βελτιστοποίηση των ταξιδιών. Αυτές οι εφαρμογές προσφέρουν δυνατότητες όπως κρατήσεις ξενοδοχείων, πτήσεων και ενοικιαζόμενων αυτοκινήτων, αναζήτηση και σύγκριση τιμών, κριτικές και αξιολογήσεις από άλλους ταξιδιώτες, καθώς και προτάσεις για αξιοθέατα, εστιατόρια και δραστηριότητες. Η χρησιμότητά τους έγκειται στην παροχή ολοκληρωμένων και εύχρηστων λύσεων για τον προγραμματισμό ταξιδιών, καθιστώντας τις διακοπές πιο οργανωμένες και απολαυστικές.

2.2.1 TripAdvisor

Το TripAdvisor είναι μια από τις μεγαλύτερες ταξιδιωτικές πλατφόρμες στον κόσμο, προσφέροντας μια τεράστια γκάμα λειτουργικοτήτων όπως συγκρίσεις τιμών και συμβουλές για ταξίδια. Η πλατφόρμα επιτρέπει στους χρήστες να αναζητούν : κριτικές και βαθμολογίες για ξενοδοχεία, εστιατόρια. Αναζήτηση και σύγκριση τιμών για πτήσεις καθώς και ενοικιάσεις αυτοκινήτων. Δυνατότητα κράτησης ξενοδοχείων, εστιατορίων και δραστηριοτήτων απευθείας από την πλατφόρμα. Η εταιρεία ιδρύθηκε τον Φεβρουάριο του 2000 στην Μασαχουσέτη των ΗΠΑ από τους Λάνγκλει Στάινερτ , Στίβεν Κάουφερ , Τομ Πάλκα και Νικ Σάννυ με στόχο να προσφέρει σε κάθε ταξιδιώτη την δυνατότητα να μπορεί να διαβάζει και να γράφει κριτικές για την εμπειρία του από κάθε ταξίδι. Στην εφαρμογή υπέχουν επίσης χάρτες και εργαλεία για καλύτερη πλοήγηση και προγραμματισμό του κάθε ταξιδιού. Τέλος δίνει την δυνατότητα δημιουργίας

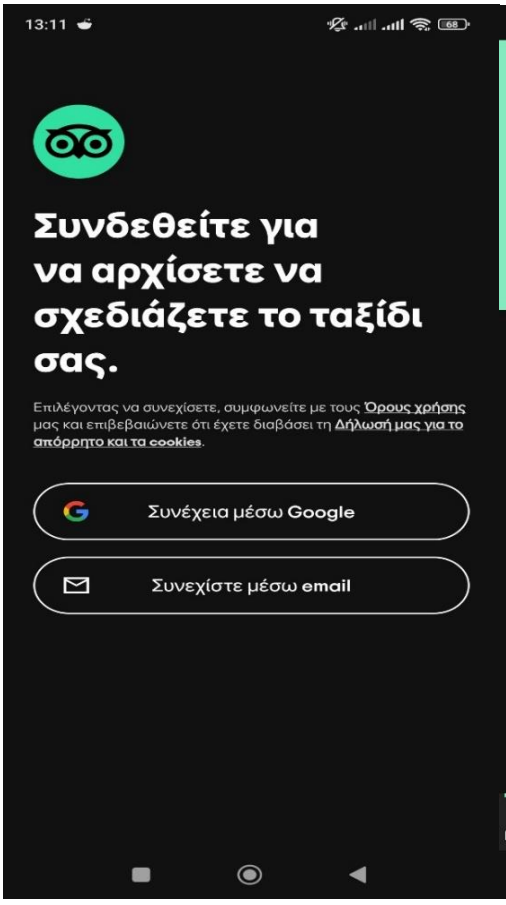
προσωπικού προφίλ όπου ο κάθε χρήστης μπορεί να αποθηκεύσει αγαπημένα μέρη για μελλοντικά ταξίδια[4].

Αυτές οι λειτουργίες καθιστούν το TripAdvisor , ένα απαραίτητο εργαλείο για κάθε ταξιδιώτη που αναζητά ευκολία σε πληροφορίες για οποιοδήποτε προορισμό.

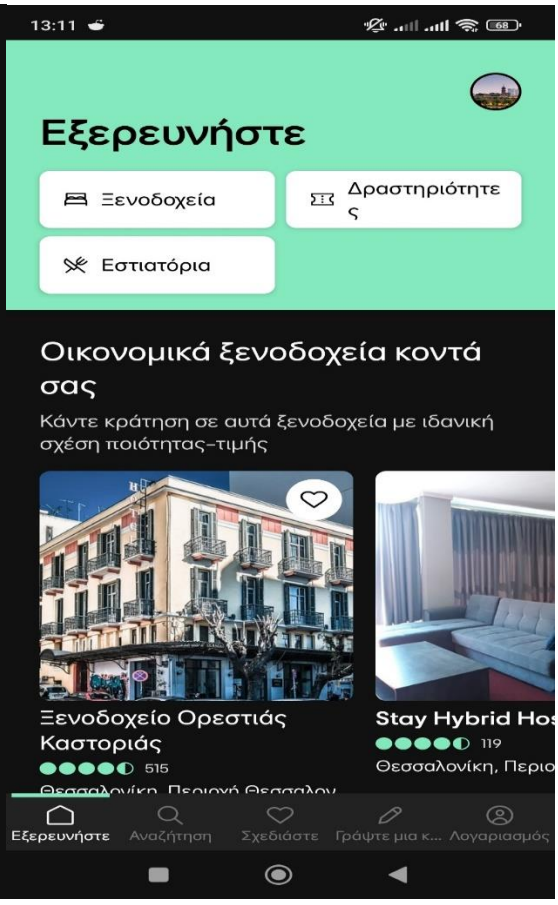
2.2.2 Τεχνολογίες που χρησιμοποιούνται στο TripAdvisor

Όπως είναι φυσικό η δημιουργία μιας παγκοσμίου εμβέλειας εφαρμογής όπως το TripAdvisor απαιτεί τη χρήση πάρα πολλών τεχνολογιών. Ονομαστικά κάποιες από αυτές είναι: Swift, JAVA, Kotlin, Flutter, React Native, AWS, Firebase, SQLite, Google maps API, GPS and Geolocation services κλπ.

2.2.3 Περιήγηση στην εφαρμογή



Εικόνα 2.1 Αρχική οθόνη TripAdvisor

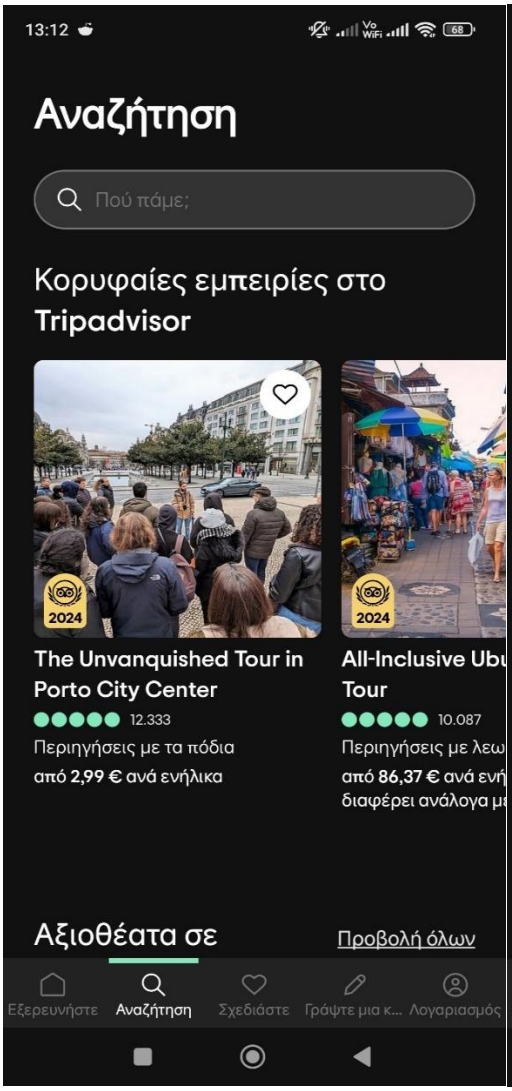


Εικόνα 2.2 Εξερεύνηση

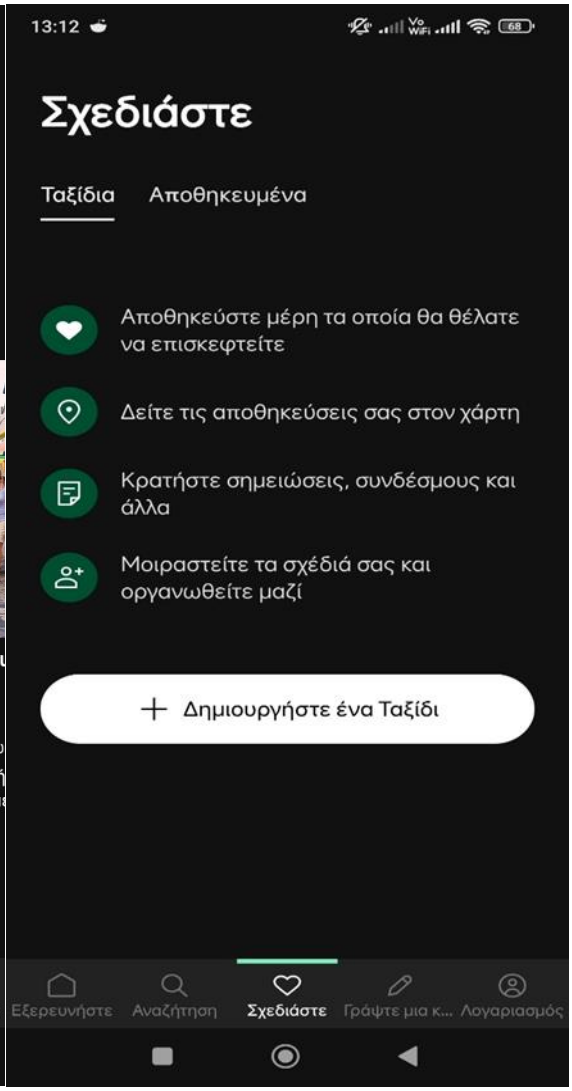
Πηγή: www.tripadvisor.com

Στην εικόνα 2.1 βλέπουμε την αρχική σελίδα του TripAdvisor το οποίο προαπαιτεί την εγγραφή και σύνδεση του χρήστη μέσω Google ή κάποιου email. Στη συνέχεια η εφαρμογή ζητάει από το χρήστη να ενεργοποιήσει τη τοποθεσία του

Αφού ο χρήστης συνδεθεί με το λογαριασμό του μπορεί να εξερευνήσει διάφορες επιλογές (εικονα 2.2) όπως ξενοδοχεία, εστιατόρια και δραστηριότητες η να δει κάποιες προτάσεις που του κάνει η ιδιά η εφαρμογή λαμβάνοντας ως κριτήριο τη τοποθεσία του χρήστη και τι διαθεσιμότητα υπάρχει σε κοντινή απόσταση.



Εικόνα 2.3 Αναζήτηση

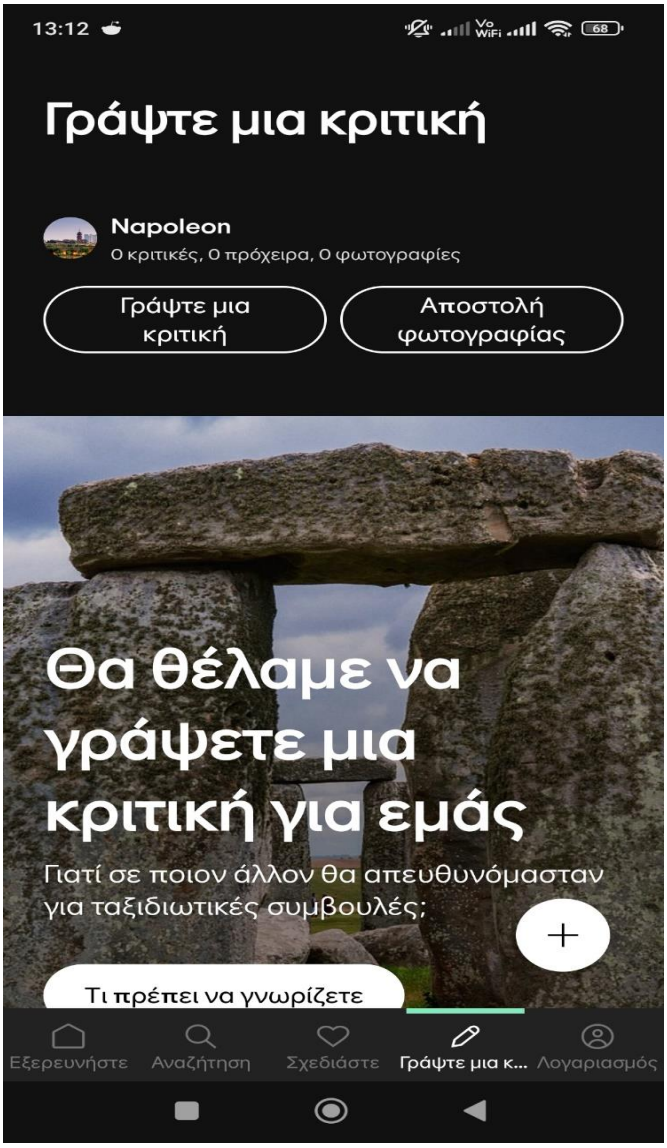


Εικόνα 2.4 Σχεδιασμός ταξιδιού

Πηγή: www.tripadvisor.com

Η εφαρμογή δίνει τη δυνατότητα στο χρήστη να αναζητήσει οτιδήποτε θέλει(εικόνα 2.3) όπως κάποιο πιθανό προορισμό πόλης ή κάποιο event ή οτιδήποτε άλλο θέλει για το ταξίδι του.

Στην εικόνα 2.4 παρουσιάζεται η δυνατότητα που δίνει στον χρήστη το TripAdvisor να δημιουργήσει το δικό του ταξίδι κρατώντας σημειώσεις, αποθηκεύοντας τα σημεία ενδιαφέροντος που θέλει να επισκεφτεί καθώς και να τα μοιραστεί με άλλους χρήστες και να οργανωθούν μαζί.



Εικόνα 2.5 Κριτικές Πηγή: www.tripadvisor.com

Ο χρήστης έχει τη δυνατότητα να αξιολογήσει και να παρέχει πληροφορίες στο Tripadvisor.

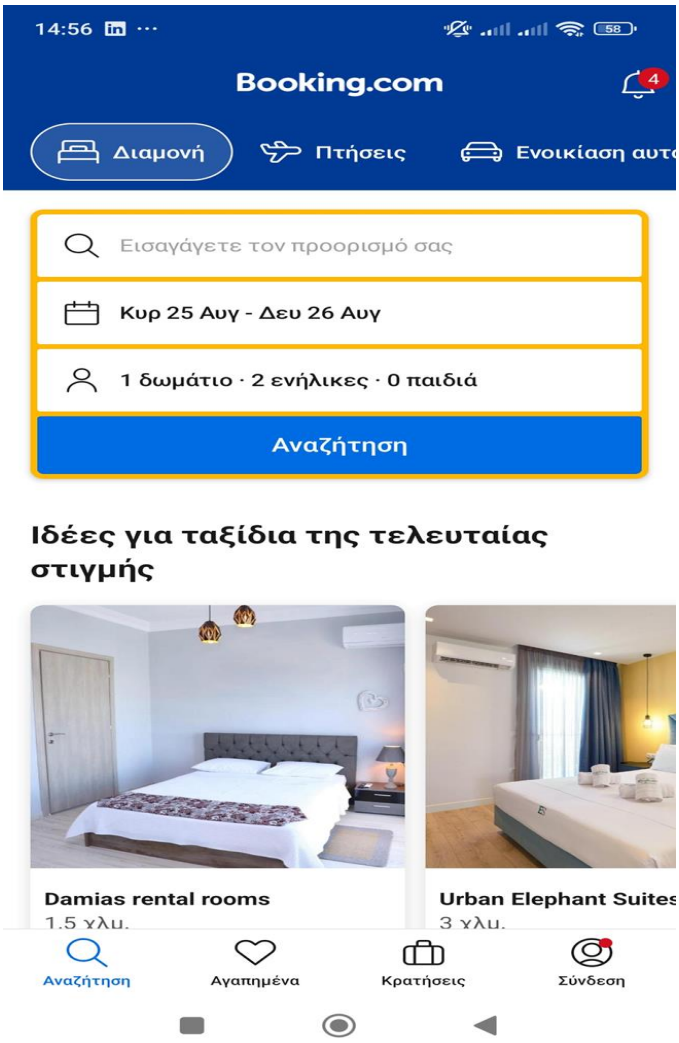
2.2.4 Booking.com

Το Booking.com είναι μια από τις μεγαλύτερες και πιο δημοφιλείς πλατφόρμες OnLine κρατήσεων καταλυμάτων στον κόσμο. Ιδρύθηκε το 1996 και έχει την έδρα της στο Άμστερνταμ της Ολλανδίας. Η πλατφόρμα επιτρέπει στους χρήστες να αναζητούν, να συγκρίνουν και να κάνουν κρατήσεις για ξενοδοχεία, διαμερίσματα, βίλες και άλλες μορφές καταλυμάτων σε παγκόσμια κλίμακα. Το Booking.com προσφέρει επίσης υπηρεσίες κρατήσεων για πτήσεις, ενοικιαζόμενα αυτοκίνητα και δραστηριότητες, καθιστώντας την μια ολοκληρωμένη λύση για τον προγραμματισμό ταξιδιών. Η πλατφόρμα είναι γνωστή για την ευκολία χρήσης της, τις ανταγωνιστικές τιμές και την αξιόπιστη εξυπηρέτηση πελατών [5].

