ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Σχεδίαση & Ανάπτυξη Εφαρμογής Android για ανάκτηση Θεατρικών δεδομένων»

Της φοιτήτριας.
Αθηνά Παπαχρήστου
Αρ. Μητρώο : 164725

Επιβλέπων
Ονοματεπώνυμο Σαλαμπάσης
Μιχαήλ
Βαθμίδα Καθηγητής

Ημερομηνία 31/08/2022

Τίτλος Δ.Ε. Σχεδίαση & Ανάπτυξη Εφαρμογής Android για το Θέατρο
Κωδικός Δ.Ε. 4160
Ονοματεπώνυμο φοιτητή/των Παπαχρήστου Αθηνά
Ονοματεπώνυμο εισηγητή Σαλαμπάσης Μιχαήλ
Ημερομηνία ανάληψης Δ.Ε. 04/2021
Ημερομηνία περάτωσης Δ.Ε. 09/2022

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

Περιεχόμενα

Πρόλογος

Η ανάπτυξη εφαρμογής σε android για ανάκτηση θεατρικών δεδομένων είναι ένα αρκετά μεγάλο συνεργατικό project που συμμετέχει μία ευρύτερη ομάδα φοιτητών, ο καθένας υλοποιώντας ένα διακριτό μέρος του πιλοτικού συστήματος. Αποτέλεσε μία ευκαιρία εκμάθησης για συνεργατική ανάπτυξη συστημάτων λογισμικού σε μεγάλη κλίμακα από αυτή που συνήθως συμβαίνει σε πτυχιακές εργασίες. Συγκεκριμένα στην δική μου πτυχιακή, με βάση την ανάλυση των στοιχείων που υπήρχαν σε μία βάση δεδομένων έπρεπε να βρεθούν ιδέες ώστε να παρουσιαστούν κατάλληλα και να υπάρχει αποτέλεσμα λειτουργικής εφαρμογής. Η ανάπτυξη της εφαρμογής της πτυχιακής μου εργασίας απαιτούσε εκμάθηση και χρήση του android studio του οποίου η χρησιμότητα είναι γνωστή σε πολλές εταιρείες οπότε η δυνατότητα χρήσης του φέρνει επαγγελματικές ευκαιρίες. Φυσικά, της ανάπτυξης της δικής μου εφαρμογής, ένα πρόσθετο σημαντικό κομμάτι που ξεχωρίζει την συγκεκριμένη πτυχιακή εργασία είναι η συνεργασία με την υπόλοιπη ομάδα φοιτητών που εκτελούσαν διαφορετικές πτυχιακές εργασίες. Επομένως η συγκεκριμένη πτυχιακή εκτός από εκμάθηση εργαλείων δίδαξε την προσομοίωση συνεργασίας η οποία θα συναντηθεί και σε επαγγελματικό επίπεδο.

Περίληψη

Android είναι ένα λογισμικό έξυπνων κινητών τηλεφώνων που βρίσκεται σε τεράστια χρήση σήμερα παγκοσμίως. Ο τρόπος ανάπτυξης εφαρμογών για android πέρασε από πολλά στάδια και πλέον υπάρχουν πολλά εργαλεία που χρησιμοποιούνται για να δημιουργηθεί μία εφαρμογή android. Το πιο γνωστό και έγκυρο εργαλείο που αφορά την ανάπτυξη android εφαρμογών είναι το android studio. Με τις πολλές λειτουργίες του και την εκτενή τεκμηρίωση που διαθέτει κατέληξε να είναι το εργαλείο που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής ανάκτησης θεατρικών δεδομένων. Το θέμα της εφαρμογής είναι αποτέλεσμα συνεργασίας μία ευρύτερης ομάδας φοιτητών που ξεκινούσε από το scraping για τη συλλογή των δεδομένων, την ανάπτυξη API για σύνδεση στη βάση και την επιστροφή των δεδομένων σε μορφή json, και τέλος την παρουσίαση αυτών δεδομένων στον τελικό χρήστη με μία web εφαρμογή, μία εφαρμογή σε IOS και τέλος σε Android, που αποτελεί το αντικείμενο αυτής της εφαρμογής. Χρησιμοποιήθηκαν εργαλεία όπως το figma και visual paradigm για την επίτευξη της καλύτερης συνεργασίας για την επίτευξη ενός κοινού αποτελέσματος με κοινή πορεία για τις τρεις από τις πέντε πτυχιακές που αφορούν τους τελικούς χρήστες, ενώ φυσικά απαιτήθηκε επικοινωνία και με την υπόλοιπη ομάδα, κυρίως με το μέρος του API, ώστε τα δεδομένα να έφταναν σε σωστή μορφή, σειρά και άλλα.

Έγινε εκτενής χρήση του android studio για την ανάπτυξη της εφαρμογής στο οποίο υπάρχουν διάφορες βιβλιοθήκες για πολλές λειτουργίες. Όπου υπήρχαν διαφορετικοί τρόποι υλοποίησης, έγινε σύγκριση για την τελική υλοποίηση ώστε να είναι πιο αποδοτικό το αποτέλεσμα. Το αποτέλεσμα που έπρεπε να επιτευχθεί είναι η έξυπνη παρουσίαση των δεδομένων θεατρικών παραστάσεων που υπήρχαν, με τρόπο φιλικό προς τους τελικούς χρήστες.

«Android app for Statistics of theatrical productions»

«Papachristou Athina»

Abstract

One team of 5 students work together as a wider team to present in a final user analytics about theater data recovery. From these 5 graduate theses there are three final products, via web app, IOS app and android app. These three apps take data from the same database. These data are sent in Json form by an api that another student has to make. So those three who develop the apps have to work together to find a similar path of how those data have to be shown and fully present in a smart and friendly for a user way. Visual Paradigm and Figma are the tools that has to be used for better communication and keep a record of the progress. Also, requirements have to be made so the one who makes the api send the data with correct form. My work is about developing the android app about theater data recovery application.

Android is a smart mobile phone software that is in huge use worldwide today. The way of android app development has gone through many stages and now there are many tools that are used to create an android app. The most well-known and authoritative tool for developing android applications is android studio. With its many features and extensive documentation, it ended up being the go-to tool for developing the theater data recovery application.

1 Κεφάλαιο : Στόχος της πτυχιακής

1.1 Στόχος Πτυχιακής

Μία ευρύτερη ομάδα φοιτητών με διαφορετικά θέματα πτυχιακών, όπου ο καθένας υλοποιώντας ένα διακριτό μέρος του πιλοτικού συστήματος συνέβαλε σε μία κοινή ιδέα ανάκτησης θεατρικών δεδομένων. Όλα ξεκίνησαν με την ανάκτηση δεδομένων για το θέατρο με τη μέθοδο scraping, με το πέρασμα αυτών των δεδομένων σε βάση δημιουργήθηκε η ανάγκη για υλοποίηση Api ώστε αυτά τα δεδομένα από τη βάση να μπορούν να συνδυαστούν και να παρουσιαστούν με Json μορφή. Αφού τα δεδομένα υπάρχουν χρειάζεται με κάποιο τρόπο να παρουσιαστούν στους τελικούς χρήστες. Έτσι δημιουργήθηκε η ανάγκη υλοποίησης εφαρμογών, δηλαδή άλλα τρία άτομα με τρεις πτυχιακές που θα υλοποιούσαν τις εφαρμογές web, σε IOS και σε android που είναι το θέμα αυτής της πτυχιακής.

Μέρος του στόχου της πτυχιακής ήταν και ο τρόπος παρουσίασης των δεδομένων. Καθώς μόνο η γνώση ότι υπάρχουν τα δεδομένα και οι συνδέσεις τους δεν φτάνει έπρεπε να ανακαλυφθούν έξυπνοι και φιλικοί προς τον τελικό χρήστη τρόποι ώστε να υπάρχει η καλύτερη δυνατή πληροφόρηση με τον καλύτερο δυνατό τρόπο, όπως σχεδιαγράμματα , αναζήτηση και άλλα. Οι ιδέες αυτές έπρεπε να γίνουν γνωστές και στην υπόλοιπη ευρύτερη ομάδα ώστε για παράδειγμα ο συνδυασμός που φτάνουν τα Json δεδομένα από το api να γίνει σύμφωνα με τις ανάγκες της εφαρμογής. Επιπλέον ο τρόπος παρουσίασης των δεδομένων θα έπρεπε να είναι κοινός για τα τρία τελικά προϊόντα οπότε ένα ακόμα μέρος της εργασίας είναι η χρήση εργαλείων για τη σχεδίαση οθονών καθώς και τη δημιουργία αιτημάτων για την καλύτερη λειτουργία της συνεργασίας.

Το θέμα της πτυχιακής είναι η ανάπτυξη android εφαρμογής για ανάκτηση θεατρικών δεδομένων, για την υλοποίησή της έγινε εκτενής χρήση του Android studio, το οποίο αναλύεται σε επόμενο κεφάλαιο η αιτιολογία της επιλογής του. Επιπλέον έγινε χρήση εργαλείων figma και visual Paradigm για την επίτευξη της καλύτερης συνεργασίας με τις υπόλοιπες πτυχιακές. Στο πρώτο έγινε η σχεδίαση των οθονών. Στο δεύτερο η συγγραφή των αιτημάτων και έγινε χρήση του και για την ίδια την android εφαρμογή για την οργάνωση του κώδικά της.

Επομένως η συγκεκριμένη πτυχιακή είχε ως στόχο εκτός από την εκμάθηση εργαλείων δίδαξε την προσομοίωση συνεργασίας η οποία θα συναντηθεί και σε επαγγελματικό επίπεδο.

2 Κεφάλαιο: Τι είναι android

2.1 Εισαγωγή

Το Android είναι μία από τις δημοφιλέστερες πλατφόρμες λειτουργικών συστημάτων. Ένα λειτουργικό σύστημα με τεράστια απήχηση τόσο στην προτίμηση των προγραμματιστών σαν μέσο δημιουργίας και ανάπτυξης εφαρμογών, όσο και στην εγκατάσταση του σε φορητές συσκευές όπως smartphones, smart TV's, tablets, κονσόλες παιχνιδιών, συστήματα αυτοκινήτων, ρολόγια και άλλες. Το Android τρέχει τον πυρήνα του λειτουργικού Linux και επιτρέπει τους προγραμματιστές να εργαστούν με την γλώσσα προγραμματισμού Java χρησιμοποιώντας κατάλληλες βιβλιοθήκες και χρήσιμα εργαλεία της Google. Ο προγραμματιστής έχει την δυνατότητα να δημιουργήσει μία εφαρμογή και να την διανείμει είτε δωρεάν είτε επί πληρωμή στο Play Store, ενώ ο τελικός χρήστης καλείται να επιλέξει ανάμεσα σε εκατομμύρια εφαρμογές διατεθειμένες να εξυπηρετήσουν και την πιο απαιτητική του προτίμηση. Συνεπώς, η χρήση του Android, μέσω των εφαρμογών, δίνει την δυνατότητα στον χρήστη για οργάνωση, διαχείριση, ενημέρωση αλλά και ψυχαγωγία με αποτέλεσμα να κάνει την εμπειρία του με την συσκευή πιο φιλική και διασκεδαστική.

2.2 Ιστορία

Τον Οκτώβριο του 2003 δημιουργήθηκε από τους Andy Rubin, Chris White, Nick Sears και Rich Miner το λειτουργικό σύστημα Android στο Palo Alto της Καλιφόρνιας από την εταιρία Android Inc. Πρόκειται για ένα λειτουργικό σύστημα, φιλικό προς τον χρήστη με βλέψεις της εταιρείας να το εγκαταστήσει σε ψηφιακές φωτογραφικές μηχανές καλλιεργώντας έτσι πρόσφορο έδαφος για επενδύσεις. Το 2004 το έργο διαφοροποιήθηκε και προοριζόταν πλέον σαν λειτουργικό σύστημα των smartphones ενώ τελικά η Android Inc αγοράστηκε το 2005 από την Google Inc. Παράλληλα, η Google Inc βάσισε τον έργο της σε Linux, εγκαθιστώντας το λειτουργικό σύστημα ανοιχτού κώδικα σε φορητές συσκευές. Στις 5 Νοεμβρίου 2007 ιδρύεται η Open Handset Alliance, μία κοινοπραξία πολλών εταιριών τεχνολογίας, όπου πρόκειται για μια επιχειρηματική οντότητα με πολλαπλά οφέλη για τις ίδιες τις εταιρείες που συμμετέχουν και αποσκοπούσε στην ανάπτυξη και προώθηση του Android, καθώς και στην υποστήριξη για εφαρμογές τρίτων. Μερικές από τις εταιρίες που συμμετείχαν ήταν οι Intel Corporation , Motorola, Inc., NVIDIA Corporation, Texas Instruments Incorporated, LG Electronics, Inc., Samsung Electronics, Sprint Nextel Corporation και T-Mobile (Deutsche Telekom). Έτσι, δημιουργείται μια σειρά δυνατοτήτων λόγου χάριν μηχανή αναζήτησης, δημιουργία και επεξεργασία υπολογιστικών φύλλων καθώς και η χρήση λογισμικού δορυφορικής χαρτογράφισης (Google Earth). Αξίζει να σημειωθεί ότι το πρώτο κινητό τηλέφωνο στο οποίο χρησιμοποιήθηκε το επαναστατικό πλεον λειτουργικό σύστημα ονομαζόταν T-Mobile G1, το οποίο κυκλοφόρησε στις 22 Οκτωβρίου 2008. Μέχρι το 2012 το Android κατάφερε να γίνει τόσο δημοφιλές που ξεπέρασε το λειτουργικό σύστημα iOS της Apple .