2.2.5 Τεχνολογίες που χρησιμοποιούνται στο Booking.com

Η ανάπτυξη της εφαρμογής Booking.com περιλαμβάνει μια ποικιλία τεχνολογιών και πλατφορμών, οι οποίες συνεργάζονται για να παρέχουν την απαραίτητη λειτουργικότητα, από την υποδομή και την επεξεργασία δεδομένων μέχρι την εμπειρία χρήστη. Ορισμένες από τις κύριες τεχνολογίες που χρησιμοποιούνται είναι οι εξής: JavaScript , Kotlin, Swift ,Objective-C, MySQL, Python, TensorFlow, RESTful APIs.

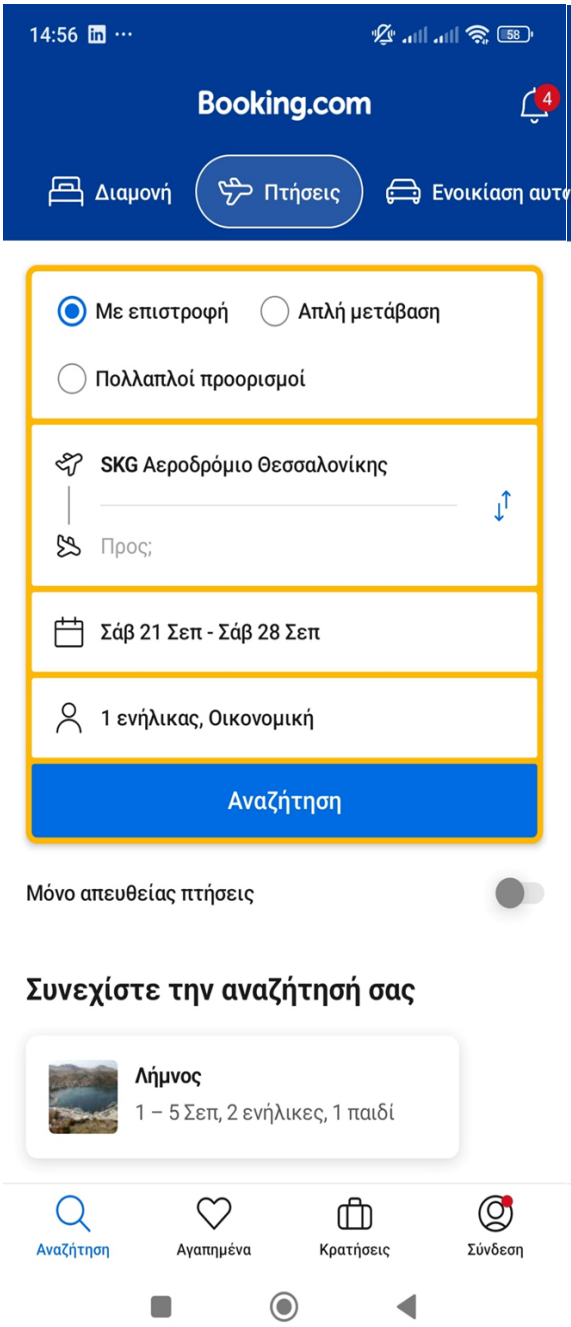
2.2.6 Περιήγηση στην εφαρμογή



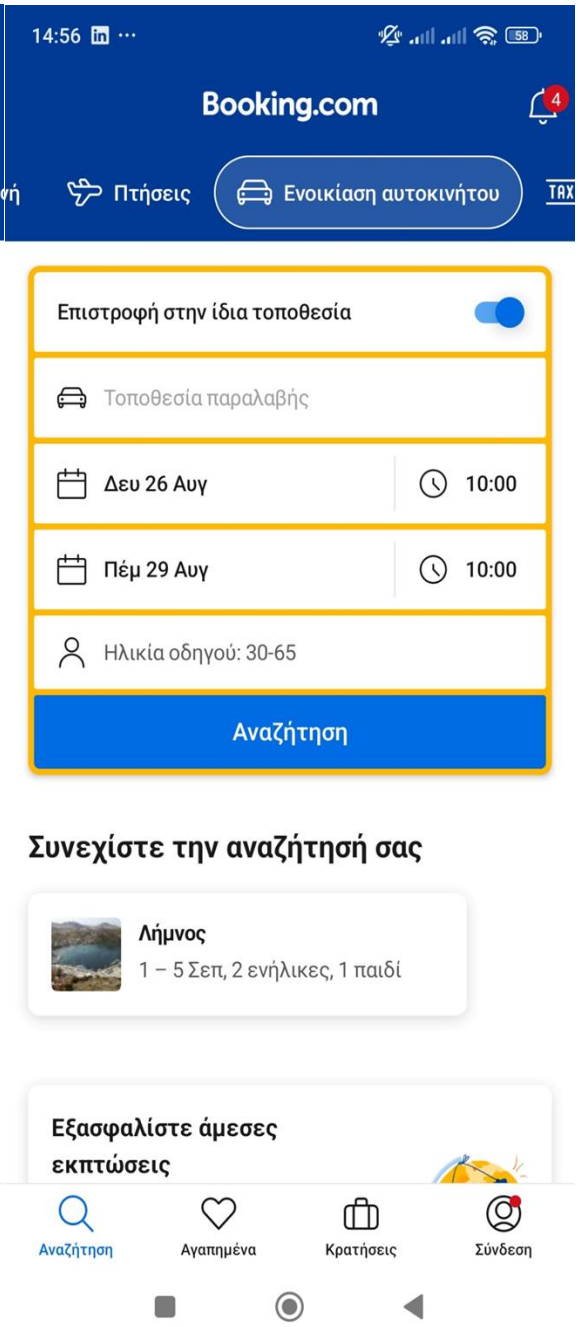
Εικόνα 2.6 Αρχική Booking Πηγή: www.booking.com

Στη φωτογραφία 2.6 βλέπουμε την αρχική σελίδα της εφαρμογής Booking.com. Ο χρήστης έχει τη δυνατότητα να αναζητήσει κάποιο μέρος που θέλει να ταξιδέψει καθώς και να επιλέξει στα φίλτρα αναζήτησης τις ημερομηνίες αλλά και τον αριθμό των ατόμων που θα είναι στο ταξίδι.

Επίσης στο κάτω μέρος υπάρχουν οι επιλογές να δει ποια καταλύματα έχει βάλει στα αγαπημένα όπως επίσης τις κρατήσεις που έχει κάνει. Τέλος δίνεται η δυνατότητα εγγραφής και σύνδεσης του χρήστη με προσωπικό λογαριασμό.



Εικόνα 2.7 Flights

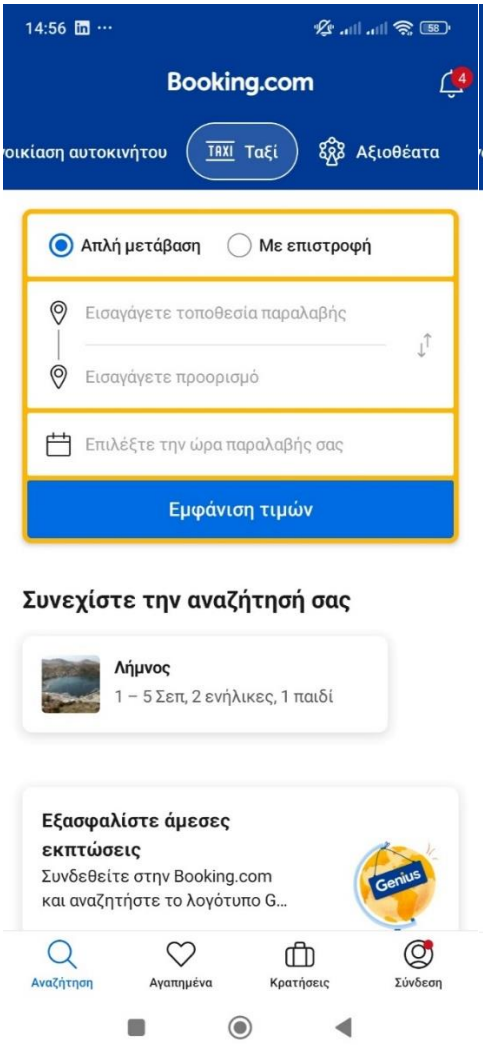


Εικόνα 2.8 Car Rental

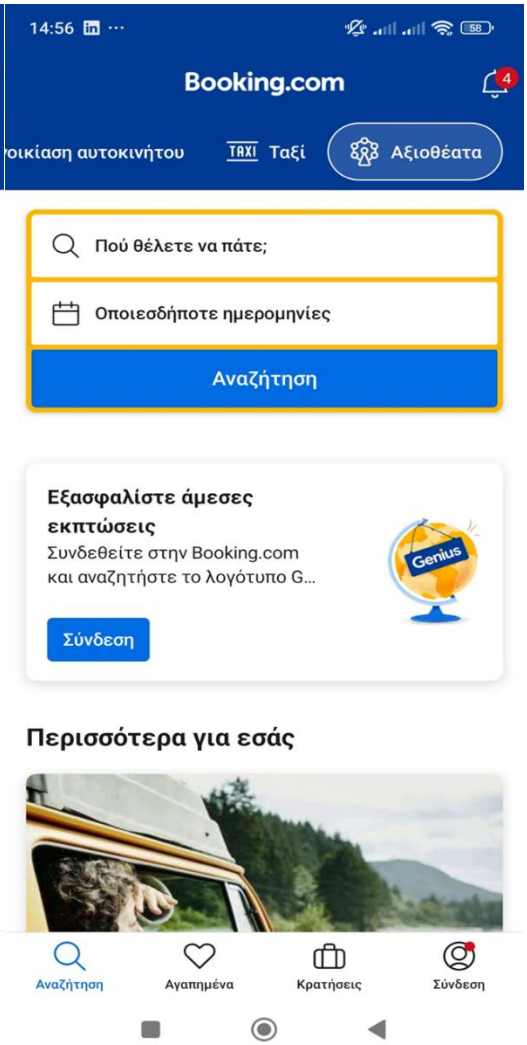
Πηγή: www.booking.com

Πέρα από καταλύματα και προορισμούς ο χρήστης μπορεί να κλείσει και αεροπορικά εισιτήρια για το ταξίδι του χωρίς να χρειαστεί να χρησιμοποιήσει άλλη εφαρμογή. (εικόνα 2.7)

Ο χρήστης έχει τη δυνατότητα να νοικιάσει αυτοκίνητο στο τόπο τον οποίο θα ταξιδέψει (εικόνα 2.8) όπως επίσης αν δεν χρειάζεται αυτοκίνητο να διαλέξει κάποιο ταξί να τον πάει στον προορισμό του (εικόνα 2.9).



Εικόνα 2.9 Taxi



Εικόνα 2.10 Αξιοθέατα

Πηγή: www.booking.com

Τέλος μπορεί να αναζητήσει διάφορα αξιοθέατα με βάση το μέρος που θέλει να επισκεφτεί.

Σε γενικές γραμμές το Booking καλύπτει μεγάλο μέρος των αναγκών ενός ταξιδιώτη.

3. Τεχνολογίες εφαρμογής Kozani Travel

Στο παρακάτω κεφάλαιο αναπτύσσονται οι τεχνολογίες που χρησιμοποιήθηκαν για τη δημιουργία της εφαρμογής μας Kozani Travel.

3.1 Android

Εικόνα 3.1 Android Logo

Πηγή:
https://upload.wikimedia.org/wikipedia/commons/6/64/Android_logo_2019_%28stacked%29.svg

Το Android είναι το λειτουργικό σύστημα που επιλέξαμε για την δημιουργία της εφαρμογής Kozani Travel. Πρόκειται για ένα λειτουργικό σύστημα που αναπτύχθηκε από τη Google με στόχο να χρησιμοποιηθεί σε κινητές συσκευές όπως smartphone, tablet κλπ. Βάση του λειτουργικού αυτού είναι το Linux που είναι ανοιχτού κώδικα και δίνει τη δυνατότητα στον δημιουργό να το προσαρμόσει όπως αυτός θέλει και καθιστά πιο εύκολη τη χρήση του Android σε μεγαλύτερο εύρος συσκευών τα οποία μπορεί να έχουν τελείως διαφορετικά χαρακτηριστικά [2].

Από την άλλη η χρήση του σε τόσο μεγάλο εύρος συσκευών οδηγεί σε μεγάλες καθυστερήσεις σε ενημερώσεις λογισμικού με αποτέλεσμα να δημιουργούνται πολλά κενά ασφαλείας καθώς το καθιστά εύκολο στόχο σε κακόβουλο λογισμικό.

Το Android εν έτη 2024 έχει κατακτήσει μεγάλο μέρος της αγοράς κινητών συσκευών. Εντοπίζεται σε Smartphones,Tablets, SmartTvs, SmartWatches, αυτοκίνητα (Android Auto) και σε πάρα πολλές άλλες συσκευές.

Εναλλακτικές του Android είναι κυρίως το λειτουργικό σύστημα της Apple το iOS το οποίο είναι ένα λειτουργικό κλειστού κώδικα, φτιαγμένο ειδικά για της συσκευές της Apple (Iphone – Ipad). Αυτό το κάνει πιο ελεγχόμενο, πιο ασφαλές αλλά λιγότερο ευέλικτο.

3.2 Android Studio

Εικόνα 3.2 Android studio logo

Πηγή:https://android-developers.googleblog.com/2023/05/android-studio-io-23-announcing-studio-bot.html

Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) της Google για την ανάπτυξη εφαρμογών Android. Βασίζεται στην πλατφόρμα IntelliJ IDEA της JetBrains και προσφέρει ένα πλήρες σετ εργαλείων και δυνατοτήτων για τη δημιουργία εφαρμογών για συσκευές Android, όπως smartphones, tablets, wearables και άλλες συσκευές που χρησιμοποιούν το λειτουργικό σύστημα Android.

Αποτελεί ένα πολύ ισχυρό εργαλείο που προσφέρει πολλές δυνατότητες για την ανάπτυξη εφαρμογών καθιστώντας το πρώτη επιλογή για τους περισσοτέρους προγραμματιστές Android. [3]

3.2.1 Πλεονεκτήματα – Μειονεκτήματα χρήσης του Android Studio

Πλεονεκτήματα:

1) Απλότητα στη χρήση
2) Συνεχής υποστήριξη από τη Google που το καθιστά συμβατό με όλες τις εκδόσεις Android παλιές και καινούριες.
3) Περιλαμβάνει πολλά εργαλεία για τη δημιουργία κώδικα, UI, αυτόματης συμπλήρωσης κώδικα κλπ.
4) Διαθέτει ανάλυση σφαλμάτων και λαθών

5) Διαθέτει ισχυρό emulator ώστε οι προγραμματιστές να δοκιμάζουν τις εφαρμογές σε διαφορετικές εκδόσεις android αλλά και πολλαπλές συσκευές.
6) Περιλαμβάνει ενσωμάτωση με το Git για την εύκολη συνεργασία στις αλλαγές του κώδικα [4].

Μειονεκτήματα:

1) Λόγω της μεγάλης γκάμας εργαλείων και δυνατοτήτων η εκμάθηση του μπορεί να είναι δύσκολη.
2) Απαιτεί ισχυρό υλικό για να τρέξει σωστά και γρήγορα καθώς η χρήση του χρειάζεται πολλούς υπολογιστικούς πόρους. Συστήματα με χαμηλότερη ισχύ ίσως να αντιμετωπίσουν προβλήματα.

3.2.2 Εναλλακτικές στο Android Studio

1) Visual Studio με Xamarin
2) IntelliJ IDEA
3) Flutter
4) Eclipse
5) Kotlin Multiplatform

Το Android Studio χρησιμοποιείται ευρέως για την ανάπτυξη εφαρμογών μέσω Kotlin και Java. Μέσω του εργαλείου αυτού υπάρχει η δυνατότητα και δοκιμής της εφαρμογής μέσω του προσομοιωτή καθώς και η βελτίωση και συντήρηση της στη πάροδο του χρόνου.

Τέλος, το Android Studio είναι το βασικό εργαλείο που χρησιμοποιούν οι καθηγητές στα Πανεπιστήμια πληροφορικής για την εκπαίδευση των φοιτητών πάνω στο Android και την ανάπτυξη εφαρμογών για αυτό.

3.3 Retrofit

Εικόνα 3.3 Retrofit logo Πηγή: https://medium.com/kufar-tech/retrofit-tips-passing-custom-parameters-from-retrofits-request-to-okhttp-s-interceptor-989b8ceff07d

Η Retrofit είναι μια δημοφιλής βιβλιοθήκη για Android, που χρησιμοποιείται για να απλοποιεί την ανάπτυξη εφαρμογών Android επιτρέποντας έτσι την εύκολη και αποτελεσματική εκτέλεση HTTP αιτημάτων. Η βιβλιοθήκη αυτή δημιουργήθηκε από την Square που ανέπτυξε τα Dagger, Okhttp , κλπ., με σκοπό να διευκολύνει την πραγματοποίησή κλήσεων API με τον πιο αποτελεσματικό τρόπο, μετατρέποντας τα JASON ή XML δεδομένα σε αντικείμενα Java[6].