2.3 Αρχιτεκτονική του Android

Όπως φαίνεται στην εικόνα 1.1 το android αποτελείται από 5 στρώματα layers. Ξεκινάμε με τη βάση που είναι ο Linux kernel(πυρήνας) οποίος αποτελεί την καρδιά της αρχιτεκτονικής. Διαχειρίζεται τα προγράμματα οδήγησης ήχου, Bluetooth, κάμερας, οθόνης κ.λ.π και είναι υπεύθυνος για τη διαχείριση της κοινόχρηστης μνήμης και την πρόσβαση σε αυτήν. Σε αυτό το επίπεδο δεν θα χρειαστεί να φτάσει ο προγραμματιστής εφαρμογών, αφορά κυρίως το hardware και τη σύνδεση του με το software.

Συνεχίζοντας το επόμενο επίπεδο είναι το Hardware Abstraction Layer(HAL). Είναι μία ομάδα από πολλές βιβλιοθήκες οι οποίες υλοποιούν κάθε μία από ένα interface που υπάρχει για κάθε τύπο υλικού. Δηλαδή υπάρχει ένα interface για κάθε μέρος του υλικού που υποστηρίζει τις δυνατότητες του. Όταν μιλάμε για υλικό εννοούμε είτε κάμερα, ηχεία, αισθητήρες και άλλα. Αυτό χρησιμοποιείται από τις εφαρμογές διότι όταν για παράδειγμα μία εφαρμογή επιθυμεί να κάνει κλήση για πρόσβαση στην κάμερα του κινητού ουσιαστικά κάνει κλήση στην κατάλληλη βιβλιοθήκη που υλοποιεί το αντίστοιχο interface.

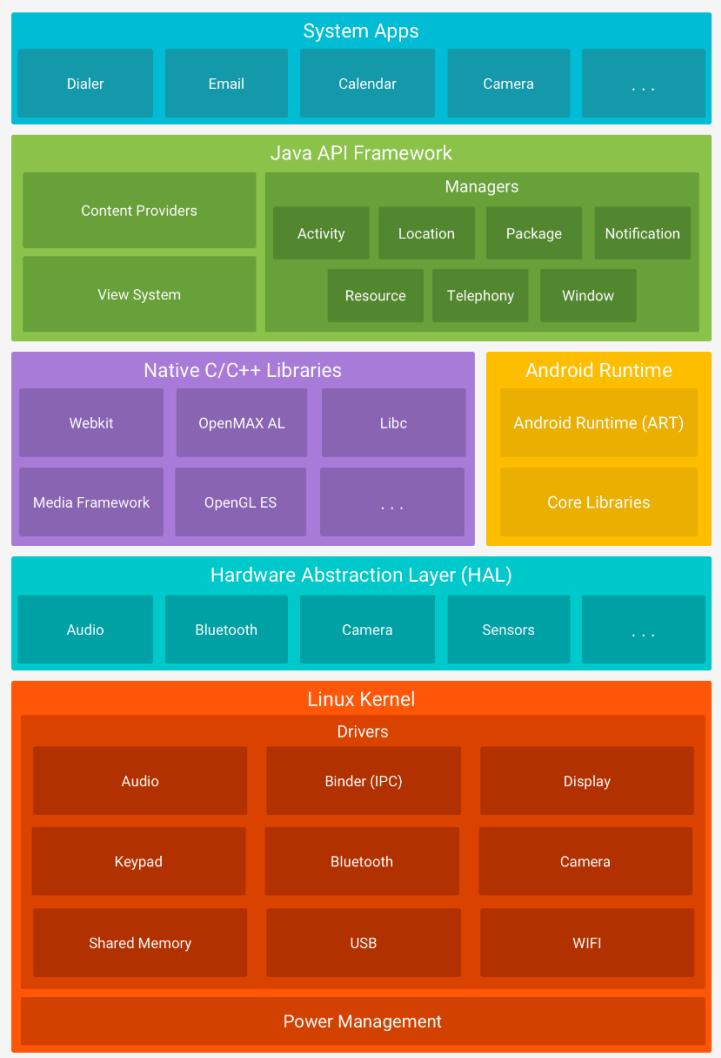
Αμέσως μετά είναι οι βιβλιοθήκες οι οποίες είναι γραμμένες σε C/C++. Αυτές χρησιμοποιούνται από διάφορα components και οι δυνατότητές τους προσφέρονται στους προγραμματιστές μέσα από το Android Application Framework, προκειμένου να χρησιμοποιηθούν σε κάποια εφαρμογή. Ένα παράδειγμα βιβλιοθήκης είναι OpenGL ES για την υποστήριξη και χρήση γραφικών στην εφαρμογή 2D και 3D.

Όσο ανεβαίνουν τα επίπεδα προς τα πάνω συναντάμε το Android runtime (ART) και τις βιβλιοθήκες πυρήνα. Είναι ο διαχειριζόμενος χρόνος εκτέλεσης που χρησιμοποιείται από εφαρμογές και ορισμένες υπηρεσίες συστήματος στο Android. Η έννοια του ART είναι ότι κάθε εφαρμογή εκτελεί τη δική της διαδικασία. Χρησιμοποιεί AOT (Ahead-Of-Time) compiler, το οποίο σημαίνει ότι όταν εγκαθίσταται μία εφαρμογή γίνεται compile ο κώδικάς της. Η διαδικασία αυτή υποστηρίζεται από το Android 5.0 και μετά και ο προκάτοχός του είναι Dalvik Virtual Machine Dalvik το οποίο είναι μια εικονική μηχανή (VM) που σχεδιάστηκες και γράφτηκε από τον Dan Bornstrein στη Google. Ο κώδικας της εφαρμογής μεταγλωττίζεται σε εντολές μηχανής που ονομάζονται bytecodes και στη συνέχεια εκτελούνται από την VM στο κινητό τηλέφωνο. Σε αυτή την περίπτωση χρησιμοποιείται η προσέγγιση JIT (Just in time) compiler, όπου ο κώδικας γίνεται compile σταδιακά σε κομμάτια. Και στις δύο περιπτώσεις υπήρχαν θετικά και αρνητικά για παράδειγμα η μέθοδος JIT χρησιμοποιεί λιγότερη μνήμη, για αυτό από το Android 7.0 χρησιμοποιείται ένας συνδυασμός των δύο compiler.

Το Application Framework παρέχει πλήρη προσβασιμότητα στους προγραμματιστές στα ίδια framework APIs που χρησιμοποιούνται από τις εφαρμογές πυρήνα. Έτσι συμπεριλαμβάνεται σε επίπεδο υψηλού επιπέδου και είναι γραμμένο σε Java. Μέρη αυτού παρέχονται από την Google και άλλα είναι επεκτάσεις των services των προγραμματιστών. Οι υπηρεσίες αυτές παρέχονται στη μορφή Java κλάσεων(Java classes).Κάποιες από τις βασικές υπηρεσίες είναι:

- Content Providers που επιτρέπουν στις εφαρμογές να έχουν πρόσβαση σε δεδομένα άλλων εφαρμογών (όπως επαφές), ή να διαμοιράζονται τα δικά τους δεδομένα.
- Resource Manager, που παρέχει προσβασιμότητα σε οτιδήποτε δεν είναι κώδικας, όπως αλφαριθμητικά, γραφικά και αρχεία που αφορούν την εμφάνιση της εφαρμογής
- Notification Manager, που επιτρέπει σε όλες τις εφαρμογές να εμφανίζουν ειδοποιήσεις στο status bar.
- Activity Manager, από την οποία παρέχονται πληροφορίες για την πλοήγηση των εφαρμογών

Πλέον βρισκόμαστε στο τελευταίο επίπεδο το οποίο είναι οι εφαρμογές System Apps οι οποίες χρησιμοποιούν τις γλώσσες Java και Kotlin. Παραδείγματα εφαρμογών είναι οι χάρτες ή αλλιώς google maps, διάφοροι email providers όπως το gmail και άλλα πολλά.



Εικόνα 1.1: Android Architecture

3 Κεφάλαιο: Android Studio

3.1 Εισαγωγή

Αφού αναλύσαμε στο παραπάνω κεφάλαιο πως ξεκίνησε το Android ήρθε στιγμή να αναλυθεί και το βασικό εργαλείο ανάπτυξης αυτών τω ειδών εφαρμογών που ονομάζεται Android Studio. Είναι το επίσημο IDE(integrated development environment) της Google. Κατασκευάστηκε από το IntelliJ IDEA της JetBrains. Η εγκατάστασή του γίνεται μέσω του Windows, Mac Os και Linux. Είναι ο απόγονος του Eclipse το οποίο ήταν το αρχικό IDE για την ανάπτυξη εφαρμογών.

3.2 Πως ξεκίνησε

Το Android Studio κυκλοφόρησε για πρώτη φορά σε μια διάσκεψη Google I/O το 2013, στις 16 Μαρτίου. Αργότερα, τον Μάιο του 2013, βρισκόταν σε φάση προεπισκόπησης ή έκδοση 0.1. Από εδώ, πέρασε στη φάση beta το 2014 της έκδοσης 0.8.

Η πρώτη σταθερή έκδοση του Android Studio δημιουργήθηκε τον Δεκέμβριο του 2014. Είναι επίσης γνωστή ως έκδοση 1.0.

Ένας άλλος παράγοντας που αξίζει να γίνει γνωστός είναι ότι στις 7 Μαΐου 2019, η Java, η πιο εγκεκριμένη γλώσσα προγραμματισμού για την ανάπτυξη εφαρμογών Android, αντικαταστάθηκε από την Kotlin. Η Java είναι ακόμα διαθέσιμη και υποστηρίζεται σε C++.

3.3 Πως Δημιουργείται μια εφαρμογή

Μετά την εγκατάσταση του Android studio στον υπολογιστή του ο προγραμματιστής το ανοίγει και το πρώτο πράγμα που του ζητείται είναι να δημιουργήσει νέο project δηλαδή εφαρμογή. Επιλέγει το όνομα του φακέλου του project και τη γλώσσα προγραμματισμού που επιθυμεί να χρησιμοποιήσει ανάμεσα στην Java και την Kotlin. Εφόσον έχουν γίνει οι απαραίτητες επιλογές που εμφανίζονται στην οθόνη εμφανίζεται το περιβάλλον που θα εργαστεί ο προγραμματιστής και θα συντάξει τον κώδικά του. Όπως φαίνεται στην εικόνα 2.1 δημιουργούνται κάποιοι φάκελοι μετά τη δημιουργία του project οι οποίοι χωρίζονται σε δύο σημεία ένα η εφαρμογή ο φάκελος Application στη συγκεκριμένη περίπτωση που παίρνει το όνομά του σύμφωνα με το όνομα του project και τον Grandle Scripts του οποίου η ανάλυση θα γίνει σε επόμενο κεφάλαιο. Όσο για αυτά που παρατηρούνται κάτω από τον φάκελο με το όνομα του project είναι:

- Manifest : android manifest είναι ένα xml αρχείο στο οποίο δίνονται τα permissions (οι άδειες) της εφαρμογής, δηλαδή ποια δεδομένα χρειάζεται να χρησιμοποιηθούν από άλλες εφαρμογές. Για παράδειγμα τη σύνδεση στο intertnet ή την πρόσβαση στην τοποθεσία. Καθορίζει επίσης το όνομα πακέτου της εφαρμογής που βοηθά το Android SDK κατά τη δημιουργία της εφαρμογής.
- Java: στον φάκελο αυτό είναι όλος ο κώδικας γραμμένος σε kotlin ή Java της εφαρμογής. Ουσιαστικά όλη η διαχείριση της εφαρμογής, για παράδειγμα μία απλή ενέργεια όπως το πάτημα ενός κουμπιού είτε η διαχείριση των δεδομένων και την εμφάνισή τους.
- Res: είναι ο φάκελος που έχει όλο το UI της εφαρμογής το οποίο είναι γραμμένο σε xml. Συμπεριλαμβάνει τα αρχεία xml στον φάκελο Layout που είναι όλες οι σχεδιασμένες οθόνες της εφαρμογής. Επιπλέον σε αυτόν τον φάκελο είναι όλες οι εικόνες και τα εικονίδια που φαίνονται στην εφαρμογή τα οποία ονομάζονται drawble. Στους υπόλοιπους φακέλους αυτής της ομάδας μπορεί να είναι τα κύρια χρώματα της εφαρμογής περασμένα σε παραμέτρους,

μορφοποιήσεις τίτλων και κειμένων αποθηκευμένα για επαναχρησιμοποίηση ακόμα και urls στα οποία δίνονται πιο σύντομα ονόματα από τον προγραμματιστή ώστε να χρησιμοποιούνται στον κώδικα πιο γρήγορα και είναι πιο εύκολη η συντήρηση σε περίπτωση αλλαγής.

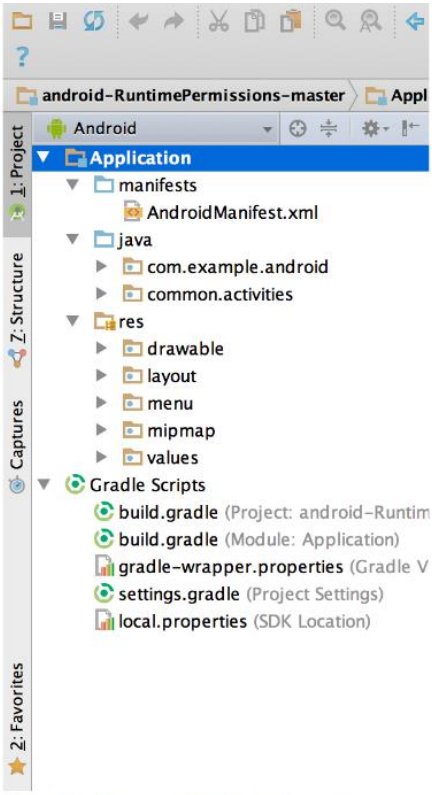


Figure 1. The project files in Android view.

Εικόνα 2.1

3.4 Γιατί Android Studio

Υπάρχουν πολλά προγραμματιστικά περιβάλλοντα (IDE) για την ανάπτυξη android εφαρμογών. Κάποια από αυτά είναι inteliJ, flatter το οποίο είναι και για IOS, android studio και ίσως σε προηγούμενες γενιές σε μεγαλύτερο βαθμό χρησιμοποιούταν το eclipse. Ωστόσο από όλα αυτά επιλέχθηκε για τις ανάγκες της πτυχιακής η ανάπτυξη της εφαρμογής σε android studio, το οποίο είναι και η επιλογή πολλών προγραμματιστών που ασχολούνται μόνο με android και δεν τους ενδιαφέρει η cross platform ανάπτυξη. Η έννοια cross platform παρουσιάζει τις εφαρμογές που προγραμματίζονται μία φορά και μπορούν να τρέξουν σε παραπάνω από ένα λειτουργικό για παράδειγμα μία εφαρμογή που τρέχει και σε android και σε IOS συσκευές.

Ένα από τα πιο βασικά πλεονεκτήματα είναι ότι το android studio είναι βασισμένο στο προγραμματιστικό περιβάλλον inteliJ και ειδικεύεται στην ανάπτυξη εφαρμογών σε android κινητά. Αυτό σημαίνει ότι είναι μια μοναδική πλατφόρμα που εκμεταλλεύεται πλήρως τις δυνατότητες μιας συμβατής συσκευής. Για παράδειγμα, οι προγραμματιστές εφαρμογών είναι ελεύθεροι να δημιουργήσουν εφαρμογές που χρησιμοποιούν οποιαδήποτε από τις βασικές λειτουργίες του τηλεφώνου όπως η αποστολή SMS, τηλεφωνικές κλήσεις, τη λήψη φωτογραφιών, το GPS κτλ. Γενικά αν και μόνο από την γνώση ότι είναι ένα περιβάλλον που φτιάχτηκε για το ζητούμενο, την ανάπτυξη

android εφαρμογών, θα μπορούσε να πείσει πολλούς προγραμματιστές να το χρησιμοποιήσουν υπάρχουν και άλλα θετικά.

Στο περιβάλλον του android studio υπάρχει σχεδιασμός οθόνης με drag and drop των διάφορων οπτικών items που μπορεί να έχει μία οθόνη εφαρμογής, όπως κουμπιά, labels, φωτογραφίες. Με αυτό τον τρόπο δίνεται η δυνατότητα στον προγραμματιστή να έχει εικόνα της οθόνης του πριν τρέξει την εφαρμογή.

Πολύ σημαντικό κομμάτι είναι η προσομοίωση. Παρέχεται έτοιμο emulator κινητού το οποίο μπορεί να επεξεργαστεί ο χρήστης και να το φτιάξει με τα χαρακτηριστικά που επιθυμεί ώστε να τρέξει το πρόγραμμά του. Αυτό δίνει την δυνατότητα στον προγραμματιστή να τρέξει, να χρησιμοποιήσει και να ελέγξει την εφαρμογή του από πολλές συσκευές και με την κάθε μία να έχει διαφορετικά χαρακτηριστικά, κάποιες από τις διαθέσιμες συσκευές που παρέχει το android studio είναι smartphone, tablets ακόμα και wearables.

Αυτό που δεν έχει αναφερθεί ακόμα είναι οι προγραμματιστικές ικανότητες. Το android studio προσφέρει έτοιμα templates τα οποία δημιουργούν από οθόνες μέχρι ολόκληρες εφαρμογές. Για παράδειγμα δίνεται η δυνατότητα δημιουργίας εφαρμογής χαρτών με μία απλή επιλογή. Έτσι καθίσταται ένα εύκολο περιβάλλον για να μάθει κάποιος χρησιμοποιώντας έτοιμο κώδικα είτε για αυτού με μεγαλύτερη εμπειρία να εξελίξουν πιο εύκολα τον κώδικά τους.

Αξίζει να σημειωθεί το intelligence του code editor το οποίο διαθέτει αυτόματη συμπλήρωση εντολών κώδικα, ενώ ταυτόχρονα συστήνει την καλύτερη πιθανή συγγραφή του για πιο ‘ευφυή’ αποτέλεσμα κώδικα. Αυτό έχει ως επακόλουθο την πιο γρήγορη και αποτελεσματικότερη συγγραφή του προγράμματος.

Εν κατακλείδι αν γίνουν αλλαγές στον κώδικα την ώρα που η εφαρμογή τρέχει σε πραγματική είτε εικονική συσκευή, δεν χρειάζεται να γίνει ούτε επανεκκίνηση ούτε rebuild, απλά ένα run και οι αλλαγές πλέον είναι ορατές στο προϊόν.

3.5 Gradle

Σημαντικό μέρος για την ανάπτυξη μίας android εφαρμογής είναι η βιβλιοθήκες και διάφορα εργαλεία που υπάρχουν διαθέσιμα για την πιο εύκολη και αποτελεσματική ανάπτυξή της. Όλα αυτά γίνονται πιο απλά και αυτοματοποιημένα με τη βοήθεια του Gradle. Είναι ένα προηγμένο σύνολο εργαλείων κατασκευής το οποίο επιτρέπει να γίνουν ευέλικτα προσαρμογές και καταστεί εύκολη τη διαμόρφωση της εφαρμογής στις αλλαγές των εκδόσεων για όλους τους πόρους της. Επίσης υπάρχει δυνατότητα επαναχρησιμοποίησης των κοινών μερών κατά τις αναβαθμίσεις των εκδόσεων.

Ουσιαστικά είναι ένα αρχείο κώδικα το οποίο συγκεντρώνει όλους τους πόρους. Χωρίζεται σε τρία μέρη plugins, android και dependencies. Η πρώτη ενότητα αφορά τη διαμόρφωση του build, εφαρμόζει την προσθήκη Android για Gradle στην αντίστοιχη έκδοση και καθιστά το μπλοκ που είναι διαθέσιμο. Η δεύτερη ενότητα διαμορφώνει τις επιλογές του build, για παράδειγμα ποιο application γίνεται compile, είτε την έκδοση SDK(software development kit)από τη μικρότερη μέχρι τη μεγαλύτερη που πρέπει να είναι διαθέσιμα από την συσκευή αυτό είναι ανάλογα με την έκδοση android της συσκευής που θα τρέξει η εφαρμογή. Η τελευταία ενότητα είναι και αυτή που χρησιμοποιείται περισσότερο αφορά τις βιβλιοθήκες. Με την εντολή implementation και το όνομα της βιβλιοθήκης το οποίο ακολουθείτε από τον αριθμό της έκδοσης γίνεται import. Παράδειγμα αρχείου Gradle είναι το παρακάτω απόσπασμα κώδικα.

```
plugins {
    id 'com.android.application'
}

android {
        compileSdkVersion 31 // <-- This
        defaultConfig {
            applicationId "com.example.theatrical_plays"
            targetSdkVersion 31 // <-- and this
            multiDexEnabled true
            minSdkVersion 19
            testInstrumentationRunner
"androidx.test.runner.AndroidJUnitRunner"
        }


    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-
optimize.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.4.2'
    implementation 'com.google.android.material:material:1.6.1'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation 'com.google.android.material:material:1.6.1'
    implementation 'androidx.legacy:legacy-support-v4:1.0.0'
    implementation 'com.google.code.gson:gson:2.8.6'
    implementation 'com.android.volley:volley:1.2.1'

}
```

Επιπλέον δυνατότητα είναι όπως αναφέρθηκε η καλύτερη διαχείριση του προγράμματος στις αναβαθμίσεις. Εκτός από την αναβάθμιση του εργαλείου Android studio υπάρχουν και οι αναβαθμίσεις των εκάστοτε βιβλιοθηκών που χρησιμοποιούνται. Έτσι σε κάθε νέα έκδοση κάποιας από αυτής ο προγραμματιστής ενημερώνεται με μία κίτρινη γραμμή στα αριστερά του κάθε import και το ίδιο το εργαλείο προτείνει την αλλαγή του αριθμού έκδοσης. Σε κάθε αλλαγή που γίνεται στο αρχείο Gradle της εφαρμογής για να μπορέσει να γίνει αντιληπτό στο πρόγραμμα ο προγραμματιστής χρειάζεται να πατήσει την επιλογή Sync now ώστε να γίνει και build στην εφαρμογή με τα καινούρια δεδομένα.

Τέλος για τη χρήση του Gradle απαιτείται μία φορά εγκατάσταση και υπάρχει ένα διαφορετικό αρχείο για κάθε εφαρμογή.

4 Κεφάλαιο : Αναβαθμίσεις Android Studio

4.1 Εισαγωγή

Ανά τα χρόνια από το 2013 που δημοσιεύθηκε το android studio υπάρχουν πολλές αναβαθμίσεις και συνεχώς κυκλοφορούσαν νέες εκδόσεις. Σχεδόν σε όλα τα έτη έχουν κυκλοφορήσει παραπάνω από μία εκδόσεις ανά έτος με το έτος 2015 να τα ξεπερνάει όλα με έντεκα εκδόσεις.

Σε κάποιες από αυτές μεταξύ προηγούμενης και επόμενης οι αλλαγές είναι μικρές, αλλά δεν παύουν να θεωρούνται σημαντικές, όπως διορθώσεις προβλημάτων είτε πιο φιλικό προς τον προγραμματιστή περιβάλλον. Παράδειγμα τέτοιων εκδόσεων είναι η 2.3 το Μάρτιο του 2017 η οποία σε σχέση την προηγούμενη της την 2.2 τον Σεπτέμβριο του 2016 είχε πιο αποδοτική λειτουργία του Instant Run, να γίνει δηλαδή Build και Run της εφαρμογής με ένα πάτημα. Κάποιοι χρήστες είχαν παρατηρήσει πρόβλημα σε αυτή τη λειτουργία και η εφαρμογή τους δεν έτρεχε, αυτό διορθώθηκε στην 2.3 μαζί με κάποιες άλλες μικρές λειτουργίες όπως είναι η μετατροπή διαφόρων αρχείων εικόνας (PNG, JPG κ.τ.λ) σε WebP η οποία προσφέρει καλύτερη συμπίεση των εικόνων.

Ωστόσο υπάρχουν και εκδόσεις όπου οι προσθήκες ήταν πολλές και οι δυνατότητες αυξάνονταν. Μία από αυτές ήταν η έκδοση 3.6 τον Φεβρουάριο του 2020. Πλέον μπορεί ο προγραμματιστής μόνο με την γλώσσα προγραμματισμού Kotlin να επισυνάψει εξωτερικές πηγές APK. Το οποίο σημαίνει ότι έγινε το πρώτο βήμα για την Kotlin ως γλώσσας προγραμματισμού για Android.