3.3.1 Πλεονεκτήματα Retrofit

Τα πλεονεκτήματα της χρήσης της Retrofit είναι :

1) Εύκολη διαχείριση των δεδομένων, καθώς χρησιμοποιεί το GSON για αυτόματη μετατροπή JSON δεδομένων σε αντικείμενα JAVA.
2) Η Retrofit επιτρέπει στον χρήστη, να διαχειρίζεται αιτήματα API με απλό και κατανοητό τρόπο.
3) Δημιουργεί άμεση επικοινωνία με υπηρεσίες web, και υποστηρίζει τόσο σύγχρονα και ασύγχρονα αιτήματα δικτύου.

3.3.2 Μειονεκτήματα Retrofit

Τα μειονεκτήματα της χρήσης της βιβλιοθήκης Retrofit είναι :

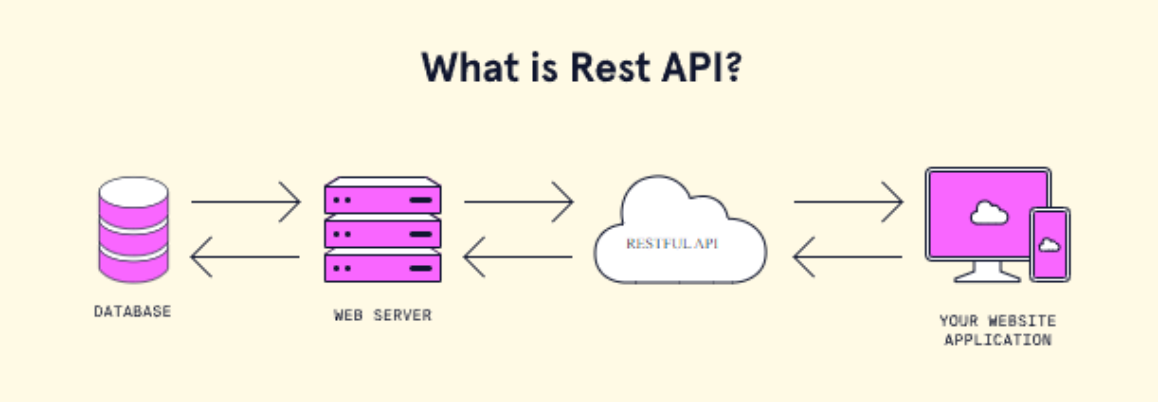
1) Το μειονέκτημα της Retrofit να βασίζεται από άλλες βιβλιοθήκες όπως το OKHttp καθώς και το GSON μπορεί να δημιουργήσει πολλές φορές προβλήματα συμβατότητας.
2) Δεν υποστηρίζει τον καθορισμό προτεραιοτήτων .
3) Έχει περιορισμένη ευελιξία , σε ορισμένες περιπτώσεις.

3.3.3 Εναλλακτικές της Retrofit

1) Volley (βιβλιοθήκη της Google).

2) OKHttp (παρέχει χαμηλότερου επιπέδου διαχείριση αιτημάτων http).

3.4 Rest API



Εικόνα 3.4 REST API logo Πηγή: https://www.codecademy.com/article/what-is-rest

Το REST (Representational State Transfer) API είναι ένα αρχιτεκτονικό στυλ για τη δημιουργία web services. Επιτρέπει την επικοινωνία μεταξύ των συστημάτων μέσω του πρωτοκόλλου HTTP. Χρησιμοποιεί πόρους, που αντιπροσωπεύονται με URLs, και χειρισμούς πόρων μέσω HTTP μεθόδων όπως GET, POST, PUT κλπ.

Το REST API χρησιμοποιείται για την ανάπτυξη web εφαρμογών και mobile εφαρμογών και επικοινωνία μεταξύ διαφορετικών συστημάτων. Είναι ιδιαίτερα χρήσιμο όταν απαιτείται εύκολη και γρήγορη ανάπτυξη και ενσωμάτωση συστημάτων που χρειάζονται να ανταλλάξουν δεδομένα μέσω του διαδικτύου [7] [8].

3.4.1 Πλεονεκτήματα και Μειονεκτήματα του REST API

Πλεονεκτήματα :

1) Είναι πολύ απλό στη χρήση και την υλοποίηση μιας και χρησιμοποιεί HTTP πρωτόκολλα και μεθόδους.
2) Δέχεται πολλούς τύπους δεδομένων όπως HTML, XML , JSON κλπ.
3) Εξυπηρετεί μεγάλες βάσεις χρηστών.

Μειονεκτήματα :

1) Δεν υποστηρίζει τις σύνθετες συναλλαγές.
2) Έχει περιορισμένη ασφάλεια.
3) Απαιτεί πολλούς πόρους όταν υπάρχει μεγάλος όγκος δεδομένων σε πραγματικό χρόνο με αποτέλεσμα να μην είναι τόσο αποδοτικό.

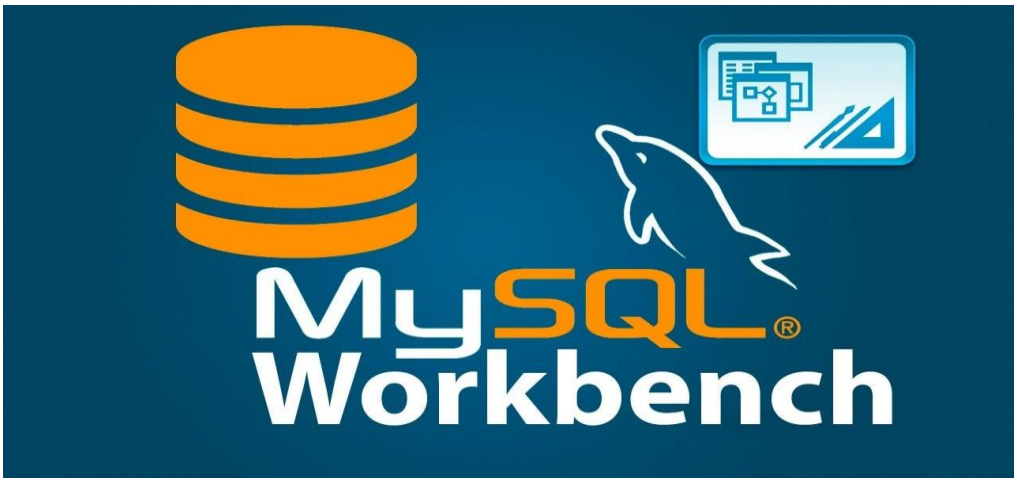
3.4.2 Εναλλακτικές του REST API

1) GraphQL

2) gRPC

3) WebSockets

3.5 MySQL workbench



Εικόνα 3.5 MySQL Workbench logo Πηγή: https://tiruppurwebhosting.com/blog/what-is-mysql-server-workbench/

Το MySQL workbench αναπτύχθηκε από την Oracle και είναι ένα εργαλείο που δίνει τη δυνατότητα σχεδιασμού και διαχείρισης βάσεων δεδομένων για το σύστημα MySQL.

Σχεδιασμός μοντέλων βάσεων δεδομένων, δημιουργία και διαχείριση βάσεων δεδομένων, εκτέλεση ερωτημάτων SQL και διαχείριση χρηστών και δικαιωμάτων είναι οι δυνατότητες που παρέχει το εργαλείο [12] [13].

Το MySQL Workbench χρησιμοποιείται κυρίως σε περιπτώσεις όπου υπάρχει ανάγκη για:

1) Σχεδιασμό και οπτικοποίηση της δομής μιας βάσης δεδομένων.
2) Διαχείριση και συντήρηση βάσεων δεδομένων MySQL.
3) Ασφάλεια και διαχείριση χρηστών και δικαιωμάτων.
4) Δημιουργία και εκτέλεση ερωτημάτων SQL.

3.5.1 Πλεονεκτήματα και μειονεκτήματα

Πλεονεκτήματα :

1) Στη δωρεάν έκδοση του παρέχονται οι περισσότερες βασικές λειτουργίες για τη διαχείριση των βάσεων.
2) Είναι αρκετά εύκολο στη χρήση με απλό γραφικό περιβάλλον.

3) Διαθέτει τη δυνατότητα σχεδιασμού και οπτικοποίησης μοντέλων δεδομένων με την χρήση ER διαγραμμάτων.
4) Επιτρέπει την αυτοματοποίηση επαναλαμβανόμενων εργασιών μέσω σεναρίων και προγραμματισμένων εργασιών.

Μειονεκτήματα :

1) Παρέχει περιορισμένη υποστήριξη για άλλες βάσεις δεδομένων πέρα από τη MySQL.
2) Όταν οι βάσεις δεδομένων είναι πολύ μεγάλες και τα ερωτήματα πολύπλοκα τότε έχει χαμηλή απόδοση.

3.5.2 Εναλλακτικές του MySQL Workbench

1) Navicat
2) HeidiSQL
3) DBeaver
4) phpMyAdmin

3.6 Hibernate

Εικόνα 3.6 Hibernate logo Πηγή: https://medium.com/@saurabh.kundu/hibernate-a-guide-to-essential-annotations-5588d459b6f

Η Hibernate είναι ένα εργαλείο σχεσιακής αντιστοίχισης αντικειμένων ανοιχτού κώδικα (ORM) για την γλώσσα του προγραμματισμού Java. Παρέχει ένα πλαίσιο χαρτογράφησης αντικειμενοστραφών μοντέλων στον τομέα με σχεσιακές βάσεις δεδομένων. Με αυτόν τον τρόπο , διευκολύνει την ανάπτυξη εφαρμογών που απαιτούν αλληλεπίδραση με βάσεις δεδομένων , μειώνοντας την ανάγκη για άμεση γραφή SQL. Η Hibernate χάρη στις δυνατότητες και την ευκολία που διαθέτει , για την διαχείριση της βάσης δεδομένων, αποτελεί μια ισχυρή επιλογή για πολλούς προγραμματιστές [14].

3.6.1 Πλεονεκτήματα και Μειονεκτήματα της Hibernate

Πλεονεκτήματα:

1) Είναι ανεξάρτητη από την βάση δεδομένων και αυτό της επιτρέπει να χρησιμοποιηθεί με πολλές διαφορετικές βάσεις δεδομένων χωρίς αλλαγές στον κώδικα.
2) Περιλαμβάνει μηχανισμούς caching που βοηθούν στην βελτίωση των εφαρμογών.
3) Η αναπτυξιακή ευελιξία που έχει , επιτρέπει την εύκολη αλλαγή και επέκταση του μοντέλου δεδομένων.

Μειονεκτήματα:

1) Η αυτοματοποίηση πολλών διεργασιών μπορεί να έχει ως αποτέλεσμα την δυσκολία ως προς την διάγνωση των προβλημάτων.
2) Η χρήση caching μπορεί να απαιτήσει μεγάλες ποσότητες μνήμης.
3) Σε πολλές περιπτώσεις η Hibernate μπορεί να είναι λιγότερο αποδοτική από ότι η χειροκίνητα γραμμένη SQL , καθώς δεν είναι ευέλικτη σε περιπτώσεις σύνθετης χαρτογράφησης.

3.6.2 Εναλλακτικές της Hibernate

1) JPA (Java Persistence API)

2) MyBatis

3) EclipseLink

3.7 POSTMAN

Εικόνα 3.7 Postman logo Πηγή: https://www.workiy.com/technologies/postman

Το Postman είναι μια εφαρμογή λογισμικού που επιτρέπει στους χρήστες να δοκιμάζουν, να τεκμηριώνουν και να μοιράζονται APIs(Application Programming Interfaces). Οι προγραμματιστές το χρησιμοποιούν ευρέως για να απλοποιήσουν τη διαδικασία δοκιμής αιτήσεων API. Έτσι το Postman παρέχει μια φιλική προς το χρήστη διεπαφή με πολλές δυνατότητες που διευκολύνουν την δημιουργία , την αποστολή καθώς και τη διαχείριση αιτήσεων API [16].

Το Postman, ένα ευρέως χρησιμοποιούμενο εργαλείο δοκιμών και ανάπτυξης API, διαθέτει το δικό του σύνολο πλεονεκτημάτων και μειονεκτημάτων

3.7.1 Πλεονεκτήματα και Μειονεκτήματα του Postman

Πλεονεκτήματα:

1) Το Postman παρέχει μια διαισθητική και φιλική προς το χρήστη διεπαφή, επιτρέποντας στους χρήστες να δημιουργούν, να δοκιμάζουν και να διαχειρίζονται εύκολα αιτήματα API.
2) Υποστηρίζει πολλούς τύπους αιτήσεων όπως : GET ,POST, DELETE καθώς και τύπους αιτήσεων HTTP.
3) Οι χρήστες μπορούν να δημιουργήσουν και να διαχειριστούν πολλαπλά περιβάλλοντα για απρόσκοπτη εναλλαγή μεταξύ διαφορετικών περιβαλλόντων ανάπτυξης, δοκιμών και παραγωγής.

Μειονεκτήματα:

1) Ορισμένα προηγμένα χαρακτηριστικά είναι διαθέσιμα μόνο στην επί πληρωμή έκδοση.
2) Προβλήματα με τη συντήρηση και την ενημέρωση της API.
3) Μπορεί να είναι απαιτητικό ως προς το σύστημα , ιδιαίτερα σε μεγαλύτερες συλλογές.

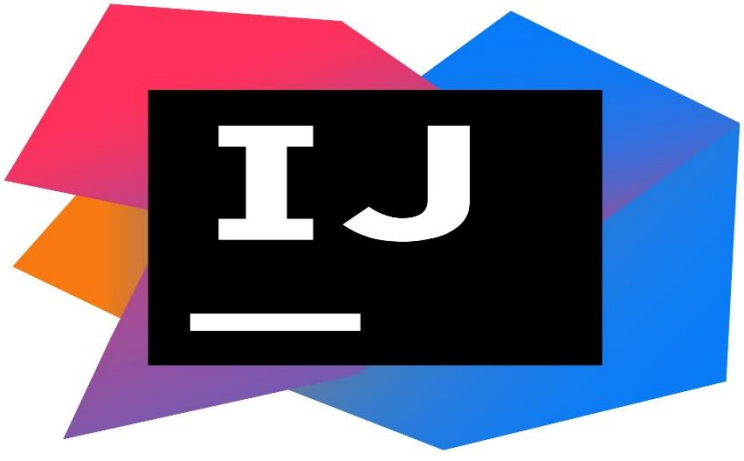
3.7.2 Εναλλακτικές του Postman

1) Insomnia (εργαλείο για την αποστολή αιτήσεων HTTP)

2) Swagger (εργαλείο για τη δημιουργία , δοκιμή και τεκμηρίωση API με βάση το OpenAPI Specification).

3) Paw (Διαθέσιμο για macOs, παρέχει δυνατότητες και γραφική διεπαφή).

3.8 IntellijUltimate



Εικόνα 3.8 IntellijUltimate logo Πηγή: https://www.shi.com/product/35759795/IntelliJ-IDEA-Ultimate-Commercial-Subscription-Licence-1-Yr

Το IntelliJ IDEA Ultimate είναι ένα ισχυρό και ολοκληρωμένο IDE (Integrated Development Environment) που αναπτύσσεται από την JetBrains.Το JetBrains Academy είναι μια διαδικτυακή πλατφόρμα για την εκμάθηση προγραμματισμού, συμπεριλαμβανομένων γλωσσών προγραμματισμού όπως η Python, η Java και η Kotlin. Είναι ευρέως χρησιμοποιούμενο για την ανάπτυξη εφαρμογών σε Java, αλλά υποστηρίζει και πολλές άλλες γλώσσες προγραμματισμού, όπως Kotlin, Groovy, Scala, κτλπ. Προσφέρει μια πληθώρα εργαλείων και χαρακτηριστικών που ενισχύουν την παραγωγικότητα και την ευκολία ανάπτυξης λογισμικού [17] [18].

3.8.1 Πλεονεκτήματα και Μειονεκτήματα του IntellijUltimate

Πλεονεκτήματα:

1) Παρέχει αυτόματη συμπλήρωση κώδικα, στατικού κώδικα καθώς και έξυπνες προτάσεις αναδιάρθρωσης.

2) Παρέχει υποστήριξη με build tools όπως Maven, Gradle καθώς επίσης και συστήματα ελέγχου εκδόσεων όπως Git.

Μειονεκτήματα:

1) είναι εμπορικό λογισμικό με συνδρομητική τιμολόγηση ,κάτι το οποίο μπορεί να είναι αποτρεπτικό για πολλούς.

2) Απαιτεί σημαντικούς πόρους συστήματος.

3.8.3 Εναλλακτικές IntellijUltimate

Το IntelliJ IDEA Ultimate χρησιμοποιείται κυρίως σε επαγγελματικά περιβάλλοντα όπου η παραγωγικότητα, η ποιότητα του κώδικα και η υποστήριξη για πολλά εργαλεία και γλώσσες είναι έντονη. Κάποιες από τις εναλλακτικές θα μπορούσαν να είναι οι εξής :

1) Eclipse (ανοιχτού κώδικα IDE για Java).
2) NetBeans (ανοιχτού κώδικα IDE για Java).

4. Βήματα υλοποίησης της εφαρμογής

4.1 Υλοποίηση εφαρμογής

Activity: SplashActivity

Σκοπός και Λειτουργικότητα

Το SplashActivity αποτελεί την αρχική οθόνη της εφαρμογής. Η οθόνη εμφανίζεται όταν ο χρήστης ξεκινά την εφαρμογή. Παραμένει για 3 δευτερόλεπτα, παρουσιάζοντας το λογότυπο με animation και στη συνέχεια γίνεται μετάβαση στην κύρια οθόνη της εφαρμογής. Η λειτουργικότητά του είναι να δημιουργήσει μια αρχική αίσθηση branding για την εφαρμογή, κάνοντας μια μετάβαση προς την κύρια λειτουργικότητα.