Στη συνέχεια αυτού του κεφαλαίου θα παρουσιαστούν οι εκδόσεις μεταξύ τον Απρίλιο του 2021 μέχρι σήμερα διότι είναι αυτές που πέρασαν κατά τη δημιουργία της εφαρμογής ανάκτησης θεατρικών δεδομένων, η οποία δημιουργήθηκε στα πλαίσια της πτυχιακής.

4.2 Android 4.2

Η πρώτη έκδοση που χρησιμοποιήθηκε για την εφαρμογή ήταν η 4.2, η οποία έκανε την εμφάνισή της τον Απρίλιου του 2021. Σε αυτή την έκδοση σύμφωνα με προστέθηκε η δυνατότητα να μπορεί ο προγραμματιστής να εκτελεί την εφαρμογή του ταυτόχρονα σε περισσότερες από μία συσκευές. Επιπλέον, έγινε εφικτή η λειτουργία apply changes με την οποία κατά την εκτέλεση της εφαρμογής σε μία συσκευή μπορεί να αλλαχθεί ο κώδικας και η αλλαγή να φανεί κατευθείαν στο output. Ωστόσο υπήρχαν και μερικά προβλήματα με αυτή την έκδοση, το πιο σημαντικό για το οποίο δόθηκε μία λύση ήταν ότι δεν άνοιγαν αποθηκευμένα project από προηγούμενη έκδοση.

4.3 Android studio Arctic Fox

Πλέον αλλάζει η αρίθμηση των εκδόσεων και προστίθενται καινούρια features. Ένα από αυτά είναι η δυνατότητα προσαρμογής layout σε διάφορες οθόνες ώστε η οθόνη της εφαρμογής να προσαρμόζεται στις διάφορες συσκευές και διαστάσεις. Το αναπτυσσόμενο μενού συσκευής αναγνωρίζει επί του παρόντος μεταξύ διαφορετικών ειδών σφαλμάτων μέσα στη ρύθμιση της συσκευής που έχει επιλεχθεί. Η εικονογραφία και οι στιλιστικές αλλαγές κάνουν πλέον διαφοροποίηση μεταξύ σφαλμάτων (επιλογές συσκευής που σε χαλασμένη ρύθμιση) και προειδοποιήσεων (επιλογές συσκευής που που θα οδηγήσουν σε εκπληκτική συμπεριφορά, αλλά εξακολουθούν να μπορούν να εκτελεστούν). Επιπλέον, το Android Studio θα προειδοποιεί εάν γίνει προσπάθεια να γίνει εκκίνηση εφαρμογής σε μια συσκευή που έχει ένα σφάλμα ή μια προειδοποίηση που σχετίζεται με αυτό. Επιπροσθέτως πλέον υπάρχει testing μεταξύ πολλαπλών συσκευών ταυτόχρονα και η ανάλογη εμφάνιση της επίδοσης για

την κάθε μία. Αυτά είναι κάποιες από τις καινούριες λειτουργίες που έχουν προστεθεί με την καινούρια έκδοση.

4.4 Android studio Bumblebee

Η νέα αυτή αναβάθμιση ανακοινώθηκε μέσα στο 2022 και η τελευταία του android studio μέχρι τώρα. Μία σημαντική νέα προσθήκη αυτής είναι η εκτέλεση της εφαρμογής και το debugging σε φυσική συσκευή χωρίς καλώδιο, το μόνο που απαιτείται είναι ο υπολογιστής και συσκευή να είναι συνδεδεμένες στο ίδιο δίκτυο, με το σκανάρισμα ενός qr κωδικού που παρέχεται από το android studio μπορεί να γίνει η σύνδεση της φυσικής συσκευής. Ενώ δεν λείπει η αναβάθμιση στης προεπισκόπηση του layout της εφαρμογής. Πλέον υπάρχει δυνατότητα ο προγραμματιστής να βγάζει στιγμιότυπα των οθονών για να τις συγκρίνει με τη εφαρμογή του που τρέχει.

5 Κεφάλαιο: Η αρχή της εφαρμογής

5.1 Βασική ιδέα

Η εφαρμογή ανάκτησης θεατρικών δεδομένων είναι ένα μέρος από ένα μεγαλύτερο project που αποτελείται από πέντε διαφορετικές πτυχιακές. Η εφαρμογή δημιουργήθηκε για την παρουσίαση των δεδομένων της βάσης με τρόπο όσο το δυνατόν έξυπνο και φιλικό προς τον τελικό χρήστη. Επομένως όλα ξεκίνησαν με το scrapping όπου συμπλήρωνε τη βάση με δεδομένα, και συνεχίστηκε με την υλοποίηση του api μέσω του οποίου φτάνουν τα δεδομένα σε μορφή Json. Η ομάδα συνεχίζεται με την υλοποίηση της web εφαρμογής για τον υπολογιστή καθώς και την υλοποίηση σε IOS και Android εφαρμογών. Με αποτέλεσμα για το καλύτερο τελικό αποτέλεσμα να υπάρχει η συνεργασία και συχνή επικοινωνία για κοινό σκοπό και εμφάνιση δεδομένων των τελευταίων τριών και συνεννόηση με την api υλοποίηση ώστε τα δεδομένα να εμφανίζονται σωστά και με τις παραμέτρους που απαιτούνται. Με αποτέλεσμα να έχουν γίνει αρκετές ομαδικές συναντήσεις ώστε να συμπληρωθούν τα κατάλληλα αιτήματα για τον τρόπο που θα έπρεπε να περνιούνται τα δεδομένα ώστε να φτάνει όσο το δυνατόν καλύτερη και πιο κατανοητή η πληροφορία στο χρήστη μέσω της εφαρμογής.

Το κομμάτι της συνεργασίας ωστόσο δεν περιορίστηκε μόνο στο πλαίσιο των αιτημάτων προς τον διαχειριστή του api καθώς ήταν αναγκαίο τα front end κομμάτια του project μεταξύ τους να έχουν κοινή πορεία.

5.2 Περιγραφή βάσης

Για να γίνει πιο κατανοητό το περιεχόμενο της εφαρμογής αξίζει να σημειωθούν τα δεδομένα της βάσης που χρησιμοποιήθηκε, ουσιαστικά οι συνδέσεις μεταξύ των πινάκων στις οποίες στηρίχθηκε η εφαρμογή και έδωσαν ιδέες για την υλοποίηση των τεχνολογιών και την αναπαράσταση των στοιχείων. Παρακάτω παρουσιάζεται το διάγραμμα της βάσης στην εικόνα 4.1

5 Κεφάλαιο

Εικόνα 4.1: ER Diagram

Οι βασικοί πίνακες που αποτελούν και τα βασικά αντικείμενα της εφαρμογής είναι οι παραστάσεις ο πίνακας production ο οποίος έχει πληροφορίες για κάθε παράσταση, το όνομα της, την περιγραφή της τη διάρκεια και άλλα. Η κάθε παράσταση συμμετέχει και σε μια διοργάνωση ο πίνακας events στον οποίο είναι οι διοργανώσεις και οι πληροφορίες αυτών όπως τιμή εισιτηρίου της παράστασης καθώς και τα διάφορα θέατρα στα οποία παίζονται. Ο συνδυασμός αυτός δίνει την ευκαιρία να αναφερθεί το δεύτερο βασικό αντικείμενο της εφαρμογής το οποίο είναι τα θέατρα τα οποία έχουν διεύθυνση και τίτλο. Συνεχίζοντας υπάρχει το τρίτο και τελευταίο αντικείμενο οι ηθοποιοί ο πίνακας persons, όπου εκτός από τίτλο έχουν ρόλους οι οποίοι αλλάζουν ανάλογα με την παράσταση που επιλέγεται όπως φαίνεται η σύνδεση αυτή αναφέρεται στον πίνακα contributions που συνδυάζει παραστάσεις και ηθοποιούς. Με αυτούς τους συνδυασμούς έγινε η υλοποίηση της εφαρμογής και τα στοιχεία αυτά προκλήθηκαν ιδέες για την πιο αποτελεσματική παρουσίαση των πληροφοριών.

6 Κεφάλαιο: Η εφαρμογή

6.1 Γρήγορη ξενάγηση στην εφαρμογή

Το τελικό αποτέλεσμα της εφαρμογής ανάκτησης δεδομένων για το θέατρο είναι η αναπαράσταση θεατρικών στοιχείων ηθοποιοί, παραστάσεις και θέατρα. Αποτελείται από 4 βασικές οθόνες οι οποίες γίνονται σχεδόν αμέσως αντιληπτές από το χρήστη με το άνοιγμα της εφαρμογής και 5 υποοθόνες που ανακαλύπτει ο χρήστης κατά τη χρήση της εφαρμογής. Η κάθε οθόνη εμφανίζει διαφορετικά στοιχεία και δίνει με διαφορετικό τρόπο τη δυνατότητα στο χρήστη να αποκτήσει μία γενικότερη εικόνα των κύριων οντοτήτων της εφαρμογής. Στόχος είναι η καλύτερη και πιο αποδοτική αναζήτηση μεταξύ στοιχείων και όσο το δυνατόν ορθότερη και στοχευμένη πληροφόρηση του τελικού χρήστη.

Η πρώτη εικόνα όταν ανοίγει κάποιος την εφαρμογή είναι η αρχική οθόνη η οποία έχει δειγματοληπτικά από τρεις παραστάσεις , θέατρα και ηθοποιοί και ένα κουμπί περισσότερα στο καθένα που το οδηγεί σε λίστες είτε ηθοποιών είτε παραστάσεων είτε θεάτρων αντίστοιχα. Με αυτό τον τρόπο γίνεται αντιληπτό το θέμα της εφαρμογής και γίνεται μία ξενάγηση ώστε με ευκολία και στοχευμένα πλέον ο χρήστης μπορεί να αναζητήσει αυτό που επιθυμεί. Επιπλέον υπάρχει στο κάτω μέρος της οθόνης μενού το οποίο πλοηγεί το χρήστη στις βασικές οθόνες.

Οι άλλες τρεις κύριες οθόνες είναι λίστες στη μία είναι λίστα από ηθοποιούς όπου με την επιλογή του ενός οδηγεί στην ξεχωριστή σελίδα του με πληροφορίες για αυτόν καθώς δίνεται η η δυνατότητα αναζήτησης σύμφωνα με το όνομα του ηθοποιού. Με την ίδια λογική είναι και οι υπόλοιπες δύο οθόνες με λίστα παραστάσεων και θεάτρων αντίστοιχα που με την επιλογή ενός συγκεκριμένου αντικειμένου ανοίγει σελίδα με πληροφορίες για το καθένα και εδώ υπάρχει αναζήτηση. Ωστόσο υπάρχουν και άλλες δυνατότητες όπως σύγκριση μεταξύ των παραστάσεων με βάση τον αριθμό διοργανώσεων που έχουν γίνει για αυτές, αυτό αναπαρίσταται με διάγραμμα και σύγκριση εισιτηρίων. Επίσης διάγραμμα υπάρχει και στα θέατρα διότι και εκεί υπάρχει σύγκριση σύμφωνα με των αριθμό των παραστάσεων που έχουν γίνει σε αυτά ώστε να υπάρχει και η στατιστική αξιολόγηση στην πληροφορία. Στα επόμενα κεφάλαια θα γίνει ανάλυση αυτών των οθονών καθώς και τις τεχνολογίες που χρησιμοποιήθηκαν για την δημιουργία τους.

6.2 Παρουσίαση δεδομένων Json

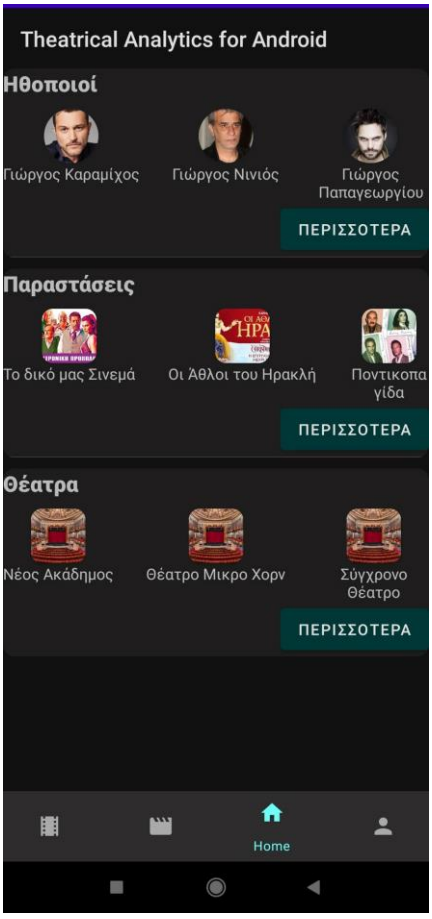
Τα δεδομένα της εφαρμογής φτάνουν σε αυτή με Json μορφή με http request με αποτέλεσμα όλη η εφαρμογή να δημιουργείται δυναμικά. Η παρουσίαση τους έγινε με τη βοήθεια βιβλιοθήκης του android studio που ονομάζεται volley σε συνδυασμό με τη βιβλιοθήκη gson οι οποίες αναλύονται στο κεφάλαιο 6. Οι πίνακες Json έχουν τις πληροφορίες size όπου είναι ο αριθμός πληροφοριών του πίνακα το οποίο κάνει εύκολο τη διαχείριση των δεδομένων από το πρόγραμμα όσα και αν είναι αυτά.