Κύρια Στοιχεία Λειτουργικότητας

1. **Animation Λογότυπου**:
    - Στο onResume() καλείται η μέθοδος animateLogo(), η οποία υλοποιεί ένα animation για το λογότυπο της εφαρμογής.
    - Το συγκεκριμένο animation (AlphaAnimation) είναι ένα fade-in εφέ που διαρκεί 3 δευτερόλεπτα.
    - Όταν ολοκληρωθεί το animation, το activity αυτόματα ξεκινά το MainActivity (το κύριο activity της εφαρμογής) και το SplashActivity κλείνει (finish()).

```
private void animateLogo() {
    final AlphaAnimation fadeIn = new AlphaAnimation(0.0f, 1.0f);
    fadeIn.setDuration(3000);
    fadeIn.setAnimationListener(new Animation.AnimationListener() {
        @Override
        public void onAnimationStart(Animation animation) {

        }

        @Override
        public void onAnimationEnd(Animation animation) {
            startActivity(new Intent( packageContext: SplashActivity.this, MainActivity.class));
            finish();
        }

        @Override
        public void onAnimationRepeat(Animation animation) {

        }
    });
    binding.splashLogo.startAnimation(fadeIn);
}
```

Εικόνα 4.1 On Resume

2. **Διακοπή Animation με Click**:
    - Το SplashActivity επιτρέπει στον χρήστη να σταματήσει το animation νωρίτερα, αν αγγίξει την οθόνη (splashContainer).
    - Αυτό επιτυγχάνεται με την κλήση της μεθόδου clearAnimation() στο splashLogo.

```
binding.splashContainer.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        binding.splashLogo.clearAnimation();
    }
});
```

Εικόνα 4.2 ClickListener

Activity: MainActivity

Σκοπός και Λειτουργικότητα

Το ListItemsActivity εμφανίζει μια λίστα αντικειμένων που σχετίζονται με μια επιλεγμένη κατηγορία. Ο χρήστης μπορεί να φιλτράρει τα αντικείμενα είτε με βάση την απόσταση είτε αλφαβητικά. Επίσης, το activity χειρίζεται υποκατηγορίες της βασικής κατηγορίας και προσφέρει δυνατότητες πλοήγησης και αναζήτησης με φιλτράρισμα.

Σκοπός και Λειτουργικότητα

Το MainActivity είναι η κεντρική οθόνη της εφαρμογής και περιλαμβάνει τη διαχείριση του μενού, τη φόρτωση κατηγοριών και την εύρεση τοποθεσίας του χρήστη.

Κύρια Στοιχεία Λειτουργικότητας

1. **Πλοήγηση μέσω Μενού**:
    - Χρησιμοποιείται το **DrawerLayout** με ένα μενού πλοήγησης (**NavigationView**) που επιτρέπει την πλοήγηση ανάμεσα στις κατηγορίες της εφαρμογής.
    - Το μενού ρυθμίζεται με τη βοήθεια του **AppBarConfiguration**, που καθορίζει τις βασικές επιλογές πλοήγησης.

2. **Φόρτωση και Διαχείριση Κατηγοριών**:

Η μέθοδος **getCategories()** είναι υπεύθυνη για τη φόρτωση των κατηγοριών από το backend, χρησιμοποιώντας τον CategoryManager

```
private void getCategories() {
    if (! new Utils().isNetworkAvailable(context)) {
        Utils.errorDialog(context, "Η σύνδεση απέτυχε. Βεβαιωθείτε πως είστε συνδεδεμένοι σ...");
        return;
    }

    //TODO: add loading dialog
    new CategoryManager().getCoreCategories(new ManagerCompleteListener() {
        @Override
        public void onComplete(Object object, int statusCode) {
        //TODO: dismiss loading dialog
            if ((statusCode == 200) &&  object instanceof List<?>) {
                if (((List<?>) object).size() > 0 && ((List<?>) object).get(0) instanceof Category) {
                    for (Object category : (List<?>) object) {
                        menuCategoriesArrayList.add((Category) category);
                    }
                    Configuration.coreCategoriesArrayList = menuCategoriesArrayList;
                    setUpMenu();
                } //else leave list empty
            } else {
                Utils.errorDialog(context, "Παρακαλώ δοκιμάστε αργότερα");
            }
        }
    });
}
```

Εικόνα 4.3 CategoryManager

Οι κατηγορίες που επιστρέφονται αποθηκεύονται στην menuCategoriesArrayList και στη συνέχεια φορτώνονται στο μενού μέσω της μεθόδου **setUpMenu()**.

```
private void setUpMenu() {
    DrawerLayout drawer = binding.drawerLayout;
    NavigationView navigationView = binding.navigationView;
    navigationView.setNavigationItemSelectedListener(this);
    navigationView.bringToFront();
    // Passing each menu ID as a set of Ids because each
    // menu should be considered as top level destinations.
    mAppBarConfiguration = new AppBarConfiguration.Builder(
            R.id.nav_home)
            .setOpenableLayout(drawer)
            .build();
    NavController navController = Navigation.findNavController( activity: this, R.id.nav_host_fragment_content_main);
    NavigationUI.setupActionBarWithNavController( activity: this, navController, mAppBarConfiguration);
    NavigationUI.setupWithNavController(navigationView, navController);
    reloadCategories();
    navigationView.setNavigationItemSelectedListener(
            new NavigationView.OnNavigationItemSelectedListener() {
                @Override
                public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
                    menuItem.setChecked(true);
                    drawer.closeDrawers();
                    return true;

                }
            });
}
```

Εικόνα 4.4 SetUpMenu

3. **Ανίχνευση Τοποθεσίας Χρήστη**:Χρησιμοποιείται η FusedLocationProviderClient για την εύρεση της τοποθεσίας του χρήστη.

```
private void getLastLocation() {
    // check if permissions are given
    if (checkPermissions()) {

        // check if location is enabled
        if (isLocationEnabled()) {
            // getting last
            // location from
            // FusedLocationClient
            // object
            mFusedLocationClient.getLastLocation().addOnCompleteListener(new OnCompleteListener<Location>() {
                @Override
                public void onComplete(@NonNull Task<Location> task) {
                    Location location = task.getResult();
                    if (location == null) {
                        requestNewLocationData();
                    } else {
                        Configuration.userSavedLang = location.getLatitude();
                        Configuration.userSavedLng = location.getLongitude();
                    }
                }
            });
        } else {
            Toast.makeText( context: this,  text: "Please turn on" + " your location...", Toast.LENGTH_LONG).show();
            Intent intent = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
            startActivity(intent);
        }
    } else {
        // if permissions aren't available,
        // request for permissions
        requestPermissions();
    }
}
```

Εικόνα 4.5 getLocation

Η μέθοδος **getLastLocation()** ελέγχει αν έχει δοθεί η απαιτούμενη άδεια χρήσης τοποθεσίας και αν είναι ενεργοποιημένη η υπηρεσία GPS.

Αν δεν υπάρχει αποθηκευμένη τοποθεσία, ζητείται νέα τοποθεσία μέσω της **requestNewLocationData()**.

```
@SuppressLint("MissingPermission")
private void requestNewLocationData() {

    // Initializing LocationRequest
    // object with appropriate methods
    LocationRequest mLocationRequest = new LocationRequest();
    mLocationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
    mLocationRequest.setInterval(5);
    mLocationRequest.setFastestInterval(0);
    mLocationRequest.setNumUpdates(1);

    // setting LocationRequest
    // on FusedLocationClient
    mFusedLocationClient = LocationServices.getFusedLocationProviderClient( activity: this);
    mFusedLocationClient.requestLocationUpdates(mLocationRequest, mLocationCallback, Looper.myLooper());
}
```

Εικόνα 4.6 NewLocationData

4. **Διαχείριση Δικαιωμάτων Τοποθεσίας**:

Η εφαρμογή ελέγχει και ζητά τα απαραίτητα δικαιώματα για την πρόσβαση στην τοποθεσία του χρήστη μέσω των μεθόδων **checkPermissions()** και **requestPermissions()**.

Αν ο χρήστης δεν έχει ενεργοποιήσει το GPS, εμφανίζεται ειδοποίηση για να ενεργοποιήσει την τοποθεσία του.

```
// method to check for permissions
private boolean checkPermissions() {
    return ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_COARSE_LOCATION) == PackageManager.PERMISSION_GRANTED
            && ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED;
}

// method to request for permissions
private void requestPermissions() {
    ActivityCompat.requestPermissions( activity: this, new String[]{
            Manifest.permission.ACCESS_COARSE_LOCATION,
            Manifest.permission.ACCESS_FINE_LOCATION}, PERMISSION_ID);
}
```

Εικόνα 4.7 RequestPerm

Λειτουργίες

- **Μενού Πλοήγησης**: Παρουσιάζει τις διαθέσιμες κατηγορίες της εφαρμογής. Οι κατηγορίες φορτώνονται δυναμικά και παρουσιάζονται σε ένα μενού πλοήγησης με τη χρήση του RecyclerView.
- **Ανίχνευση Τοποθεσίας**: Καθορίζει την τρέχουσα τοποθεσία του χρήστη και την αποθηκεύει στις μεταβλητές Configuration.userSavedLang και Configuration.userSavedLng.
- **Διαχείριση Δικαιωμάτων**: Ζητά δικαιώματα χρήσης τοποθεσίας αν δεν είναι ήδη διαθέσιμα και ελέγχει αν η τοποθεσία είναι ενεργοποιημένη.
- **Πλοήγηση ανάμεσα σε Fragment**: Το MainActivity χρησιμοποιεί τον NavController για τη διαχείριση της πλοήγησης στα fragments που περιέχει, όπως το HomeFragment.

Activity: ListItemsActivity

Σκοπός και Λειτουργικότητα

Το ListItemsActivity εμφανίζει μια λίστα αντικειμένων που σχετίζονται με μια επιλεγμένη κατηγορία. Ο χρήστης μπορεί να φιλτράρει τα αντικείμενα είτε με βάση την απόσταση είτε αλφαβητικά. Επίσης, το activity χειρίζεται υποκατηγορίες της βασικής κατηγορίας και προσφέρει δυνατότητες πλοήγησης και αναζήτησης με φιλτράρισμα.

Κύρια Στοιχεία Λειτουργικότητας

1. **Πλοήγηση και Ρύθμιση Toolbar**:

Το toolbar εμφανίζει τον τίτλο της επιλεγμένης κατηγορίας και παρέχει δυνατότητα πλοήγησης προς τα πίσω.

```
private void setUpToolbar() {
    Objects.requireNonNull(getSupportActionBar()).setTitle(category.getTitle());
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);
    getSupportActionBar().setDisplayShowHomeEnabled(true);
}
```

Εικόνα 4.8 Toolbar

Η μέθοδος **setUpToolbar()** ρυθμίζει το toolbar για το activity.

2. Διαχείριση Υποκατηγοριών

Η μέθοδος **getSubCategories()** καλεί τον CategoryManager για να φορτώσει τις υποκατηγορίες της τρέχουσας κατηγορίας από το backend.

Αν η κατηγορία έχει δύο υποκατηγορίες, αυτές εμφανίζονται σε tabs, επιτρέποντας στον χρήστη να επιλέξει μία από τις υποκατηγορίες για να εμφανιστούν τα σχετικά αντικείμενα.

```
private void getSubCategories() {
    if (! new Utils().isNetworkAvailable(context)) {
        Utils.errorDialog(context, "Η σύνδεση απέτυχε. Βεβαιωθείτε πως είστε συνδεδεμένοι σ...");
        return;
    }

    //TODO: add loading dialog
    new CategoryManager().getSubcategory(category.getCoreCategoryId(), new ManagerCompleteListener() {
        @Override
        public void onComplete(Object object, int statusCode) {
            //TODO: dismiss loading dialog
            if ((statusCode == 200) &&  object instanceof List<?>) {
                if (((List<?>) object).size() > 0 && ((List<?>) object).get(0) instanceof Category) {
                    for (Object category : (List<?>) object) {
                        subCategories.add((Category) category);
                    }
                    setupView();
                } //else leave list empty
            } else {
                Utils.errorDialog(context, "Παρακαλώ δοκιμάστε αργότερα");
            }
        }
    });
}
```

Εικόνα 4.9 SubCategories

3. **Πλοήγηση με Tabs**:

Χρησιμοποιείται ένα **TabLayout** για να επιτρέψει στον χρήστη να επιλέξει ανάμεσα σε διαφορετικές υποκατηγορίες.
Η μέθοδος **setupView()** δημιουργεί τα tabs και διαχειρίζεται την επιλογή τους, φορτώνοντας τα αντικείμενα της επιλεγμένης υποκατηγορίας.

```
private void setupView() {
    getListOfItems( tabNumber: 0);
    if (subCategories.size() != 2) {
        binding.appBarLayout.setVisibility(View.GONE);
        return;
    }
    binding.tabs.addTab(binding.tabs.newTab().setText(subCategories.get(0).getTitle()));
    binding.tabs.addTab(binding.tabs.newTab().setText(subCategories.get(1).getTitle()));
    binding.tabs.addOnTabSelectedListener(new TabLayout.OnTabSelectedListener() {
        @Override
        public void onTabSelected(TabLayout.Tab tab) { getListOfItems(tab.getPosition()); }

        @Override
        public void onTabUnselected(TabLayout.Tab tab) {
        }

        @Override
        public void onTabReselected(TabLayout.Tab tab) {
        }
    });
}
```

Εικόνα 4.10 Tabs

4. **Φόρτωση και Εμφάνιση Αντικειμένων**:

Η μέθοδος **getListOfItems(int tabNumber)** φορτώνει τα αντικείμενα της επιλεγμένης υποκατηγορίας μέσω του ItemManager.
Τα αντικείμενα αποθηκεύονται στην itemsArrayList και στη συνέχεια εμφανίζονται στη λίστα.

```
private void getListOfItems(int tabNumber) {

    itemsArrayList.clear();
    if (! new Utils().isNetworkAvailable(context)) {
        Utils.errorDialog(context, "Η σύνδεση απέτυχε. Βεβαιωθείτε πως είστε συνδεδεμένοι σ...");
        return;
    }

    //TODO: add loading dialog

    new ItemManager().getItems(subCategories.get(tabNumber).getCategoryId(), new ManagerCompleteListener() {
        @Override
        public void onComplete(Object object, int statusCode) {
            //TODO: dismiss loading dialog
            if ((statusCode == 200) &&  object instanceof List<?>) {
                if (((List<?>) object).size() > 0 && ((List<?>) object).get(0) instanceof Item) {
                    for (Object item : (List<?>) object) {
                        itemsArrayList.add((Item) item);
                    }
                }
            } else {
                Utils.errorDialog(context, "Παρακαλώ δοκιμάστε αργότερα");
            }
            setupRecyclerView();
        }
    });
}
```

Εικόνα 4.11 tabNumber

5. Ταξινόμηση Αντικειμένων

Η εφαρμογή επιτρέπει φιλτράρισμα και ταξινόμηση των αντικειμένων είτε με βάση την απόσταση είτε αλφαβητικά.

Η μέθοδος sortList(int by) ταξινομεί τα αντικείμενα με βάση την επιλεγμένη παράμετρο.

```
private void sortList(int by) {
    Collections.sort(itemsArrayList, new Comparator<Item>(){
        public int compare(Item item1, Item item2) {
            if (by == 0) {
                return Double.compare(item1.getDistance(), item2.getDistance());
            } else {
                return item1.getTitle().compareToIgnoreCase(item2.getTitle());
            }
        }
    });

}
```

Εικόνα 4.12 Sortlist

6. **Αναζήτηση και Φιλτράρισμα Αντικειμένων**

Η μέθοδος **openFiltersPopUp()** ανοίγει ένα dialog για τη ρύθμιση των φίλτρων με τη βοήθεια του FiltersAlertDialog.

```
private void openFiltersPopUp() {
    new FiltersAlertDialog(context, selectedFilter).show(getSupportFragmentManager(), tag: null);
}
```

Εικόνα 4.13 filter

7. **Αναπαράσταση Αντικειμένων**:

Η εμφάνιση των αντικειμένων γίνεται μέσω ενός RecyclerView.

Η μέθοδος **setupRecyclerView()** είναι υπεύθυνη για την προετοιμασία του RecyclerView και την ανανέωση των δεδομένων ανάλογα με τα επιλεγμένα φίλτρα.

```
private void setupRecyclerView() {
    if (itemsArrayList.size() > 0) {
        binding.noLabel.setVisibility(View.GONE);
        binding.recyclerView.setVisibility(View.VISIBLE);
        reload( selectedFilter: 0);
    } else {
        binding.noLabel.setVisibility(View.VISIBLE);
        binding.recyclerView.setVisibility(View.GONE);
    }
}

public void reload(int selectedFilter) {
    sortList(selectedFilter);
    this.selectedFilter = selectedFilter;
    binding.recyclerView.setLayoutManager(new GridLayoutManager( context: this,  spanCount: 2));
    ItemsAdapter adapter = new ItemsAdapter(itemsArrayList, context);
    binding.recyclerView.setAdapter(adapter);
}
```

Εικόνα 4.14 Recycler

Λειτουργίες

- **Πλοήγηση μεταξύ Υποκατηγοριών**: Ο χρήστης μπορεί να επιλέξει διαφορετικές υποκατηγορίες μέσω των tabs, και η λίστα των αντικειμένων θα ανανεώνεται ανάλογα με την επιλεγμένη υποκατηγορία.
- **Εμφάνιση Λίστας Αντικειμένων**: Παρουσιάζει μια λίστα από αντικείμενα της τρέχουσας κατηγορίας, χρησιμοποιώντας το RecyclerView.
- **Φιλτράρισμα Αντικειμένων**: Επιτρέπει στον χρήστη να φιλτράρει τη λίστα αντικειμένων με βάση την απόσταση ή αλφαβητικά.
- **Ενημέρωση Toolbar**: Εμφανίζει τον τίτλο της κατηγορίας και επιτρέπει την επιστροφή στην προηγούμενη οθόνη.

Activity: ItemActivity

Σκοπός και Λειτουργικότητα

Το ItemActivity παρουσιάζει λεπτομέρειες για ένα επιλεγμένο στοιχείο και παρέχει λειτουργίες πλοήγησης, κλήσης, προβολής εικόνων και πρόσβασης στον ιστότοπο του στοιχείου. Οι βασικές πληροφορίες φορτώνονται από τον server μέσω του ItemManager.

Σκοπός και Λειτουργικότητα

1. **Ανάκτηση Πληροφοριών του Αντικειμένου**:

Κατά την εκκίνηση του activity, η μέθοδος **getItemDetails()** καλεί το ItemManager για να ανακτήσει τις λεπτομέρειες του επιλεγμένου αντικειμένου.

Εάν το αντικείμενο ανακτηθεί επιτυχώς, φορτώνονται επίσης οι εικόνες του

```
private void getItemDetails() {
    if (! new Utils().isNetworkAvailable(context)) {
        Utils.errorDialog(context, "Η σύνδεση απέτυχε. Βεβαιωθείτε πως είστε συνδεδεμένοι σ...")
        return;
    }

    //TODO: add loading dialog
    new ItemManager().getDetailedItem(itemId, new ManagerCompleteListener() {
        @Override
        public void onComplete(Object object, int statusCode) {
        //TODO: dismiss loading dialog
            if ((statusCode == 200) &&  object instanceof Item) {
                item = ((Item) object);
                getMedia(item.getItemId());
                setUpView();
            } else {
                Utils.errorDialog(context, "Παρακαλώ δοκιμάστε αργότερα");
            }
        }
    });
}
```

αντικειμένου μέσω της μεθόδου **getMedia(int itemId).**

```
private void getMedia(int itemId) {
    if (! new Utils().isNetworkAvailable(context)) {
        Utils.errorDialog(context, "Η σύνδεση απέτυχε. Βεβαιωθείτε πως είστε συνδεδεμένοι σ...");
        return;
    }

    //TODO: add loading dialog
    new ItemManager().getMedia(itemId, new ManagerCompleteListener() {
        @Override
        public void onComplete(Object object, int statusCode) {
        //TODO: dismiss loading dialog
            if ((statusCode == 200) &&  object instanceof List<?>) {
                if (((List<?>) object).size() > 0 && ((List<?>) object).get(0) instanceof Media) {
                    for (Object item : (List<?>) object) {
                        images.add((Media) item);
                    }
                }
            } else {
                Utils.errorDialog(context, "Παρακαλώ δοκιμάστε αργότερα");
            }
            setUpImages();
        }
    });
}
```

2. **Εμφάνιση Πληροφοριών του Αντικειμένου**:

Η μέθοδος **setUpView()** ρυθμίζει τα διάφορα στοιχεία UI που εμφανίζουν τις λεπτομέρειες του αντικειμένου, όπως ο τίτλος, η περιγραφή, η διεύθυνση, το τηλέφωνο, και οι ώρες λειτουργίας.
Αν οι ώρες λειτουργίας είναι κενές, η διάταξη που τις περιέχει κρύβεται.

```
private void setUpView() {
    setUpToolbar();
    binding.titleTV.setText(item.getTitle());
    binding.contentTV.setText(item.getDescription());
    binding.addressTV.setText(item.getAddress());
    binding.phoneTV.setText(item.getPhone());
    binding.workingHoursTV.setText(Html.fromHtml(item.getWorkingHours()));
    if (item.getWorkingHours().isEmpty()) {
        binding.workingHoursLayout.setVisibility(View.GONE);
    }
    setListeners();
}
```

3. **Προβολή Εικόνων**:

Οι εικόνες του αντικειμένου φορτώνονται από τον server και προβάλλονται σε ένα ImageSlider.

Η μέθοδος setUpImages() διαχειρίζεται τη δημιουργία μιας λίστας από εικόνες (SlideModel) και τις προβάλλει στο slider.

```
private void setUpImages() {
    ArrayList<SlideModel> slideModels = new ArrayList<~>() ; // Create image list
    for (Media image : images) {
        slideModels.add(new SlideModel(image.getImage(), title: "", ScaleTypes.CENTER_CROP));
    }
    if (slideModels.size() > 0)
        binding.slider.setImageList(slideModels, ScaleTypes.CENTER_CROP);
}
```

Εικόνα 4.15 SetImages

4. **Ενέργειες Χρήστη**:

Το activity παρέχει διάφορες ενέργειες στον χρήστη μέσω των ακόλουθων:

1) **Κλήση Τηλεφώνου**: Πατώντας το εικονίδιο του τηλεφώνου, το ItemActivity επιχειρεί να καλέσει το τηλέφωνο του αντικειμένου. Ελέγχεται η άδεια κλήσης και αν δεν έχει δοθεί, ζητείται από τον χρήστη.

```
binding.callIV.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String phone = item.getPhone();
        Intent intent = new Intent(Intent.ACTION_CALL);
        intent.setData(Uri.parse("tel:" + phone));
        if (ContextCompat.checkSelfPermission(context,
                Manifest.permission.CALL_PHONE)
                != PackageManager.PERMISSION_GRANTED) {

            ActivityCompat.requestPermissions( activity: ItemActivity.this,
                    new String[]{Manifest.permission.CALL_PHONE},
                    MY_PERMISSIONS_REQUEST_CALL_PHONE);
        } else {
            try {
                startActivity(intent);
            } catch(SecurityException e) {
                e.printStackTrace();
            }
        }
    }
});
```

Εικόνα 4.16 Call

2) Πλοήγηση με Χάρτη: Πατώντας το εικονίδιο του χάρτη, το ItemActivity εκκινεί την εφαρμογή Google Maps για πλοήγηση στη θέση του.

```
binding.mapIV.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        double latitude = item.getLat();
        double longitude = item.getLon();

        Uri gmmIntentUri = Uri.parse("google.navigation:q="+latitude+","+longitude);
        Intent mapIntent = new Intent(Intent.ACTION_VIEW, gmmIntentUri);
        mapIntent.setPackage("com.google.android.apps.maps");
        startActivity(mapIntent);
    }
});
```

Εικόνα 4.17 Map

3) **Πρόσβαση στον Ιστότοπο**: Πατώντας το εικονίδιο ιστότοπου, ανοίγει ο προεπιλεγμένος περιηγητής στη σελίδα του αντικειμένου.

```
binding.webIV.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String webSite = item.getWebsite();
        if (!webSite.startsWith("http://") && !webSite.startsWith("https://")) {
            webSite = "http://" + webSite;
        }
        Intent browserIntent = new Intent(Intent.ACTION_VIEW, Uri.parse(webSite));
        startActivity(browserIntent);
    }
});
```

Εικόνα 4.18 Web

5. **Άδειες Πρόσβασης**:

Το activity ζητά άδεια για να πραγματοποιήσει τηλεφωνικές κλήσεις, χρησιμοποιώντας το **Manifest.permission.CALL_PHONE**.
Η άδεια ελέγχεται κατά την προσπάθεια κλήσης και αν δεν έχει δοθεί, ζητείται από τον χρήστη μέσω του **ActivityCompat.requestPermissions**.

6. **Πλοήγηση**:

Το toolbar του activity εμφανίζει τον τίτλο του αντικειμένου και παρέχει την επιλογή επιστροφής στην προηγούμενη οθόνη.

Λειτουργίες

- **Φόρτωση και Εμφάνιση Στοιχείων**: Εμφανίζει λεπτομερείς πληροφορίες για το επιλεγμένο στοιχείο (όπως περιγραφή, διεύθυνση, τηλέφωνο, ώρες λειτουργίας).
- **Πλοήγηση Χρήστη**: Επιτρέπει στον χρήστη να εκκινήσει κλήσεις, να πλοηγηθεί στη θέση του στοιχείου ή να επισκεφτεί τον ιστότοπό του.
- **Προβολή Εικόνων**: Φορτώνει και προβάλει τις εικόνες του στοιχείου σε slider.
- **Διαχείριση Αδειών**: Ζητά από τον χρήστη την άδεια για να εκτελέσει τηλεφωνικές κλήσεις.

Fragment: HomeFragment

Σκοπός και Λειτουργικότητα

Ο κύριος σκοπός του HomeFragment είναι η προβολή των κύριων κατηγοριών της εφαρμογής σε μια λίστα, ώστε ο χρήστης να μπορεί να επιλέξει και να πλοηγηθεί σε επιμέρους στοιχεία της κάθε κατηγορίας.

Κύρια Στοιχεία Λειτουργικότητας

1. **Ανάκτηση Κατηγοριών**:

Η μέθοδος **getCategories()** καλεί το CategoryManager για να ανακτήσει τις κύριες κατηγορίες από τον server.

```
public void getCategories() {
    if (! new Utils().isNetworkAvailable(context)) {
        Utils.errorDialog(context, "Η σύνδεση απέτυχε. Βεβαιωθείτε πως είστε συνδεδεμένοι σ...");
        return;
    }

    //TODO: add loading dialog

    new CategoryManager().getCoreCategories(new ManagerCompleteListener() {
        @Override
        public void onComplete(Object object, int statusCode) {
    //TODO: dismiss loading dialog

            if ((statusCode == 200) &&  object instanceof List<?>) {
                if (((List<?>) object).size() > 0 && ((List<?>) object).get(0) instanceof Category) {
                    for (Object category : (List<?>) object) {
                        menuCategoriesArrayList.add((Category) category);
                    }
                    setUpCategoriesRecycler();
                } //else leave list empty
            } else {
                Utils.errorDialog(context, "Παρακαλώ δοκιμάστε αργότερα");
            }
        }
    });
}
```

Εικόνα 4.19 GetCategories

Αν η ανάκτηση είναι επιτυχής, οι κατηγορίες αποθηκεύονται στην λίστα **menuCategoriesArrayList** και στη συνέχεια η μέθοδος **setUpCategoriesRecycler()** ρυθμίζει την προβολή τους.

```
private void setUpCategoriesRecycler() {
    if(menuCategoriesArrayList.size() > 0 ){
        binding.recyclerView.setVisibility(View.VISIBLE);
        reloadCategories();
    }else{
        binding.recyclerView.setVisibility(View.GONE);
    }
}
```

Εικόνα 4.20 CatRecycler

2. **Ρύθμιση του RecyclerView**:

Το RecyclerView χρησιμοποιείται για την εμφάνιση της λίστας των κατηγοριών.
Αν η λίστα των κατηγοριών δεν είναι κενή, η μέθοδος **setUpCategoriesRecycler()** εμφανίζει το RecyclerView, αλλιώς το κρύβει.

Η μέθοδος **reloadCategories()** ρυθμίζει το RecyclerView με ένα κάθετο LinearLayoutManager και έναν CategoriesAdapter που είναι υπεύθυνος για την παρουσίαση των κατηγοριών.

```
private void reloadCategories(){
    LinearLayoutManager llm = new LinearLayoutManager(requireContext(), LinearLayoutManager.VERTICAL, reverseLayout: false);
    binding.recyclerView.setLayoutManager(llm);
    binding.recyclerView.setHasFixedSize(true);

    CategoriesAdapter adapter = new CategoriesAdapter(menuCategoriesArrayList, context);
    binding.recyclerView.setAdapter(adapter);
}
```

Εικόνα 4.21 RecyclerView

3. **Διαχείριση Δικτύου**:

Το Fragment ελέγχει αν υπάρχει διαθέσιμη σύνδεση στο διαδίκτυο μέσω της μεθόδου **isNetworkAvailable()** από την κλάση Utils.

Σε περίπτωση που δεν υπάρχει σύνδεση, εμφανίζεται ένα διάλογος σφάλματος στον χρήστη με μήνυμα σφάλματος που παρέχεται από τους πόρους (strings.xml).

```
public void getCategories() {
    if (! new Utils().isNetworkAvailable(context)) {
        Utils.errorDialog(context, getResources().getString(R.string.noInternet));
        return;
    }
```

Εικόνα 4.22 ConnError

4. **Ενέργειες Κατά Την Εμφάνιση και Εξαφάνιση**:

Η μέθοδος **onCreateView()** είναι υπεύθυνη για τη δημιουργία της εμφάνισης του Fragment και την εκκίνηση της ανάκτησης των κατηγοριών.

Όταν το Fragment καταστρέφεται, η μέθοδος **onDestroyView()** ρυθμίζει την τιμή του binding σε null για την αποδέσμευση των πόρων.

```
public void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); }

public View onCreateView(@NonNull LayoutInflater inflater,
                         ViewGroup container, Bundle savedInstanceState) {

    binding = FragmentHomeBinding.inflate(inflater, container,  attachToParent: false);
    View root = binding.getRoot();
    context = getContext();

    getCategories();

    return root;
}

@Override
public void onDestroyView() {
    super.onDestroyView();
    binding = null;
}
```

Εικόνα 4.23 OnCreate

Λειτουργίες

- **Ανάκτηση και Εμφάνιση Κατηγοριών**: Ανακτά και εμφανίζει τις κύριες κατηγορίες από τον server.
- **Έλεγχος Σύνδεσης**: Ελέγχει τη διαθεσιμότητα του δικτύου πριν από κάθε ανάκτηση δεδομένων.
- **Διαχείριση RecyclerView**: Εμφανίζει ή αποκρύπτει τη λίστα κατηγοριών, ανάλογα με την επιτυχία της ανάκτησης των δεδομένων.

Activity: ItemActivity

Σκοπός και Λειτουργικότητα

Το ItemActivity παρουσιάζει λεπτομέρειες για ένα επιλεγμένο στοιχείο και παρέχει λειτουργίες πλοήγησης, κλήσης, προβολής εικόνων και πρόσβασης στον ιστότοπο του στοιχείου. Οι βασικές πληροφορίες φορτώνονται από τον server μέσω του ItemManager.

Κύρια Στοιχεία Λειτουργικότητας

1. **Ανάκτηση Πληροφοριών του Αντικειμένου**:

Κατά την εκκίνηση του activity, η μέθοδος **getItemDetails()** καλεί το ItemManager για να ανακτήσει τις λεπτομέρειες του επιλεγμένου αντικειμένου.
Εάν το αντικείμενο ανακτηθεί επιτυχώς, φορτώνονται επίσης τα μέσα (εικόνες) του αντικειμένου μέσω της μεθόδου **getMedia(int itemId)**.

2. **Εμφάνιση Πληροφοριών του Αντικειμένου**:

Η μέθοδος **setUpView()** ρυθμίζει τα διάφορα στοιχεία UI που εμφανίζουν τις λεπτομέρειες του αντικειμένου, όπως ο τίτλος, η περιγραφή, η διεύθυνση, το τηλέφωνο, και οι ώρες λειτουργίας.
Αν οι ώρες λειτουργίας είναι κενές, η διάταξη που τις περιέχει κρύβεται.

3. **Προβολή Εικόνων**:

Οι εικόνες του αντικειμένου φορτώνονται από τον server και προβάλλονται σε ένα ImageSlider.

Η μέθοδος **setUpImages()** διαχειρίζεται τη δημιουργία μιας λίστας από εικόνες (SlideModel) και τις προβάλλει στο slider.

4. **Ενέργειες Χρήστη**:

Το activity παρέχει διάφορες ενέργειες στον χρήστη μέσω των ακόλουθων:
Α) Κλήση Τηλεφώνου: Πατώντας το εικονίδιο του τηλεφώνου, το ItemActivity επιχειρεί να καλέσει το τηλέφωνο του αντικειμένου. Ελέγχεται η άδεια κλήσης και αν δεν έχει δοθεί, ζητείται από τον χρήστη.
Β) Πλοήγηση με Χάρτη: Πατώντας το εικονίδιο του χάρτη, το ItemActivity εκκινεί την εφαρμογή Google Maps για πλοήγηση στη θέση του αντικειμένου.
Γ) Πρόσβαση στον Ιστότοπο: Πατώντας το εικονίδιο ιστότοπου, ανοίγει ο προεπιλεγμένος περιηγητής στη σελίδα του αντικειμένου.

5. **Άδειες Πρόσβασης**:

Το activity ζητά άδεια για να πραγματοποιήσει τηλεφωνικές κλήσεις, χρησιμοποιώντας το **Manifest.permission.CALL_PHONE**.
Η άδεια ελέγχεται κατά την προσπάθεια κλήσης και αν δεν έχει δοθεί, ζητείται από τον χρήστη μέσω του **ActivityCompat.requestPermissions**.

6. **Πλοήγηση**:

Το toolbar του activity εμφανίζει τον τίτλο του αντικειμένου και παρέχει την επιλογή επιστροφής στην προηγούμενη οθόνη.
Λειτουργίες

- **Φόρτωση και Εμφάνιση Στοιχείων**: Εμφανίζει λεπτομερείς πληροφορίες για το επιλεγμένο στοιχείο (όπως περιγραφή, διεύθυνση, τηλέφωνο, ώρες λειτουργίας).
- **Πλοήγηση Χρήστη**: Επιτρέπει στον χρήστη να εκκινήσει κλήσεις, να πλοηγηθεί στη θέση του στοιχείου ή να επισκεφτεί τον ιστότοπό του.
- **Προβολή Εικόνων**: Φορτώνει και προβάλει τις εικόνες του στοιχείου σε slider.
- **Διαχείριση Αδειών**: Ζητά από τον χρήστη την άδεια για να εκτελέσει τηλεφωνικές κλήσεις.