6.3 Αρχική οθόνη

Η πρώτη εικόνα του χρήστη με το άνοιγμα της εφαρμογής είναι η αρχική οθόνη. Η οποία αποτελείται από ένα fragment το οποίο ονομάζεται home_fragment και ένα bottom navigation το οποίο ανήκει στο main activity του project (κεφάλαιο 6 bottom navigation). Η οθόνη αυτή αποτελείται από τρία cardView στα οποία έχει δοθεί attribute cardCornerRadius εξού και το οι στρογγυλές άκρες για καλύτερο διαχωρισμό της οθόνης. CardView είναι ένα item tag στο xml αρχείο το οποίο μπορεί να πάρει όποια μορφή και σχήμα επιθυμεί ο προγραμματιστής και μέσα σε αυτό μπορούν να μπουν άλλα αντικείμενα όπως κουμπιά φωτογραφίες και άλλα.

Κάθε cardView έχει από τρία αντικείμενα τα οποία αποτελούνται από τον τίτλο παράστασης είτε όνο ηθοποιού είτε όνομα θεάτρου αντίστοιχα και μια εικόνα η οποία αναπαρίσταται με imageView(άλλο ένα item tag) και φορτώνεται από url μέσω της βιβλιοθήκης picasso η οποία έχει χρησιμοποιηθεί για την εμφάνιση όλων των φωτογραφιών καθώς όλες οι φωτογραφίες της βάσης είναι σε μορφή url από το διαδίκτυο. Επιπροσθέτως όλα τα στοιχεία τίτλος-εικόνα έχουν τη δυνατότητα να επιλεχθούν και να εμφανιστεί η αντίστοιχη οθόνη που έχει πληροφορίες για το καθένα.

Συνεχίζοντας στο τέλος κάθε cardView υπάρχει ένα κουμπί περισσότερα αυτό οδηγεί στην κεντρική οθόνη είτε των παραστάσεων είτε ηθοποιών είτε θεάτρων που θα αναλυθούν παρακάτω. Με αυτή την οθόνη ο χρήστης έχει μία πρώτη εντύπωση της εφαρμογής και σκοπός της ήταν να γίνει όσο πιο εύκολη γίνεται η εξήγηση της εφαρμογής και η πληροφορία που προσφέρει.



Εικόνα 5.1: Αρχική οθόνη

6.4 Οθόνη ηθοποιών

Βρίσκεται στο τέλος του navigation menu με το icon ανθρώπου, εδώ παρουσιάζεται η λίστα των ηθοποιών. Είναι όλοι οι ηθοποιοί της βάσης και η πληροφορία που εμφανίζεται είναι η φωτογραφία και το όνομα του καθενός η φωτογραφία του ηθοποιού χρησιμοποιεί τη βιβλιοθήκη Picasso διότι είναι internet url ενώ σε περίπτωση που δεν υπάρχει φωτογραφία για αυτό τον ηθοποιό υπάρχει default εικόνα. Η λίστα έχει τη μορφή recyclerView του android studio, η οποία είναι μία βιβλιοθήκη με την ίδια ονομασία αναλύεται στο κεφάλαιο 4. Για να μην υπάρχει αργή ανταπόκριση της οθόνης ανάλογα με το μέγεθος της πληροφορίας δεν φορτώνονται όλοι οι ηθοποιοί από την αρχή υπάρχει στο τέλος της σελίδας φόρτωση περισσότερων το οποίο φορτώνει 50 ηθοποιούς τη φορά, ώστε αν υπάρξουν περισσότερα δεδομένα στη βάση η εφαρμογή να μην επηρεαστεί.

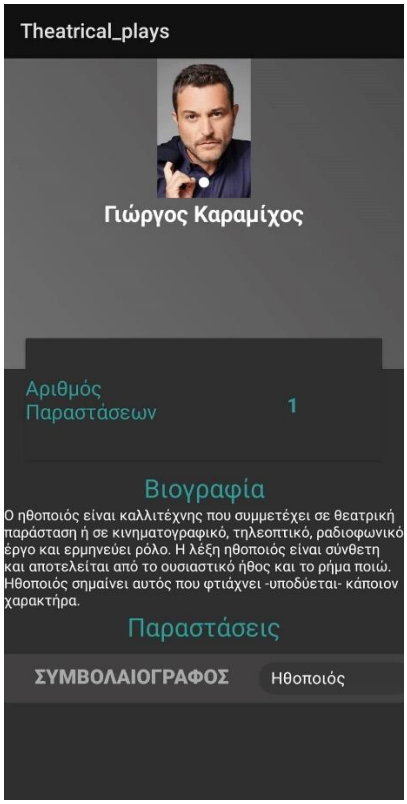
Μία ακόμα σημαντική λειτουργία της συγκεκριμένης οθόνης είναι η δυνατότητα αναζήτησης μεταξύ των ηθοποιών με βάση τον τίτλο τους. Ο χρήστης στο πάνω μέρος της οθόνης εκεί που βρίσκεται ο τίτλος μπορεί να πατήσει το εικονίδιο της αναζήτησης που είναι ο μεγεθυντικός φακός γνωστή εικόνα από άλλες ιστοσελίδες και εφαρμογές για την αναζήτηση. Έτσι εμφανίζεται το πληκτρολόγιο του κινητού και ένα πλαίσιο κειμένου με κέρσορα ώστε να χρήστης να γράψει το όνομα του ηθοποιού είτε μέρος αυτού και να πατήσει υποβολή. Η υποβολή γίνεται με ένα κουμπί κάτω δεξιά στο πληκτρολόγιο το οποίο ορίστηκε από την εντολή : searchView.setImeOptions(EditorInfo.IME_ACTION_DONE), η οποία προέρχεται από την εισαγωγή της android.view.inputmethod.EditorInfo η οποία γίνεται import στον κώδικα του fragment και 'διαχειρίζεται' την αναζήτηση και δίνει τη δυνατότητα το πληκτρολόγιο να έχει μορφή υποβολής και αυτό επιτυγχάνεται με το κουμπί κάτω δεξιά στο πληκτρολόγιο το οποίο το χρησιμοποιεί ο χρήστης αφού έχει ολοκληρώσει τη λέξη που επιθυμεί να αναζητήσει ώστε να υποβάλει την αυτό που επιθυμεί να αναζητήσει. Με αυτό πυροδοτεί ένα http request με παράμετρο ότι έχει γραφτεί από το χρήστη, το οποίο αναζητά τους ηθοποιούς έτσι αλλάζει η λίστα των ηθοποιών και τη θέση της παίρνει μία λίστα που πληροί τα κριτήρια της αναζήτησης. Παρακάτω παρουσιάζεται παράδειγμα κώδικα της αναζήτησης με τη χρήση της onQueryTextSubmit η οποία καλείται όταν γράφει ο χρήστης αυτό που θέλει να αναζητήσει και πατάει το κουμπί submit στο πληκτρολόγιό του.

```
   public boolean onQueryTextSubmit(String query) {
1                  if(query != null) {
2                      mRecyclerViewItems.clear();
3                      QUEUE = Volley.newRequestQueue(getContext());
4                      URLHTTP =
5  "http://195.251.123.174:8080/api/people/search?q=fullName~" + query ;
6                      httpGET(URLHTTP);
7                      setHasOptionsMenu(true);
8                      return true;
9                  }
10                 else {
11                     mRecyclerViewItems.clear();
12                     QUEUE = Volley.newRequestQueue(getContext());
13                     URLHTTP =
14 getResources().getString(R.string.urlserver);
15                     httpGET(URLHTTP);
16                     setHasOptionsMenu(true);
17                     return true;
18                 }
               }
```

Για κάθε ηθοποιό υπάρχει δυνατότητα να γίνει κλικ πάνω στη θέση του και να εμφανιστεί η οθόνη με της πληροφορίες του για να καταλάβει η εφαρμογή ποιον ηθοποιό επιθυμούμε να δούμε τις πληροφορίες περνάει με intent από το fragment στο activity (οθόνη του ηθοποιού) το id του ηθοποιού με το οποίο έχουμε τη δυνατότητα με http request να ζητήσουμε τις πληροφορίες του. Η μεταφορά των μεταβλητών από τη μία οθόνη σε μία άλλη στην προκειμένη περίπτωση από fragment σε activity γίνεται με τη βοήθεια του intent η οποία αναλύεται στο κεφάλαιο 6.2.4.

Στην νέα οθόνη που εμφανίστηκε υπάρχει στο πάνω μέρος της οθόνης ένα slider με φωτογραφίες του ηθοποιού η οποίες έρχονται από τη βάση ως url σε μορφή Json με http request το οποίο δέχεται ως παράμετρο το id του ηθοποιού. Η εμφάνιση αυτή έγινε με τη βοήθεια της βιβλιοθήκης denzcoskun:ImageSlideshow. Πιο κάτω στην οθόνη υπάρχει ο αριθμός των παραστάσεων που έχει

συμμετάσχει ο ηθοποιός το οποίο είναι αποτέλεσμα αντίστοιχου http request που δέχεται και αυτό ως παράμετρο το id του ηθοποιού. Σε αυτή την περίπτωση χρησιμοποιήθηκε η μεταβλητή size που υπάρχει στα Json δεδομένα και αντιστοιχεί στον αριθμό εγγραφών δηλαδή τον αριθμό παραστάσεων. Επιπλέον υπάρχει και η βιογραφία του κάθε ηθοποιού σε μορφή κειμένου. Ενώ δεν θα μπορούσε να λείπει και recyclerView με τις παραστάσεις που έχει συμμετάσχει και το ρόλο που είχε σε κάθε μία από αυτές, άλλη μία χρήση ενός end point από την υλοποίηση του api που δέχεται ως παράμετρο το id του ηθοποιού. Στις παραστάσεις αυτές αν γίνει κλικ σε μία από αυτές οδηγείται ο χρήστης στη σελίδα με τις πληροφορίες της παράστασης που αναλύεται στο υποκεφάλαιο 5.7.



Εικόνα 5.2: Οθόνη Ηθοποιού

6.5 Οθόνη παραστάσεων

Στο μενού της εφαρμογής με την επιλογή του εικονιδίου της κινηματογραφικής πλακέτας το fragment αλλάζει και πλέον ο χρήστης βρίσκεται στην οθόνη των παραστάσεων. Εκεί ως πρώτη εικόνα είναι μία λίστα με παραστάσεις οι οποίες όπως και στους ηθοποιούς εμφανίστηκαν με recyclerview. Η κάθε γραμμή παράστασης περιέχει τον τίτλο, τη διάρκεια της και στην δεξιά μερία ένα checkbox το οποίο θα αναλυθεί παρακάτω για ποιο λόγο χρησιμοποιείται. Για καλύτερη διαχείριση των δεδομένων οι παραστάσεις δεν φορτώνονται όλες αλλά σε πακέτα των 50 δηλαδή σε κάθε scroll down που κάνει ο χρήστης και φτάνει τις 50 παραστάσεις χρειάζεται να πατήσει το κουμπί περισσότερα ώστε να φορτωθούν οι άλλες 50.

Στο πάνω μέρος της οθόνης υπάρχουν δύο κουμπιά το πρώτο πάνω κουμπί που έχει τίτλο φίλτρα με το πάτημα του γίνεται expand (διευρύνεται) η οθόνη και εμφανίζεται ένα LinearLayout με ένα Slider δηλαδή ολισθητή όπου οι τιμές αναπαριστούν δευτερόλεπτα που αφορούν την διάρκεια της παράστασης, ένα checkbox με τίτλο ‘Πρόσφατες Παραστάσεις’ και δύο κουμπιά ένα με τίτλο

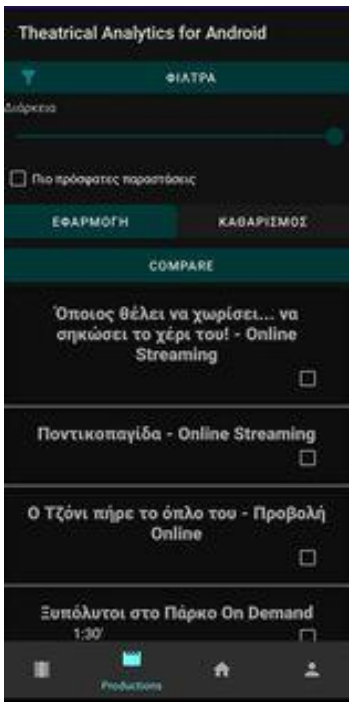
'Εφαρμογή' και ένα με τίτλο 'Καθαρισμό'. Η διευρυμένη οθόνη ουσιαστικά έχει τον ρόλο φίλτρων για τις παραστάσεις, τα φίλτρα όπως γίνεται αντιληπτό είναι διάρκεια των παραστάσεων και η εμφάνιση των πιο πρόσφατων. Η εφαρμογή των φίλτρων γίνεται με το πάτημα του κουμπιού 'Εφαρμογή' ενώ με το πάτημα του κουμπιού καθαρισμός γυρίζουν όλα στην αρχική τους μορφή, δηλαδή ο ολισθητής αποκτά τη μέγιστη τιμή και το checkbox δεν είναι πλέον επιλεγμένο, όπως και η λίστα των παραστάσεων έχει τα αρχικά δεδομένα και με τη σειρά που εμφανίστηκαν όταν ανοίχτηκε η οθόνη. Όταν ξαναπατηθεί το κουμπί 'Φίλτρα' το κομμάτι που διευρύνθηκε γυρνάει στην αρχική του μορφή και δεν είναι πλέον ορατό στο χρήστη.

Τι είναι όμως αυτά τα φίλτρα και πως λειτουργούν. Στο φίλτρο της διάρκειας ανάλογα με τον αριθμό που έχει 'σύρει' ο χρήστης στον ολισθητή εμφανίζονται και οι παραστάσεις μέχρι αυτή τη διάρκεια, η δυσκολία στη συγκεκριμένη υλοποίηση ήταν ότι αρχικά οι διάρκειες των παραστάσεων από τη βάση ήταν στη μορφή αλφαριθμητικού(string) και ήταν της μορφής για παράδειγμα (1:30), επομένως έπρεπε να γίνει διαχωρισμός της τιμής με βάση την άνω κάτω τελεία ωστόσο δεν έφτανε μόνο αυτό διότι σε κάποιες στιγμές υπήρχαν και διάφοροι άλλοι χαρακτήρες όπως γράμματα είτε τόνοι τα οποία έπρεπε να προβλεφθούν ώστε η διάρκεια να πάρει μία μορφή ώρας και λεπτών. Αφότου έγιναν όλοι οι απαραίτητοι έλεγχοι οι δύο πλέον τιμές ώρα και λεπτά μετατράπηκαν σε αριθμητικούς χαρακτήρες με την εντολή 'parseDouble(τιμή που επιθυμούμε να μετατραπεί)' και με τις κατάλληλες πράξεις μετατράπηκαν σε λεπτά διότι όπως αναφέρθηκε η διάρκεια έχει τιμή λεπτών για να γίνει η σύγκριση και να εμφανιστούν οι παραστάσεις που έχουν διάρκεια μέχρι την τιμή που έχει δώσει ο χρήστης. Στο φίλτρο των πρόσφατων παραστάσεων τα πράγματα είναι πιο απλά διότι υπάρχει έτοιμο http request οπότε χρειάζεται η αλλαγή του url που έφερνε τα Json δεδομένα ώστε να αλλάξουν τα στοιχεία του recyclerview. Παρατίθεται ο κώδικας της μεθόδου η οποία μετατρέπει τη διάρκεια των παραστάσεων σε έγκυρη χρονική διάρκεια.

```
1 public double FormatPrice(String priceRange){
2         priceRange = priceRange.replaceAll("από", "");
3         String[] newPrice =  priceRange.split("-");
4         double price;
5         if(newPrice.length == 1)
6         {
7             StringBuffer sb = new StringBuffer(priceRange);
8             sb.deleteCharAt(sb.length()-1);
9             String[] p =  String.valueOf(sb).split(",");
10            price = Double.parseDouble(p[0]);
11        }
12        else
13        {
14            StringBuffer sb = new StringBuffer(newPrice[1]);
15            sb.deleteCharAt(sb.length()-1);
16            String[] p =  String.valueOf(sb).split(",");
17            price = Double.parseDouble(p[0]);
18
19
20        }
21        return price;
22    }
```

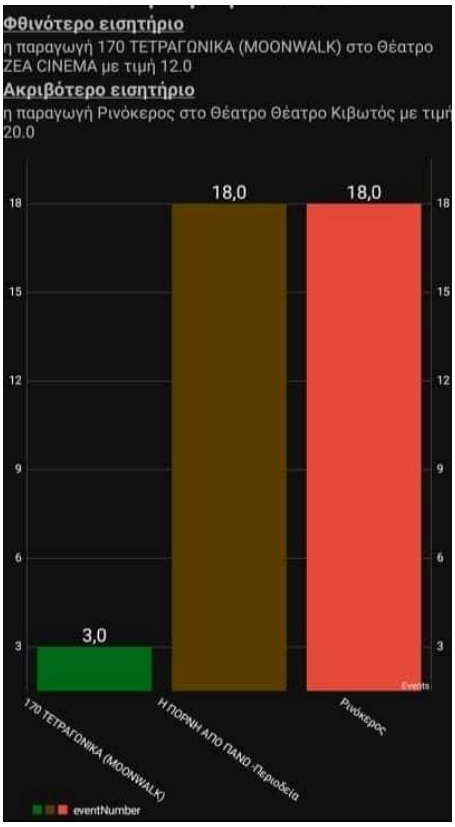
Φεύγοντας από τα φίλτρα ακριβώς πιο κάτω ο χρήστης θα παρατηρήσει ένα ακόμα κουμπί το οποίο ονομάζεται 'Compare' και εδώ έχουν το ρόλο τους και τα checkboxes που αναφέρθηκαν παραπάνω ότι υπάρχουν δεξιά του τίτλο σε κάθε παράσταση στη λίστα. Εδώ υπάρχει η δυνατότητα σύγκρισης

των παραστάσεων ουσιαστικά χρειάζεται να επιλεχθούν τα checkbox όσων παραστάσεων επιθυμεί ο χρήστης και να πατηθεί το κουμπί ‘Compare’.



Εικόνα 5.3: Οθόνη παραστάσεων με την εκτεταμένη οθόνη ανοιχτή

Αμέσως μετά οδηγείται σε νέο Activity το οποίο συγκρίνει τις παραστάσεις. Στην αρχή της οθόνης εμφανίζονται οι παραστάσεις με το ακριβότερο και το φθηνότερο εισιτήριο. Για να είναι σωστό το αποτέλεσμα αυτής της σύγκρισης χρειάστηκε να γίνουν κάποιες προεργασίες διότι οι τιμές των εισιτηρίων από τη βάση ήταν σε αλφαριθμητική μορφή και όχι μεμονωμένες τιμές σε κάποια στοιχεία υπάρχουν τιμές της μορφής: από 12 εώς 13, σε κάποια άλλα αντί για το λεκτικό κομμάτι υπάρχει ‘-’, σε κάποια άλλα απλά μία τιμή και σε κάποια άλλα πολλαπλά κενά(spaces) οπότε έπρεπε να γίνει η κατάλληλη μορφοποίηση των τιμών και να ληφθεί υπόψη η ακριβότερη τιμή. Αφού έγινε η μορφοποίηση των τιμών και η μετατροπή τους σε αριθμούς έγινε η σύγκριση και το αποτέλεσμα παρουσιάζεται σε αυτή την οθόνη. Η σύγκριση όμως των παραστάσεων συνεχίζεται και έτσι στην υπόλοιπη οθόνη υπάρχει ραβδόγραμμα το οποίο αναπαριστά τις διάφορες παραστάσεις στον άξονα των χ και στον άξονα των y είναι ο αριθμός των διοργανώσεων που έχει η κάθε παράσταση, ουσιαστικά φαίνεται ποιες παραστάσεις έχουν γίνει περισσότερες φορές και πόσες φορές. Η υλοποίηση του ραβδογράμματος έγινε με τη χρήση της βιβλιοθήκης MP Android Chart, με την οποία για να δημιουργηθεί το διάγραμμα έπρεπε να περάσουν τα στοιχεία και να δοθεί μορφή σχεδίαση και χρώματα προγραμματιστικά. Έτσι με τα στοιχεία που περάστηκαν από τις παραστάσεις που επέλεξε ο χρήστης μέσω intent έγιναν πολλαπλά http requests από κάθε παράσταση για τις διοργανώσεις οι οποίες και μετρήθηκαν για κάθε παράσταση και ο αριθμός αυτών περάστηκε στο διάγραμμα ώστε να γίνει η εμφάνισή του. Έτσι με μια γρήγορη ματιά φαίνεται αν κάποια παράσταση είχε επιτυχία κατά τη γνώμη του χρήστη είτε όχι σε σύγκριση με τις υπόλοιπες που τον ενδιαφέρουν.



Εικόνα 5.4: Ραβδόγραμμα σύγκρισης παραστάσεων

Σημαντική λειτουργία στην οθόνη αυτή είναι η αναζήτηση των παραστάσεων η οποία ακολουθεί τη διαδικασία που υπάρχει στην αναζήτηση των ηθοποιών με τη διαφορά ότι η αναζήτηση γίνεται πλέον στους τίτλους των παραστάσεων. Η διαφορά στη δυσκολία σε αυτή την οθόνη είναι ότι χρειαζόταν να διατηρηθεί η επιλογή του χρήστη για σύγκριση παραστάσεων που έχει γίνει από προηγούμενη αναζήτηση αφού όπως έχει ειπωθεί οι οθόνες δημιουργούνται δυναμικά. Όπως και έγινε με τη δημιουργία μια ξεχωριστής λίστας με παραστάσεις στον αντίστοιχο Adapter.

Επιπλέον δεν μπορεί να μην σημειωθεί ότι μπορεί να γίνει κλικ σε κάθε στοιχείο στη λίστα παραστάσεων και να ανοίξει η οθόνη-Activity με πληροφορίες της παράστασης που επιλέχθηκε η οποία είναι η ίδια με την οθόνη παραστάσεων που εμφανίζεται όταν επιλεχθεί η παράσταση στην οθόνη του ηθοποιού(κεφάλαιο 5.1).

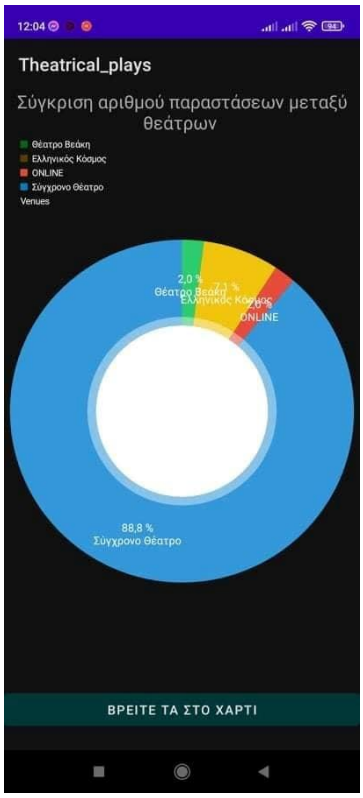
6.6 Οθόνη Θεάτρων

Η Τρίτη και τελευταία κεντρική οθόνη της εφαρμογής είναι αυτή των θεάτρων. Εμφανίζεται είτε με το πάτημα του πρώτου εικονιδίου στο μενού είτε όπως και για τις υπόλοιπες οθόνες με το κουμπί περισσότερα στην αρχική οθόνη στο μέρος που αφορά τα θέατρα. Η οθόνη είναι σε recyclerView μία λίστα από θέατρα και στο πάνω μέρος της σελίδας ένα κουμπί ‘Compare’ όπως την οθόνη παραστάσεων. Οι πληροφορίες για το κάθε θέατρο είναι ο τίτλος του και η διεύθυνσή του. Στη διεύθυνση του κάθε θεάτρου έχει στην αρχή ένα δείκτη που θυμίζει αυτό στους διαδικτυακούς χάρτες είτε αυτό είναι google maps είτε κάποια άλλης εφαρμογής χαρτών όταν επιλέγεται τοποθεσία-στόχος, ενώ στην δεξιά μεριά υπάρχει ένα checkbox όπως αυτό στην οθόνη των παραστάσεων.