4.2 Βάση δεδομένων

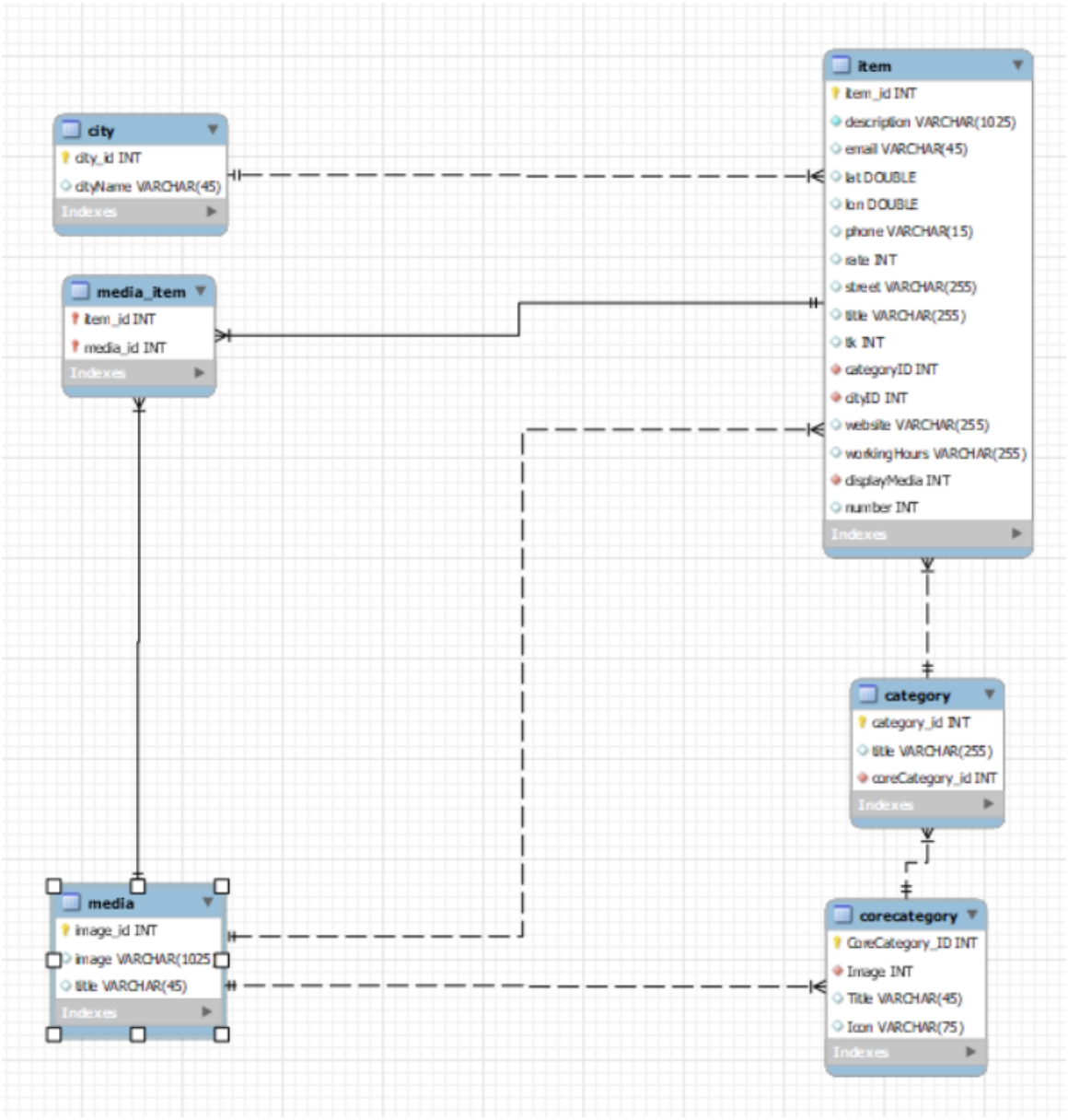
Για τα πλαίσια της ανάπτυξης εφαρμογής υλοποιήθηκε μια σχεσιακή βάση δεδομένων σε MySQL 8 . Το σχήμα της βάσης δεδομένων αποτελείται απο 6 πίνακες με σκοπό την βέλτιστη αρχειοθέτηση απο θέμα χώρου των δεδομένων και ελαχιστοποίησης των κλήσεων που

απαιτούνται για την ενμέρωση των καρτελών των επιχηρήσεων που περιλαμβάνονται στο Kozani Travel. Αναλυτικά:

- Πίνακας City:
    - Primary Key: city_ID (unique)
    - Περιγραφή: περιλαμβάνει όλους τους οικισμού που συμπεριλαμβάνονται στο Κοζάνη Travel.
- Πίνακας Core Category:
    - Primary Key: CoreCategory_ID (unique)
    - Foreign Key: Image
        - Κάθε CoreCategory συσχέετε με μερικές εικόνες, όποτε επιλέχθηκε μια many-to-one συσχέτιση μεταξύ των εικόνων και του core_category.
    - Περιγραφή: περιλαμβάνει τις κύριες κατηγορίες των επιχηρήσεων έτσι όπως είναι οργανωμένες στο Kozani Travel frontend. Συγκεκριμένα περιλαμβάνει τις κατηγορίες
        - Διαμονή
        - Εστίαση
        - Διασκέδαση
        - Αξιοθέατα
        - Δραστηριότητες
- Πίνακας Category:
    - Primary Key: category_ID (unique)
    - Foreign Key: coreCategory_id
        - Κάθε Category ανοίκει σε ένα CoreCategory, όποτε επιλέχθηκε μια one-to-many συσχέτηση μεταξύ του CoreCategory και του Category.
    - Περιγραφή: περιλαμβάνει τις υποκατηγρίες κάθε core category. Αναλυτικα:
        - Διαμονή/Accomodation (CoreCategory – Διαμονή)
        - ΚΑΦΕ (CoreCategory – Διασκέδαση)
        - ΜΠΑΡ (CoreCategory – Διασκέδαση)
        - Restaurants (CoreCategory – Εστίαση)
        - Sights (CoreCategory –Αξιοθέατα)
        - Activities (CoreCategory – Δραστηριότητες)
- Πίνακας: Media
    - Primary Key: image_ID (unique)
    - Περιγραφή: Περιλαμβάνει τις εικόνες και τα εικονίδια που χρησιμοποιούνται απο την εφαρμογή. Τα media αποθηκεύονται ως 1) relative path του filesystem στον server, 2) λεζάντες για το κάθε αντικέιμενο.
- Πίνακας media-item
    - Primary Key: Composite(item_id, media_id)
    - Περιγραφή: Βοηθητικός πίνακας για να υλοποιηθεί η many-to-many συσχέτιση των εικονών της εφαρμογής με τις καρτέλες επιχειρήσεων. Αυτή

η συσχέτιση επιλέχθηκε για να επιτραπεί η επαναχρησιμοποιήση εικονών απο τις επιχειρήσεις.

- Πίνακας Item
    - Primary Key: item_id
    - Foreign Key: displayMedia, categoryID, cityID
    - Περιγραφή: Κύρια καρτέλα όπου περιλαμβάνονται όλα τα στοιχεία τις επιχείρησης, και συσχετίζει τα στοιχεία της επιχείρησης με τους υπόλοιπους πίνακες. Αναλυτικά:
        - Διεύθυνση, η οποία ορίζεται απο τους πίνακες lat, lon, street, title, TK, number, cityID.
        - Rate, η βαθμολογία της συγκεκριμένς επιχείρησης.
        - Description, η περιγραφή που αναγράφετε στην καρτέλα της επιχείρησης.
        - Website, η ιστοσελίδα της επιχείρησης
        - Title, η ονομασία της επιχείρησης
        - Email, η ταχυδρομική διεύθυνση της επιχείρησης
        - Display Media, οι εικόνες που διαφημίζουν την επιχείρηση
        - Working Hours, οι ώρες λειτουργίας σε HTML μορφή
        - Phone, το τηλεφωνο της επιχείρισης.



Εικόνα 4.24 Database Schema

4.3 Υλοποίηση υπηρεσιών ιστού

Σε αυτό το κεφάλαιο θα αναλυθεί η αρχιτεκτονική και το επιμέρους στοιχεία του web service που αποτελεί το backend της εφαρμογής Kozani Travel. Το web service είναι υπεύθυνο για την πρόσβαση στα δεδομένα της βάση δεδομένων της εφαρμογής μέσω απο μια σειρά από Rest Endpoints. Στο κεφάλαιο 4.3.1 θα αναλυθεί η αρχιτεκτονική της εφαρμογής, ενώ στο κεφάλαι 4.3.2 θα παρουσιαστούν τα endpoints τα οποία χρησιμοποιούνται απο το frontend της εφαρμογής Kozani Travel.

4.3.1 Αρχιτεκτονική

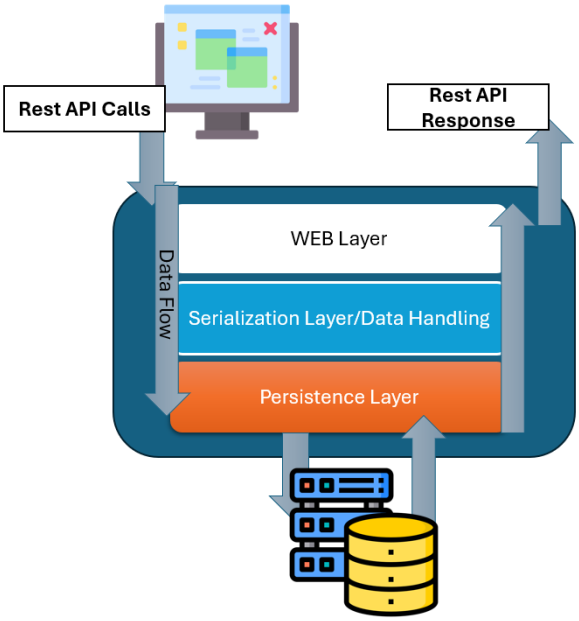
Το backend της εφαρμογής αποτελείτε απο ένα Jersey-based web service, το οποιο ακολουθεί μια αρχιτεκτονική επιπέδων (layer based), όπου κάθε επίπεδο επικοινωνεί μονάχα με τα συγκοινωνούντα επίπεδα. Τα πλεονεκτήματα μιας τέτοιας αρχιτεκτονικής είναι:

- Separation of Concerns. Κάθε επίπεδο έχει συγκεκριμένα καθήκοντα στα πλαίσια του λογικού διαχωρισμού του προγράμματος. Ένας τέτοιος διαχωρισμός ενισχύει την ικανότητα συντήρησης του κώδικα, αφού οι αλλαγές σε ένα επίπεδο δεν επηρεάζουν τα υπόλοιπα.
- Reusability. Ο διαχωρισμος των καθηκόντων κάθε τμήματος κώδικα επιτρέπει την επαναχρησιμοποίηση τους σε πολλά διαφορετικά μέρη του προγράμματος.
- Testability. Κάθε επίπεδο μπορεί να ελεγχθεί επιμέρους για την ορθότητα του. Για παράδειγμα το Persistence Layer μπορεί να ελεγχθεί για την ορθή επικοινωνία με τη βάση δεδομένων χωρίς να χρειάζεται αν ελεγχθεί το format που θα χρησιμοποιηθεί στο HTTP Response.
- Scalability. Η απομονωμένη λογική των επιμέρους κομματιών του κώδικα επιτρέπει την ανάπτυξη ή προσθήκη νέων μεθόδων δίχως να επηρεάζονται τα υπόλοιπα τμήματα του κώδικα. Για παράδειγμα η προσθήκη ενός νέου Rest Endpoint που θα επιστρέφει τις πόλεις ανάλογα τη χιλιομετρική απόσταση, δεν θα απαιτούσε την αλλαγή σε οποιοδήποτε σημείο του Persistence ή Serialization επιπέδου
- Maintainability.
- Flexibility. Η δυνατότητα αντικατάστασης επιμέρους κομματιών κώδικα δίχως να χρειαστούν δραστικές αλλαγές στον κώδικα, πχ η αλλαγή σε άλλη βάση δεδομένων.

Το Web layer διαχειρίζεται τα HTTP requests/responses της εφαρμογής παρέχοντας 7 Restfull endpoints. Όλα τα HTTP αιτήματα δρομολογούνται στις αντίστοιχες μεθόδου του serialization layer ή απευθείας στου persistence layer αναλόγως αν το αίτημα συνοδεύετε απο παραμέτρους.

Το Serialization Layer παρέχει μεθόδους για τη μετρατροπή των παραμέτρων των HTTP Responses & των αποτελεσμάτων των κλήσεων αυτών στις υποστηριζόμενες μορφές της υπηρεσίας.

Το Persistence Layer είναι υπευθυνο για την επικοινωνία με τη βάση δεδομένων μέσω του Hibernate ORM, το οποίο παρέχει μια αντικειμενοστρεφή εικόνα της βάσης δεδομένων σε μορφή κλάσεων Java.



Εικόνα 4.25 Rest Api

4.3.2 Υλοποίηση

Η υλοποίηση του backend βασίστηκε σε μια σειρά τεχνολογιων που ανοίκουν στο Java Enterprise οικοσυστήμα. Αναλυτικά:

- Jersey. Αυτό το framework παρέχει το Rest Api στην εφαρμογή καθώς και τον HTTP listening server της εφαρμογής, επιτρέποντας την δημιουργία Java RESTful web services. Το Jersey υλοποιεί το REST API στο servlet org.glassfish.jersey.servlet.ServletContainer, ενώ η παραμετροποίηση του γίνεται μέσω ενος config αρχείου σε μορφή XML, του web.xml

```
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
  <servlet>
    <servlet-name>Jersey Web Application</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
    <init-param>
      <param-name>jersey.config.server.provider.packages</param-name>
      <param-value>np.ptx.travelapp</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>Jersey Web Application</servlet-name>
    <url-pattern>/webapi/*</url-pattern>
```

```
    </servlet-mapping>
</web-app>
```

- Jackson & GSon. Αυτές οι βιβλιοθήκες χρησιμοποιούνται για την μετροπή των δεδομένων απο και σε JSON format. Η βιβλιοθήκη χρησιμοποιείτε στο Serialization Layer, και συγκεκριμένα απο όλα τα java classes τα οποία χρησιμοποιούν την ονομασια ***(Collection)? $WrappedClassName Serializer.java*** τα οποία υλοποιούν το ***interface JsonSerializer<T>*** που παρέχεται απο την βιβλιοθήκη.

- Hibernate & Java Persistence API (JPA). Το Hibnerate framework, όπως παρουσιάστηκε στο Κεφάλαιο 3.6 χρησιμοποιείτε απο την εφαρμογή για την μορφοποιήση και τον συχρονισμό των tables της database ως java classes. Το JPA αποτελεί το specification με το οποίο υλοποιείτε το Hibernate framework. Η εφαρμογή παραμετροποιεί το hibernate παρέχοντας ενα xml αρχείο, το hibernate.cfg.xml που παραθέτετε στο . Στο αρχείο οριζεται η διασύνδεση με τη βάση δεδομένων, παρέχοντας τα στοιχεία σύνδεσης της βάσης (username & password), καθώς και το database connection driver που χρησιμοποείτε. Το database connection driver αποτελεί το σημείο του κώδικα όπου γίνεται η επικοινωνία μεταξύ της εφαρμογής και του database management system, υλοποιώντας το specification της συγκεκριμένης βάσης δεδομένων.

```
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

  <session-factory>

    <!-- JDBC Database connection settings -->
    <property name="connection.driver_class">com.mysql.cj.jdbc.Driver</property>
    <property name="connection.url">jdbc:mysql://127.0.0.1:3306/travelapp</property>
    <property name="connection.username">root</property>
    <property name="connection.password">root</property>

    <!-- Select our SQL dialect -->
    <property name="dialect">org.hibernate.dialect.MySQL5Dialect</property>

    <!-- Echo the SQL to stdout -->
    <property name="show_sql">true</property>

    <!-- Set the current session context -->
    <property name="current_session_context_class">thread</property>

    <!-- Drop and re-create the database schema on startup -->
    <property name="hibernate.hbm2ddl.auto">validate</property>
```

```
<!-- dbcp connection pool configuration -->
<!-- Assuming you have chosen to use a dedicated connection pooling library, these should be replaced
     or configured according to the chosen library's documentation -->

 </session-factory>

</hibernate-configuration>
```

- MySQL. Η βάση δεδομένων που χρησιμοποείτε απο την εφαρμογή παρουσιάστηκε στο κεφάλαιο 4.2 .Η επικοινωνία με τη βάση γίνεται μέσω του Hibernate framework.
- Maven. Το build και dependency management system που χρησιμοποιείτε απο το backend της εφαρμογής Kozani Travel. Το maven παρέχοντας το pom.xml configuration file, κατεβάζει απο τα official mirrors όλες τις προαπαιτούμενες βιβλιοθήκες του προγράμματος και αναλαμβάνει τη μετατροπή του πηγαίου κώδικα σε εκτελέσιμο (build process).
- Tomcat. Αποτελεί τον server και servlet container που χρησιμοποιείτε για να γίνει deploy η εφαρμογή. Η διασύνδεση με τον tomcat server, γίνεται με΄σω του Maven tomcat plugin.

Η εφαρμογή προσφέρει 7 Restful endpoints τα οποια επληρώνουν τις ανάγκες του frontend, τα endpoints της εφαρμογής παρατίθενται και σε μορφή openAPI 1.0 στο Appendix. Βασικός σκοπός της υλοποίησης ήταν η ελλάτωση των κλήσεων που χρειάζονται ανάλογα με τις ανάγκες του frontend.