Με το πάτημα του εικονιδίου δείκτη περνιέται στον κώδικα ο τίτλος της διεύθυνσης και με τη χρήση της βιβλιοθήκης Geocoder και της μεθόδου getFromLocationName(όνομα διεύθυνσης) που υπάρχει στη συγκεκριμένη βιβλιοθήκη βρίσκονται οι συντεταγμένες της διεύθυνσης. Αμέσως ανοίγει στο χρήστη ένα pop up παράθυρο με τίτλο 'Lanch Maps' και δίνει στο χρήστη μία λίστα από επιλογές εφαρμογών χαρτών που χρησιμοποιεί έχει ο χρήστης στο κινητό του, όπως google maps και moovit. Με το άνοιγμα μιας από αυτές τις εφαρμογές 'μαρκάρετε' με το δείκτη η τοποθεσία του θεάτρου. Η συγκεκριμένη λειτουργία προκλήθηκε αφού περάστηκαν με Intent οι συντεταγμένες της διεύθυνσης οι οποίες μετατράπηκαν κατάλληλα με τη βοήθεια της βιβλιοθήκης Uri για να έχουν την κατάλληλη μορφή για να περάσουν οι συντεταγμένες στην εφαρμογή που επιλέχθηκε. Για την εμφάνιση της επιλογής ευθύνεται η εντολή του αντικειμένου intent το createChooser στην οποία περνάνε ως παράμετροι το αντικείμενο intent που έχει και τις συντεταγμένες και ο τίτλος του παραθύρου στην προκειμένη περίπτωση 'Lanch Maps'.

```
menuItemHolder.button.setOnClickListener(view -> {
            String address = menuItemHolder.title.getText().toString();
            if (address != "ONLINE")
            {
                List<Address> addressVenue = null;
                Geocoder geocoder = new Geocoder(mContext);
                try {
                    addressVenue = geocoder.getFromLocationName(address,
2);
                }
                catch (IOException e)
                {
                    e.printStackTrace();
                }
                Intent intent = new Intent(Intent.ACTION_VIEW);
                intent.setData(Uri.parse("geo:0,0?q="+
Uri.encode(address)));
                Intent chooser = Intent.createChooser(intent, "Launch
Maps");
                mContext.startActivity(chooser);
            }
```

Η επόμενη δυνατότητα της συγκεκριμένης οθόνης είναι η σύγκριση θεάτρων. Ο χρήστης επιλέγει τα θέατρα που επιθυμεί να συγκρίνει πατώντας το checkbox που υπάρχει σε κάθε στοιχείο της λίστας και αφού επιλέξει αυτά που επιθυμεί πατάει το κουμπί 'Compare' και οδηγείται σε μια άλλη οθόνη-Activity στο οποίο υπάρχουν ένα διάγραμμα πίτας το οποίο απεικονίζει με ποσοστά το πλήθος των παραστάσεων που έχει το κάθε θέατρο σε συνάρτηση με τα υπόλοιπα, έτσι φαίνεται πιο είχε τις περισσότερες παραστάσεις. Όπως και στο διάγραμμα της σύγκρισης παραστάσεων έτσι και εδώ χρησιμοποιήθηκε η βιβλιοθήκη MP Android Chart. Τα στοιχεία προήλθαν από των αριθμό των παραστάσεων που έχει το κάθε θέατρο τα οποία μετατράπηκαν σε ποσοστά με βάση τον συνολικό αριθμό από όλα τα θέατρα που επιλέχθηκαν. Στο κάτω μέρος αυτής της οθόνης υπάρχει ένα κουμπί βρείτε στο χάρτη όπου με την ίδια λογική της εύρεσης τοποθεσίας ενός συγκεκριμένου θεάτρου, με το πάτημα του κουμπιού εμφανίζεται το ίδιο pop up επιλογής εφαρμογής χαρτών με τη διαφορά ότι τώρα θα εμφανιστούν οι διευθύνσεις όλων των θεάτρων που επιλέχθηκαν ώστε να μπορεί να δει ο χρήστης και ποια από αυτά βρίσκονται κοντά του είτε την απόσταση απλά μεταξύ τους.



Εικόνα 5.5: *Πίτα Σύγκρισης θεάτρων*

Δεν θα μπορούσε να λείπει σε αυτή την οθόνη η αναζήτηση των θεάτρων. Σε αντίθεση όμως με τις άλλες δύο οθόνες δεν έχει υλοποιηθεί με http request αλλά με φιλτράρισμα του recyclerView των θεάτρων με την μέθοδο filter η οποία σύμφωνα με τη λέξη που αναζητά ο χρήστης φιλτράρει τον τίτλο του θεάτρου και συγκρίνει τα string μεταξύ τους και σύμφωνα με αυτή τη σύγκριση επιλέγεται αν μπορεί ένα στοιχείο στη λίστα να παραμένει ορατό ή όχι.

Εν κατακλείδι όπως σε όλες τις υπόλοιπες οθόνες με την επιλογή ενός στοιχείου της λίστας ανοίγει οθόνη-Activity με επιπλέον πληροφορίες για το συγκεκριμένο στοιχείο ανάλυση αυτής της οθόνης υπάρχει στο υποκεφάλαιο 5.8

6.7 Οθόνη πληροφορίες παράστασης

Η οθόνη των πληροφοριών της κάθε παράστασης είναι ένα activity το οποίο παίρνει τις πληροφορίες του με intent από τις οθόνες που το καλούν οι οποίες έχουν τις πληροφορίες αυτές από το Json αρχείο που έχουν καλέσει για όλες τις παραστάσεις που αφορούν ξεχωριστές οθόνες. Οι πληροφορίες που προσφέρει για κάθε παράσταση είναι ο τίτλος, φωτογραφία, trailer , περιγραφή και παραστάσεις.

Η φωτογραφία της κάθε παράστασης έρχεται σε μορφή διαδικτυακού url. Η εμφάνιση γίνεται στο xml αρχείο με ένα imageView το οποίο παίρνει μορφή με τη βοήθεια της βιβλιοθήκης Picasso η οποία έχει την ικανότητα να εμφανίζει τις φωτογραφίες από το url τους και αν δεν το βρει να τοποθετεί μία προκαθορισμένη εικόνα.

Κάτω από τη φωτογραφία υπάρχει ένα κουμπί με τον τίτλο ‘Trailer’ το οποίο αν πατηθεί οδηγεί τον χρήστη είτε στο περιβάλλον ενός προγράμματος περιήγησης ιστού είτε στο youtube ανάλογα με το url που υπάρχει στη βάση ώστε να δει ο χρήστης το απόσπασμα της παράστασης (trailer).

Η περιγραφή της παράστασης είναι ένα κείμενο textView το οποίο ο χρήστης μπορεί να κάνει scroll down και έχει υποστεί μορφοποίηση. Διότι από το json αρχείο υπήρχαν αρκετά περιττά στοιχεία στο

κείμενο όπως μεγάλα κενά και ξένοι χαρακτήρες με αποτέλεσμα να χαλάνε την εικόνα του κειμένου χρειάστηκε μορφοποίηση. Έτσι έγιναν αντικατάσταση κάποιον χαρακτήρων και πολλαπλών κενών με τη μέθοδο replace στο String.

Τέλος μετά από αυτές τις πληροφορίες υπάρχει μία λίστα από τις διοργανώσεις που έγιναν σε αυτή την παράσταση. Είναι ένα recyclerView από διοργανώσεις με πληροφορίες για το θέατρο που έγινε η παράσταση, την τιμή του εισιτηρίου και τη μέρα της παράστασης για κάθε οργάνωση και την πληρότητα των θέσεων σε κάθε μία από αυτές.

6.8 Οθόνη πληροφορίες θεάτρου

Στην οθόνη-activity για τις πληροφορίες του θεάτρου παρουσιάζονται ο τίτλος του θεάτρου και η εικόνα του στην αρχή της οθόνης. Υπάρχει ένα ποσοστό σε σχήμα πίτας που φαίνεται κατά μέσο όρο την πληρότητα του θεάτρου για όλες τις παραστάσεις, ώστε να είναι γνωστή και η επισκεψημότητά του. Επιπλέον δεν μπορεί να λείπει μία λίστα με παραστάσεις που έχουν γίνει σε αυτό το θέατρο οι οποίες μπορούν να επιλεχθούν και να οδηγήσουν το χρήστη στην εκάστοτε σελίδα της παράστασης. Η λίστα όπως όλες οι άλλες στην εφαρμογή είναι ένα recyclerView.

7 Κεφάλαιο : Υλοποίηση

7.1 Συστατικά εφαρμογής

7.1.1 Activities

Όταν λέμε activity εννοούμε μια οθόνη, ένα παράθυρο στο οποίο σχεδιάζεται UI εφαρμοφής και είναι το ορατό αποτέλεσμα στο χρήστη. Όταν δημιουργείται μία εφαρμογή δημιουργείται αυτόματα ένα main activity πάνω στο οποίο χτίζεται, είναι η κύρια οθόνη. Τα activities όμως χρησιμοποιούνται και ως απλές οθόνες όπως για παράδειγμα στην εφαρμογή ανάκτησης δεδομένων για το θέατρο activities είναι όλες οι οθόνες για μεμονωμένες πληροφορίες στοιχείων όπως αυτή του ενός ηθοποιού, παράστασης ακόμα και τα διαγράμματα. Πάνω στο main activity έχει γίνει το bottom navigation της εφαρμογής και γίνονται οι εναλλαγές των οθονών.

7.1.2 Fragments

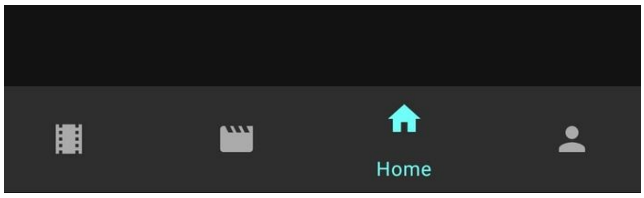
Όπως τα activities έτσι και και τα fragments είναι οι οθόνες τις εφαρμογής. Για να γίνει αντιλυπτή η διαφορά τους το fragment είναι sub-activity δηλαφή εξαρτάται άμεσα από την υπάρξη ενός activity. Ωστόσο έχει τη δική του μορφοποίηση και εμφάνιση και τις δικές τους λειτουργίες. Μπορούν να γίνουν εναλλαγές πολλών fragments μέσα στο ίδιο activity όπως γίνεται και στη περίπτωση του navigation της εφαρμογής κεφάλαιο 6.1.4.

7.1.3 RecyclerView

Με την προσθήκη του depency στο grandel implementation 'androidx.recyclerview:recyclerview:1.2.1' μπορεί να χρησιμοποιηθεί το recyclerview. Είναι μία λίστα για να παρουσιαστούν πολλά δεδομένα τα οποία να είναι scrollable στην οθόνη. Με αυτό παρουσιάζονται όλες οι λίστες που υπάρχουν στην εφαρμογή. Μπορεί ο χρήστης με ένα απλό xml layout να δώσει τη μορφή που θέλει να εμφανίζεται το ένα στοιχεία στη λίστα και όλα τα υπόλοιπα να ακολουθούν αυτή τη μορφοποίηση καθώς και ολόκληρη η λίστα να έχει την κατεύθυνση που θέλει ο χρήστης είτε κάθετα είτε οριζόντια. Για παράδειγμα στην οθόνη των ηθοποιών ο κάθε ηθοποιός στη λίστα έχει όνομα και φωτογραφία, που ονομάζεται item, η μορφοποίηση αυτή υπάρχει από ξεχωριστώ xml. Τα στοιχεία γεμίζουν με adapters οι οποίοι θα αναλυθούν σε επόμενο κεφάλαιο. Αν και υπάρχει και άλλος τρόπος εμφάνισης δεδομένων σε λίστα το ListView επιλεχθεί το recyclerView διότι, μπορεί να έχει το ίδιο scrollable αποτέλεσμα και πιο εύκολη διαδικασία γεμίσματος στοιχείων η μορφή είναι σταθερά κάθετη και δεν υπάρχει δυνατότητα αλλαγής στην εμφάνιση των στοιχείων.

7.1.4 Navigation

Όπως αναφέρθηκε στη εφαρμογή έχει χρησιμοποιηθεί η λειτουργία navigation με fragments. Για την υλοποίηση έγινε implement το εξής dependency 'androidx.navigation:navigation-ui:2.4.2'. Στη συγκεκριμένη περίπτωση είναι ένα μενού στο κάτω μέρος της οθόνης και βοηθάει τους χρήστες να ξεναγηθούν στην εφαρμογή. Τα icons που φαίνονται στην παρακάτω εικόνα είναι drawbles δηλαδή εικονίδια που έχουν επιλεχθεί μέσα από το android studio. Το κάθε ένα από αυτά αντιπροσωπεύει ένα item σε ένα xml αρχείο που ονομάζεται menu το οποίο του δίνει μορφή, τίτλο και id .