- GET: /getCategories
  - Παράμετροι
    - None
  - Return Value: coreCategoryId: ID της κύριας κατηγορίας στην οποία ανοίκει
    - title: Όνομα κατηγορίας
    - categoryId: ID κατγορίας
  - Περιγραφή: Αυτή η κλήση επιστρέφει όλες τις κατηγορίες σε μορφή Json. Το http response αποτελείτε απο ένα Json Array απο Json objects που το καθένα περιέχει το id και το όνομα της κατηγορίας (JsonAttributes: categoryId, title). Αυτή η κλήση χρησιμοποιείτε στην έναρξη της εφαρμογής έτσι ώστε να το frontend να συχρονιστεί το frontend με το backend.
- GET: /getCoreCategories
  - Παράμετροι
    - None
  - Return Value: Json Array of Json Objects
    - coreCategoryId: ID της κύριας κατηγορίας στην οποία ανοίκει
    - title: Όνομα κατγορίας
    - icon: URL εικονιδίου

        - image: URL εικόνας
    - Περιγραφή: Αυτή η κλήση επιστρέφει όλες τις κύριες κατηγορίες σε μορφή Json. Το http response αποτελείτε απο ένα Json Array απο Json objects που το καθένα περιέχει το id, το όνομα, το εικονίδιο και την εικόνα της κύριας. Αυτή η κλήση χρησιμοποιείτε στην έναρξη της εφαρμογής έτσι ώστε να το frontend να συχρονιστεί το frontend με το backend.
- GET: /getCategoriesByCoreId
    - Παράμετροι
        - Int coreId (default: -1)
    - Return Value: Json Array of Json Objects
        - title: Όνομα κατηγορίας
        - categoryId: ID κατγορίας
    - Περιγραφή: Αυτή η κλήση επιστρέφει όλες τις κατηγορίες που αντιστοιχούν στο συγκεκριμένο coreCategoryID. Το HTTP response αποτελείτε απο ένα Json Array απο Json objects που το καθένα περιέχει το id των κατηγοριών.
- GET: /getCities
    - Παράμετροι
        - None
    - Return Value: Json
        - city_id: ID πόλης
        - city_name: όνομα πόλης
    - Περιγραφή: Αυτή η κλήση επιστρέφει όλες τις πόλεις που υποστηρίζει η εφαρμογή. Το HTTP response αποτελείτε απο ένα Json Array απο Json objects που το καθένα περιέχει το όνομα και το id των πόλεων.
- GET: /getMedia
    - Παράμετροι
        - Integer itemId (default: -1)
    - Return Value: Json Array of Json Objects
        - image: URL εικόνας
        - title: Λεζάντα εικόνας
    - Περιγραφή: Αυτή η κλήση επιστρέφει όλες τις εικόνες που αντιστοιχούν στη συγκεκριμένη επιχείρηση με id itemId.
- GET: /getBusiness
    - Παράμετροι
        - Integer id (default: -1)
    - Return Value: Json Object
        - itemId: ID της επιχείρησης
        - title: Ονομασία επιχείρησης
        - Description: Περιγραφή επιχείρησης
        - workingHours: Ώρες λειτουργίας επιχείρησης
        - displayMedia: Κύρια εικόνα επιχείρησης
        - city: ID της πόλης της επιχείρησης
        - website: Website επιχείρησης
        - phone: Τηλέφωνο επιχείρησης
        - rate: Rating επιχείρησης

        - email: ηλεκτρονική ταχυδρομική διεύθυνση επιχείρησης
        - tk: Ταχυδρομικός κώδικας επιχείρησης
        - lat: latitude τοποθεσίας επιχείρησης
        - lon: longitude τοποθεσίας επιχείρησης
    -
- GET: /getBusinessesByCategory
    - Παράμετροι
        - integer category (default: 1)
    - Return Value: Json Array of Json Objects
        - itemId: ID της επιχείρησης
        - title: Ονομασία επιχείρησης
        - Description: Περιγραφή επιχείρησης
        - workingHours: Ώρες λειτουργίας επιχείρησης
        - displayMedia: Κύρια εικόνα επιχείρησης
        - city: ID της πόλης της επιχείρησης
        - website: Website επιχείρησης
        - phone: Τηλέφωνο επιχείρησης
        - rate: Rating επιχείρησης
        - email: ηλεκτρονική ταχυδρομική διεύθυνση επιχείρησης
        - tk: Ταχυδρομικός κώδικας επιχείρησης
        - lat: latitude τοποθεσίας επιχείρησης
        - lon: longitude τοποθεσίας επιχείρησης
    - Περιγραφή: Αυτή η κλήση επιστρέφει όλες τις επιχειρήσεις της κατηγορίας που ζητήθηκε. Ο τύπος των δεδομένων της κατηγοριας είναι integer και αντιστοιχεί στο index της κατηγορίας που είναι κοινή ανάμεσα σε frontend & backend. Η προεπιλεγμένη επιλογή της παραμέτρους category αντιστοιχεί στη πρώτη κατηγορια (index 1). Το http response αποτελείτε απο ένα Json Array απο Json objects που το καθένα περιέχει τα στοιχεία των επιχειρήσεων που ανοίκουν στη ζητούμενη κατηγορία.
    -
- Περιγραφή: Αυτή η κλήση επιστρέφει όλα τα στοιχεία για την συγκεκριμένη επιχείρηση, υπάρχει διαχωρισμός μεταξύ αυτής της κλήσης και της κλήσης για τα media, έτσι ώστε να είναι πιο responsive to frontend, αφού μπορεί να παρέχει έτσι με μικρότερη κλήση τα στοιχεία της επιχείρησης και να προσφέρει τα media σε δεύτερο χρόνο. Το HTTP response αποτελείτε απο ένα JsonObject που περιέχει όλα τα στοιχεία της επιχείρησης.

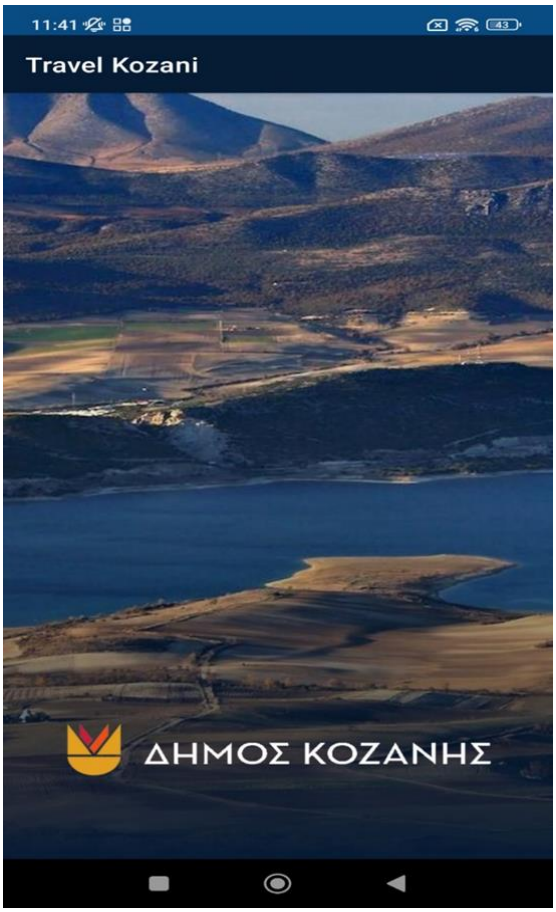
5. Kozani Travel

5.1 Εγχειρίδιο χρήσης εφαρμογής (user manual)

Η Εφαρμογή αναπτύχθηκε με στόχο την καλύτερη εξυπηρέτηση των επισκεπτών της πόλης της Κοζάνης. Προσφέρει εύκολη και γρήγορη πρόσβαση στις πληροφορίες που χρειάζεται ο χρήστης

5.1.1 Είσοδος στην εφαρμογή

Κατά την είσοδο του χρήστη στην εφαρμογή εμφανίζεται η παρακάτω εικόνα.

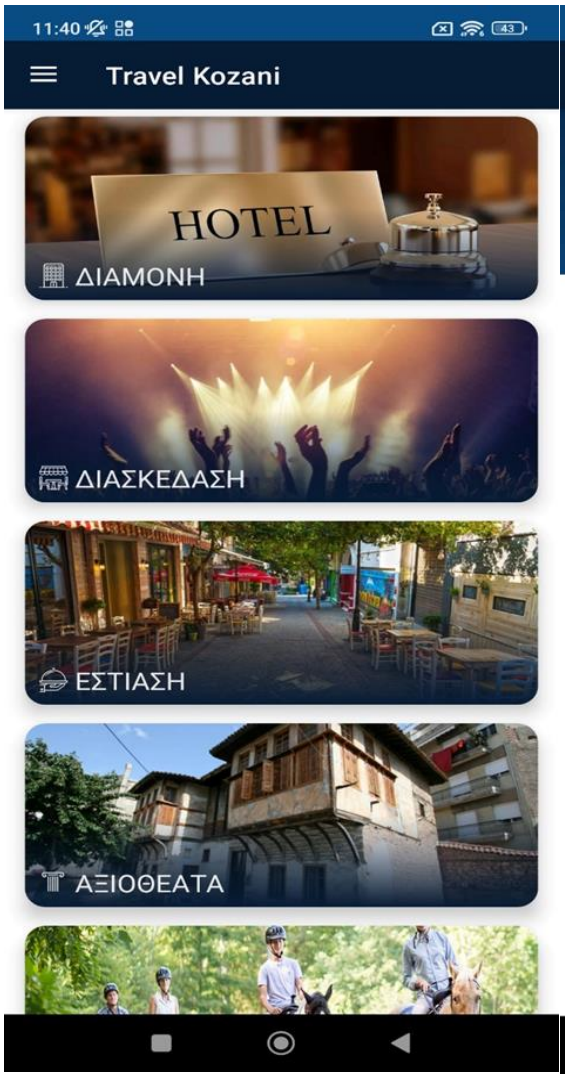


Εικόνα 5.1 Αρχική οθόνη εφαρμογής

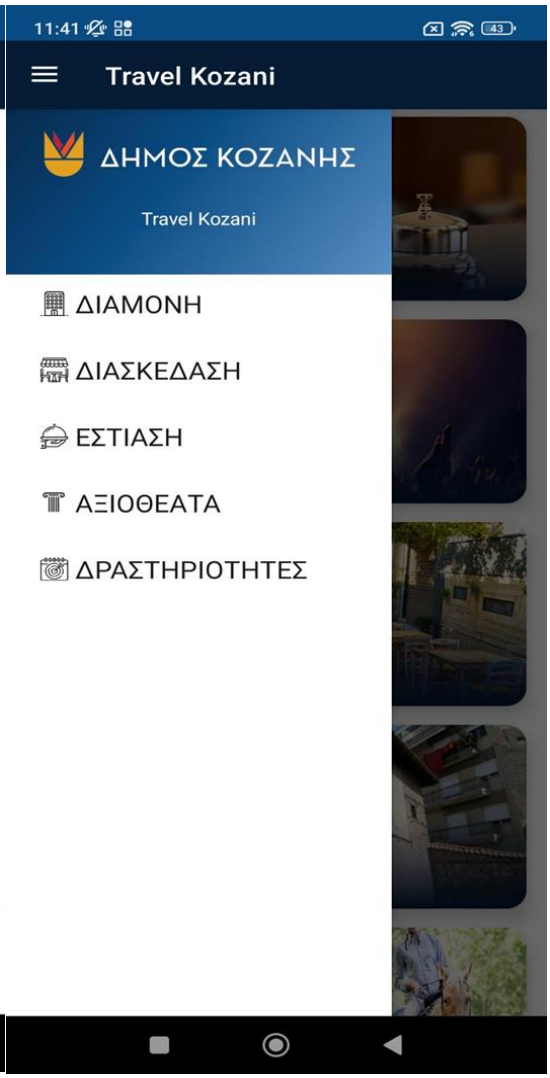
Μετά την εμφάνιση της αρχικής οθόνης η εφαρμογή κάνει δυο ελέγχους ώστε να μπορεί να λειτουργήσει. Η λειτουργία της απαιτεί την ύπαρξη σύνδεσης στο διαδίκτυο και ενεργοποίησης της τοποθεσίας του χρήστη. Σε περίπτωση που τουλάχιστον το ένα

από τα δυο προ απαιτούμενα δεν είναι ενεργοποιημένο η εφαρμογή εμφανίζει αντίστοιχο μήνυμα και ζητάει από τον χρήστη να τα ενεργοποιήσει.

5.2 Περιήγηση στην εφαρμογή



Εικόνα 5.2 Αρχικό μενού



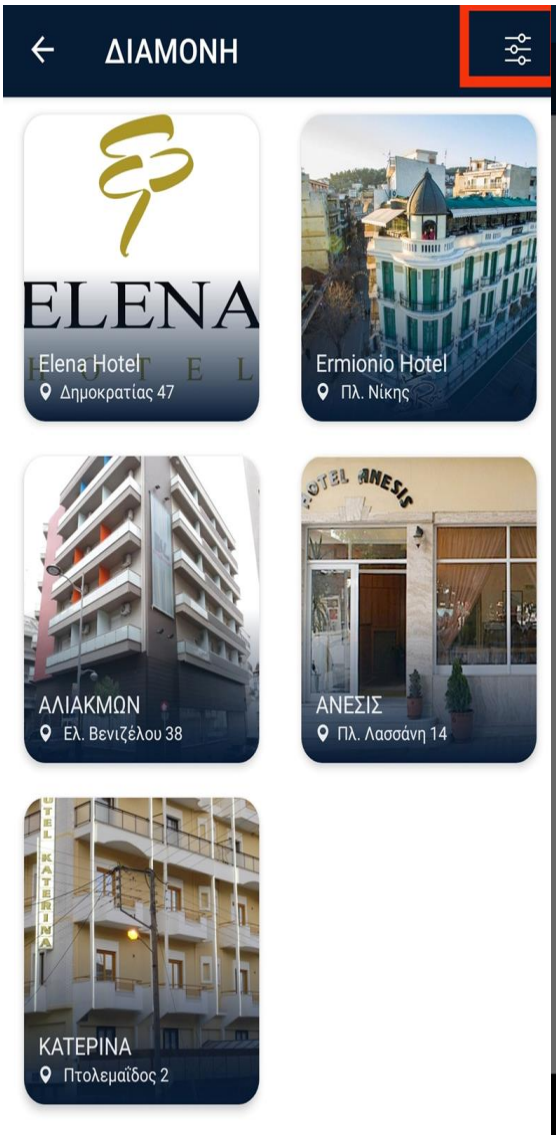
Εικόνα 5.3 πλαϊνό μενού

Στην παραπάνω εικόνα(5.2) εμφανίζεται το αρχικό μενού. Ο χρήστης έχει τη δυνατότητα να επιλέξει την κατηγορία ενδιαφέροντος του. Οι κατηγορίες είναι 5.

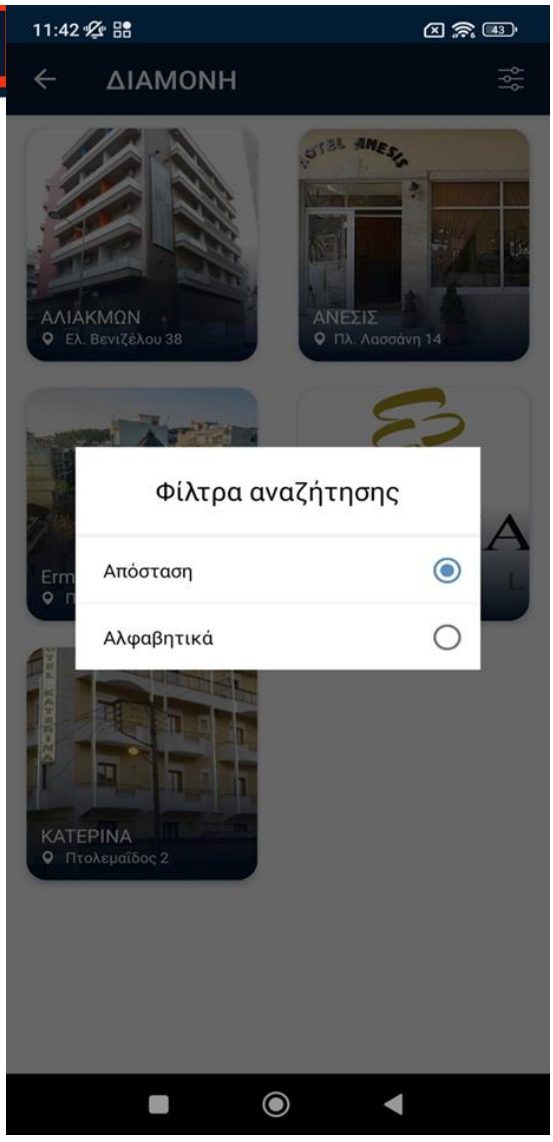
- Διαμονή σε περίπτωση που ψάχνει κάποιο ξενοδοχείο για διανυκτέρευση.
- Διασκέδαση αν θέλει να βρει κάποιο μπαρ ή καφέ.
- Εστίαση αν επιθυμεί να βρει κάποιο εστιατόριο για φαγητό.
- Αξιοθέατα
- Δραστηριότητες.

Παράλληλα πάνω αριστερά υπάρχει ένα κουμπί το οποίο εμφανίζει σαν πλαϊνό μενού τις κατηγορίες (εικόνα 5.3).

5.2.1 Διαμονή και φίλτρο αναζήτησης



Εικόνα 5.4 Διαμονή

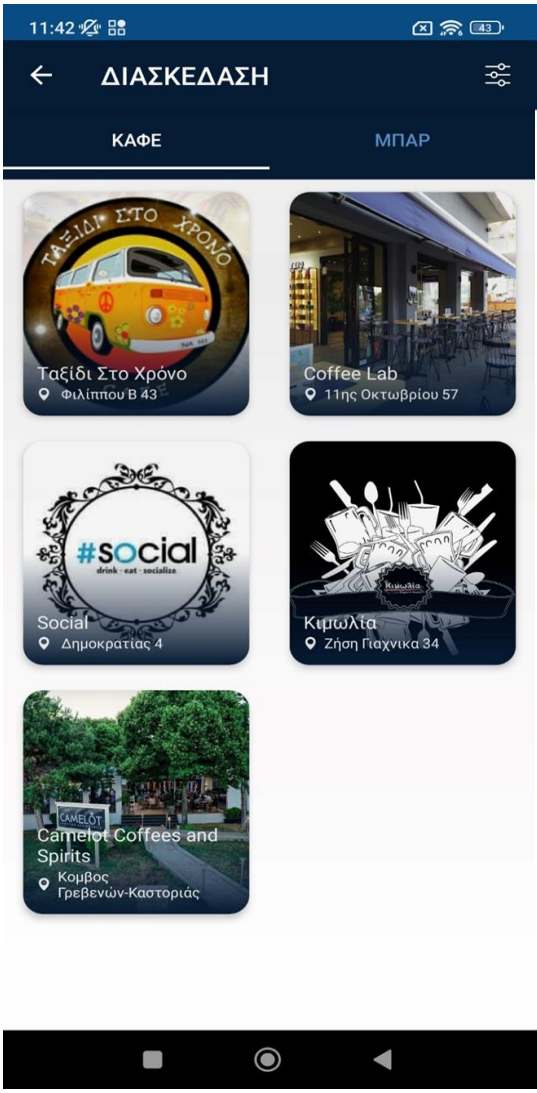


Εικόνα 5.5 Φίλτρο Αναζήτησης

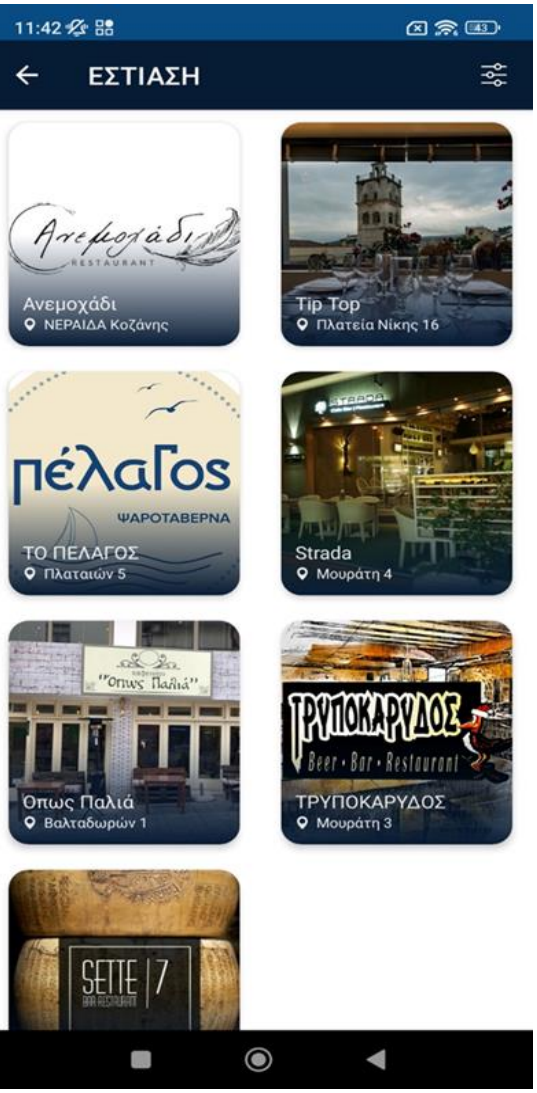
Πρώτη κατηγορία είναι αυτή της διαμονής. Επιλέγοντας τη κατηγορία αυτή εμφανίζονται στον χρήστη όλα τα διαθέσιμα ξενοδοχεία (εικόνα 5.4) με βάση την τοποθεσία του. Πρώτα εμφανίζονται τα πιο κοντινά. Αυτό είναι το προεπιλεγμένο φίλτρο αναζήτησης. Ο χρήστης παρόλα αυτά με το κουμπί πάνω δεξιά έχει τη δυνατότητα να επιλέξει το φίλτρο αναζήτησης που θα εμφανίζει τα ξενοδοχεία με αλφαβητική σειρά (εικόνα 5.5).

Το φίλτρο αναζήτησης με βάση την απόσταση είναι προεπιλεγμένο και ισχύει για όλες τις κατηγορίες.

5.2.2 Διασκέδαση και Εστίαση



Εικόνα 5.6 Διασκέδαση

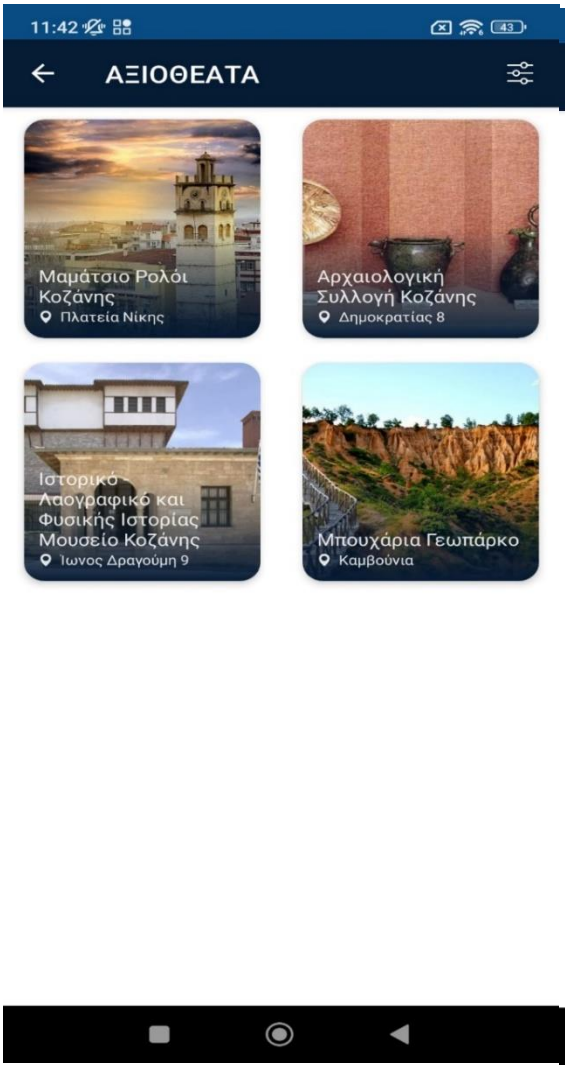


Εικόνα 5.7 Εστίαση

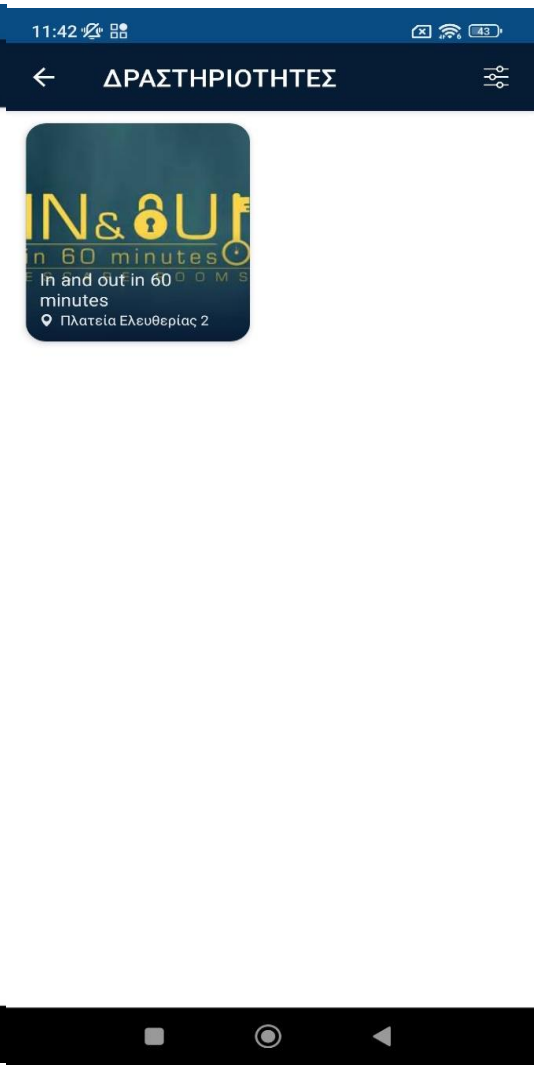
Στη κατηγορία της διασκέδασης ο χρήστης μπορεί να αναζητήσει κάποιο καφέ ή μπαρ κατά την επίσκεψη του στην Κοζάνη. Στο πάνω μέρος της εικόνας 5.5 υπάρχει ο διαχωρισμός σε υποκατηγορίες μπαρ – καφέ για ακόμα μεγαλύτερη ευκολία στη χρήση.

Στην κατηγορία εστίασης ο χρήστης μπορεί να ψάξει τα διαθέσιμα εστιατόρια της Κοζάνης.

5.2.3 Αξιοθέατα και Δραστηριότητες



Εικόνα 5.8 Αξιοθέατα

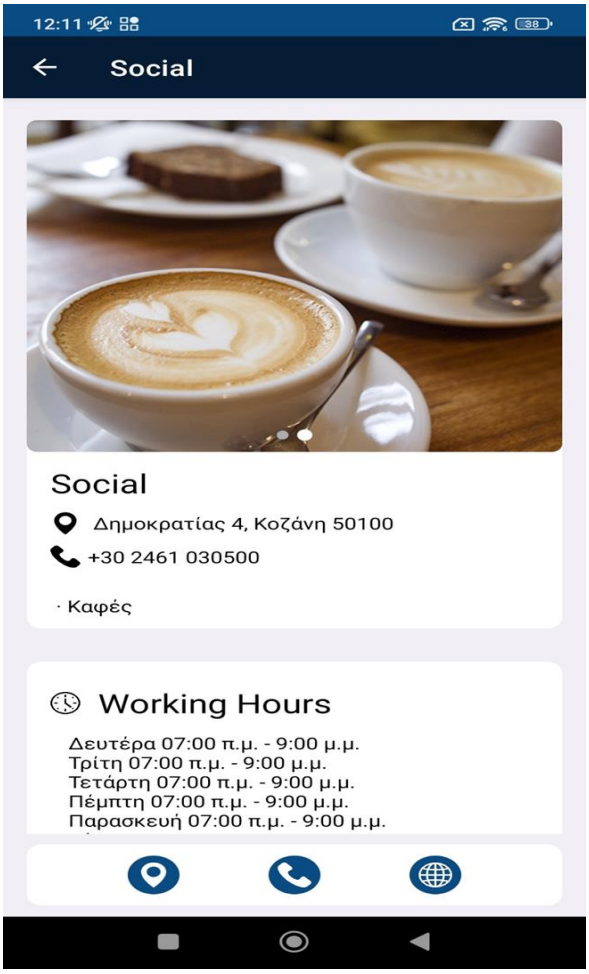


Εικόνα 5.9 Δραστηριότητες

Στη κατηγορία αξιοθέατα ο χρήστης μπορεί να εντοπίσει διαφορά σημεία ενδιαφέροντος που αξίζουν επίσκεψη (εικόνα 5.8).

Στη κατηγορία δραστηριότητες ο χρήστης μπορεί να ανακαλύψει διαφορά event και δραστηριότητες για να περάσει ευχάριστα τον χρόνο του(εικόνα 5.9).

5.2.4 Πληροφορίες κάθε καταστήματος



Εικόνα 5.10 Καφέ Social

Αφού ο χρήστης επιλέξει ποια κατηγορία επιθυμεί και βρει πιο κατάστημα τον ενδιαφέρει στην επόμενη σελίδα μπορεί να βρει πολλές πληροφορίες γι αυτό. Στην εικόνα 5.10 φαίνονται όλες οι πληροφορίες για το καφέ Social. Συγκεκριμένα ο χρήστης θα βρει κατά σειρά τις παραπάνω πληροφορίες

1) Εικόνες από το καφέ
2) Διεύθυνση
3) Τηλέφωνο επικοινωνίας
4) Περιγραφή
5) Ώρες λειτουργίας

Στο κάτω μέρος υπάρχουν 3 κουμπιά

Πατώντας το αριστερό κουμπί ο χρήστης μπορεί να πλοηγηθεί μέσω των χαρτών στο καφέ social. Με το μεσαίο κουμπί μπορεί να καλέσει αυτόματα τον αριθμό του καταστήματος από το κινητό του και με το δεξιά κουμπί μπορεί να επισκεφθεί τη σελίδα του καφέ εφόσον αυτή είναι διαθέσιμη με τη χρήση browser.

Το ίδιο ακριβώς ισχύει και για τις υπόλοιπες κατηγορίες που θα βρει ο χρήστης στην εφαρμογή.

6. Συμπεράσματα και μελλοντικές επεκτάσεις

Κατά την εκπόνηση της εργασίας καταλάβαμε ότι η ανάπτυξη μιας ολοκληρωμένης εφαρμογής τουριστικού οδηγού απαιτεί πάρα πολύ ερευνά και χρήση πολλών τεχνολογιών πάνω στων τομέα της πληροφορικής. Η δημιουργία της όμως έχει διευκολύνει πάρα πολύ τους ανθρώπους που θέλουν να επισκεφτούν την Κοζάνη.

Η εξέλιξη της τεχνολογίας και η ολοένα μεγαλύτερη ανάπτυξη του τουρισμού οδηγεί στην αναγκαιότητα τέτοιων εφαρμογών, ειδικά στη χώρα μας οπού ο τουρισμός είναι το νούμερο ένα εξαγώγιμο προϊόν μας

Στο μέλλον υπάρχει η δυνατότητα να προστεθούν πολύ περισσότερα σημεία ενδιαφέροντος για την Κοζάνη καθώς επίσης να βελτιωθεί η λειτουργικότητα της. Η εφαρμογή θα μπορούσε να επεκταθεί και να μην είναι μόνο για τη συγκεκριμένη πόλη αλλά και για άλλες της Ελλάδας. Επίσης μελλοντικά μπορούν να βελτιωθούν και άλλο τα γραφικά ώστε να γίνει ακόμα πιο φιλικό προς τον χρήστη και να γίνει και για το IOS λειτουργικό και να χρησιμοποιηθεί και σε Apple συσκευές.

Βιβλιογραφία

[1] J. DiMarzio, «android a programmers guide» *McGraw-Hill, Inc.*, (1st. ed.), USA (2008)

[2]J.Muir, «A short history of software development,» [Ηλεκτρονικό]:https://blogs.oregonstate.edu/articles/a-short-history-of-software-development/ [Πρόσβαση 11 Νοεμβρίου 2023].

[3]Android Studio, [Ηλεκτρονικό] : https://developer.android.com/studio/intro [Πρόσβαση 7 Νοεμβρίου 2023].

[4] Android studio, [Ηλεκτρονικό] : https://developer.android.com/studio/preview/gemini/data-and-privacy [Πρόσβαση 7 Νοεμβρίου 2023].

[5] Booking.com, [Ηλεκτρονικό] : https://www.booking.com/index.el.html

[6] Retrofit, [Ηλεκτρονικό]:https://www.digitalocean.com/community/tutorials/retrofit-android-example-tutorial [Πρόσβαση 7 Νοεμβρίου 2023].

[7]RestAPI, [Ηλεκτρονικό] : https://aws.amazon.com/what-is/restful- api/

[Πρόσβαση 10 Φεβρουαρίου 2024]

[8]RestAPI,[Ηλεκτρονικό]:https://www.ibm.com/topics/rest-apis [Πρόσβαση 10 Φεβρουαρίου 2024].

[9] D. Cassel, [Ηλεκτρονικό] : https://thenewstack.io/the-hardware-and-software-used-in-space/ [Πρόσβαση 7 Νοεμβρίου 2023].

[10] M. L. Gillenson, Βασικές Αρχές Συστημάτων Διαχείρισης Βάσεων Δεδομένων, 2η Έκδοση επιμ., Broken Hill Publishers LTD, 2012.

[11] Retrofit,[Ηλεκτρονικό]: https://square.github.io/retrofit/ [Πρόσβαση 7 Νοεμβρίου 2023].

[12] MySQLworkbench, [Ηλεκτρονικό] : https://www.mysql.com/products/workbench/ [Πρόσβαση 7 Νοεμβρίου 2023].

[13] MySQL workbench Install Instructions: https://dev.mysql.com/downloads/workbench/

[14] Hibernate, [Ηλεκτρονικό]:https://github.com/hibernate/hibernate-tools [Πρόσβαση 10 Φεβρουαρίου 2024].

[15] Hibernate, [Ηλεκτρονικό]: https://hibernate.org/search/tooling/ [Πρόσβαση 10 Φεβρουαρίου 2024].

[16] Postman, [Ηλεκτρονικό] : https://www.postman.com/product/what-is-postman/ [Πρόσβαση 7 Νοεμβρίου 2023].

[17] IntelliJIdea, [Ηλεκτρονικό]: https://www.jetbrains.com/idea/features/#intelligent-editor [Πρόσβαση 10 Φεβρουαρίου 2024].

[18] Intellij IDEA Install Instructions: https://www.jetbrains.com/idea/download/#section=windows

[19]Tripadvisor, https://tripadvisor.mediaroom.com/gr-about-us[Πρόσβαση 15 Μαρτίου 2024].