Εικόνα 6.1: Bottom Navigation Menu

Οι εναλλαγές οθονών του bottom Navigation γίνεται με κώδικα που υπάρχει στο main activity της εφαρμογής. Στο παρακάτω απόσπασμα κώδικα φαίνονται οι επιλογές σε cases κάθε id είναι ένα εικονίδιο που θα γίνει κλικ από το χρήστη και θα τον οδηγήσει στο αντίστοιχο fragment. Ουσιαστικά με κάθε πάτημα από το navigation πυροδοτείτε η μέθοδος beginTransaction() η οποία κάνει replace το fragment που εμφανίζεται στο χρήστη με αυτό που έχει οριστεί από τον κώδικα. Η διαδικασία αυτή είναι η γραμμή 23 του κώδικα.

```
  public BottomNavigationView.OnNavigationItemSelectedListener
1 bottomNavMethod= item -> {
2
3         Fragment fragment = null;
4
5         switch (item.getItemId())
6         {
7             case R.id.movie:
8                 fragment = new MovieFragment();
9                 break;
10            case R.id.theater:
11                fragment = new TheaterFragment();
12                break;
13            case R.id.home:
14                fragment = new HomeFragment();
15                break;
16            case R.id.actors:
17                fragment = new ActorFragment();
18                break;
19        }
20
21
22 getSupportFragmentManager().beginTransaction().replace(R.id.container,
23 fragment). commit();
24
          return true;
      };
```

7.2 Μορφή κώδικα

Σε όλη τον Java κώδικα της εφαρμογής ακολουθείται μια λογική η οποία σε κλάσεις και διαχωρισμό σε packages. Κάθε package έχει το όνομα μία από τις οθόνες της εφαρμογής. Στο κάθε package υπάρχουν οι κλάσεις για τα activity ή το fragment αντίστοιχα, adapters για το recyclerView και η κλάσεις με ονόματα των βασικών στοιχείων της εφαρμογής (π.χ. ηθοποιοί, παραστάσεις)με τις μεταβλητές που έχει ο κάθε ρόλος σύμφωνα με τη βάση και τα στοιχεία που φτάνουν για αυτό από το API.

7.2.1 Java class Fragment and activity

Για κάθε fragment και actitvity που παριστάνει μια οθόνη στην εφαρμογή υπάρχει και μια αντίστοιχη Java , Οι συγκεκριμένες κλάσεις δημιουργούνται αυτόματα από το android studio μετά από κάθε προσθήκη framgment είτε activity μαζί με τα xml αρχεία τους. Με την αυτόματη δημιουργία τους υπάρχει η μέθοδος onCreate η οποία είναι κενή αν το επιθυμήσει ο προγραμματιστής στην προεπιλογή, και μέσα σε αυτής γίνονται όλες που εργασίες που κρίνονται απαραίτητες κατά τη δημιουργία της οθόνης, δηλαδή με την εμφάνισή της στον χρήστη. Σε αυτή διαχειρίζονται όλα τα items που υπάρχει στο αντίστοιχο xml και θέλει ο προγραμματιστείς να εμφανίσει. Όπως είναι εμφάνιση εικόνων, αναζήτηση, γέμισμα recyclerView και όλα τα events που προκαλούνται με τα πατήματα των διάφορων κουμπιών καθώς και μέσα σε αυτές γίνονται τα get από τα http requests για την φόρτωση των δεδομένων.

Παράδειγμα της onCreate είναι στο παρακάτω απόσπασμα κώδικα όπου φαίνεται η δημιουργία της final μεταβλητής rootview στην οποία περνάει το αντίστοιχο xml με την μέθοδο Inflate. Η inflate αφού έχουμε δώσει ως παράμετρο το id του xml αρχείου στην προκειμένη περίπτωση fragment_movie, μετατρέπει το αρχείο σε object της Java και το περνάει στην μεταβλητή rootView. Η μεταβλητή αυτή χρησιμοποιείται όατν είναι επιθυμητό να επικαλεστούμε ένα συγκεκριμένο item του xml. Όπως φαίνεται στην γραμμή 10 του κώδικα με τη rootView και τη μέθοδο findViewById που δέχεται ως παράμετρο το id του item του αρχείου xml πλέον έχουμε το java Object του item, στο οποίο όπως πρόκειται στο παράδειγμα που το item είναι ένα button(κουμπί) να προστεθεί ένα Event όπως το Onclick ώστε να γίνουν κάποιες ενέργειες όπως φαίνεται η εξαφάνιση και η εμφάνιση ενός View.

Επιπλέον σημαντική υλοποίηση στα fragment και actitvity έχει αυτή της κλάσης R. Η κλάση R δημιουργείται αυτόματα από τους πόρους της εφαρμογής. Περιέχει τα id για τα item ή widgets των xml αρχείων πόρους. Περιέχεται στο πακέτο στην ετικέτα <manifest> στο αντίστοιχο AndroidManifest. xml αρχείο. Χρησιμοποιείται όταν επιθυμούμε να επικαλούμαστε ένα συγκεκριμένο item στον Java κώδικα.

```
1    public View onCreateView(LayoutInflater inflater, ViewGroup container,
 2 Bundle savedInstanceState) {
 3         final  View rootView =  inflater.inflate(R.layout.fragment_movie,
 4 container, false);
 5         rv.setLayoutManager(new GridLayoutManager(getContext(),1));
 6         QUEUE = Volley.newRequestQueue(getContext());
 7         URLHTTP = getResources().getString(R.string.productions);
 8         httpGET(URLHTTP);
 9         setHasOptionsMenu(true);
10         expandedView = rootView.findViewById(R.id.expandedView);
11         filters = rootView.findViewById(R.id.filtersButton);
12         home = rootView.findViewById(R.id.home);
13  filters.setOnClickListener(new View.OnClickListener() {
14             @Override
15             public void onClick(View v) {
16                 if (expandedView.getVisibility() == v.GONE)
17                 {
18                     TransitionManager.beginDelayedTransition(home, new
19 AutoTransition());
20                     expandedView.setVisibility(v.VISIBLE);
21                 }else
22                 {
23                     TransitionManager.beginDelayedTransition(home, new
```

```
24 AutoTransition());
                        expandedView.setVisibility(v.GONE);
                    }
                }
            });
```

7.2.2 Κλάσεις βασικών ρόλων

Έχουν δημιουργηθεί κλάσεις για κάθε οντότητα της εφαρμογής που εμφανίζονται στοιχεία και πληροφορίες για αυτά Events για τις διοργανώσεις Actor για ηθοποιούς και λοιπά. Κάθε μία από αυτές τις κλάσεις έχει τις απαραίτητες μεταβλητές δηλαδή πληροφορίες για κάθε οντότητα σύμφωνα με τις πληροφορίες που δίνει η βάση για το καθένα. Υπάρχει ένας constructor και μέθοδοι set και get σε κάθε μεταβλητή. Παρακάτω παρουσιάζεται ως παράδειγμα η κλάση για τους ηθοποιούς.

```
1 public class Actor {
2
3     String name;
4     int id;
5     String imageUrl;
6
7
8     public Actor( String name, String imageUrl, int id) {
9         super();
10        this.name = name;
11        this.imageUrl = imageUrl;
12        this.id = id;
13    }
14
15    public Actor() {
16    }
17
18
19    public String getName() {
20        return name;
21    }
22
23    public void setName(String name) {
24        this.name = name;
25    }
26
27    public String getImageUrl() {
28        return imageUrl;
29    }
30
31    public void setImageUrl(String imageUrl) {
32        this.imageUrl = imageUrl;
33    }
34
35    public int getId() {
36        return id;
37    }
38
39    public void setId(int id) {
40        this.id = id;
```

```
41      }
42 }
```

7.2.3 Adapters

Σε αυτό το κεφάλαιο θα αναλυθεί ένας σετ κώδικα που ονομάζονται adapters και συνδέονται με το recyclerView που αναλύεται στο κεφάλαιο 6,1,3. Είναι μία κλάση η οποία πάει ζευγάρι με κάθε recyclerView που υπάρχει στην εφαρμογή. Όπως αναφέρθηκε το recyclerView σε μια πιο ελεύθερη επεξήγηση είναι μία λίστα από το ίδιο xml layout αρχείο. Εκεί έρχεται η δουλειά του adapter, κληρονομεί την abstract κλάση RecyclerView.Adapter με τύπο παραμέτρου ViewHolder. ViewHolder είναι ένα αντικείμενο προβολής και metadata το οποίο βρίσκεται σε ένα RecyclerView, όπως για παράδειγμα ένα textView.

Η κλήση του adapter γίνεται από την κλάση fragment ή activity ανάλογα που επιθυμεί ο προγραμματιστής να εμφανιστεί το recyclerView. Το βασικό αντικείμενο που διαθέτει στον δομητή του (contractor) είναι μία λίστα αντικειμένων της κλάσης που μας ενδιαφέρει. Στην περίπτωση της εφαρμογής θεατρικών παραστάσεων αναφερόμαστε σε μία από τις κλάσεις των βασικών ρόλων (κεφ. 6,2,1). Η λίστα αυτή παίρνει τιμές στην αντιστοιχεί κλάση fragment ή activity όπου δημιουργείται και αντικείμενο της κλάσης Adapter.

Εκτός από το πως καλείται σημαντική είναι και η λειτουργία του adapter και τι περιέχεται σε αυτό. Αρχικά δημιουργήθηκε μέσα στο αρχείο μία private κλάση MenuItemViewHolder η οποία κληρονομεί την RecyclerView.ViewHolder, ο ρόλος της είναι η αντιστοίχηση των id των items στο xml αρχείο που έχει φτιαχτεί για την εμφάνιση του κάθε στοιχείου της recyclerView λίστας, με μεταβλητές στη Java με την findViewById όπως αναφέρθηκε στο κεφάλαιο 6,2,1. Γυρνόντας στο εσωτερικό του adapter είναι οι μέθοδοι που αποτελούν υλοποιήσεις της RecyclerView.Adapter. Αρχικά είναι η onCreateViewHolder η οποία είναι παρόμοια δημιουργεί Java αντικείμενο του xml αρχείου του στοιχείου στη λίστα με το id του και επιστρέφει ένα καινούριο αντικείμενο της κλάσης MenuItemViewHolder με παράμετρο αυτό το Java object. Παρακάτω είναι η onBindViewHolder, όπως φαίνεται και από το όνομά της κάνει αντιστοίχιση δηλαδή για κάθε θέση στη λίστα αντικειμένων στον δομητή αντιστοιχίζει την τιμή του με το xml item. Για παράδειγμα επιθυμείτε η εμφάνιση του τίτλου στο TextView με id ίσο με title, ο κώδικας δημιουργεί αντικείμενο για την MenuItemViewHolder ώστε να πάρει το σωστό item και δημιουργεί άλλο ένα αλλά από μία κλάση βασικών ρόλων με το αντικείμενο της λίστα που βρίσκεται σε μία συγκεκριμένη θέση. Τέλος με την setText πλέον το xml item έχει τον τίτλο που υπάρχει στη λίστα. Η τελευταία μέθοδος έχει και events όπως είναι onClick και ότι άλλο μπορεί να έχει η OnCreate στην περίπτωση της κλάσης Fragment και εκτελείται για κάθε θέση στη λίστα αντικειμένων του adapter. Παράδειγμα αυτής είναι ο παρακάτω κώδικας.

```
1  public void onBindViewHolder(RecyclerView.ViewHolder holder, int
   position) {
2  MenuItemViewHolder menuItemHolder = (MenuItemViewHolder) holder;
3  final Actor actor = (Actor) recyclerViewItems.get(position);
4  menuItemHolder.name.setText(actor.getName());
5  String url = actor.getImageUrl();
6  if(!(url.equals(""))) {
7  Picasso.get().load(url).placeholder(R.drawable.ic_actor).into(menuIte
   mHolder.imageUrl);
8  }
9  else {
10    menuItemHolder.imageUrl.setImageResource(R.drawable.actor1);
```

```
11 }

12 }
```

7.2.4 Αρχείο Manifest

Στο κεφάλαιο 2.3 που αφορά το στήσιμο μιας εφαρμογής με android studio αναφέρθηκε η χρησιμότητα του αρχείου AndroidManifest το οποίο είναι γραμμένο σε xml. Δημιουργείται αυτόματα όταν φτιάχνεται καινούριο project. Μέσα σε αυτό είναι όλα τα δικαιώματα που χρειάζεται να έχει πρόσβαση η εφαρμογή ώστε να λειτουργήσει ομαλά χωρίς προβλήματα. Ένα σημαντικό που χρειάζεται η εφαρμογή ανάκτησης δεδομένων για το θέατρο κυρίως για την φόρτωση των δεδομένων είναι η πρόσβαση στο διαδίκτυο. Το tag που χρειάζεται ο προγραμματιστής σε αυτή την περίπτωση είναι το <uses-permission> το οποίο ακολουθείται από το όνομα της προσβάσιμης πληροφορίες για παράδειγμα android:name="android.permission.INTERNET" που είναι για το διαδίκτυο της συσκευές που θα τρέξει η εφαρμογή.

Επιπλέον σε αυτό το αρχείο έχει ένα tag <application> στο οποίο έχει κάποιες πληροφορίες για την εφαρμογή. Όλα τα ονόματα από τα activity που διαθέτει και τα δικαιώματά τους, δηλαδή αν μπορούν να χρησιμοποιηθούν από άλλες εφαρμογές. Ενώ άλλες πληροφορίες όπως ο τίτλος της εφαρμογής που εμφανίζεται στο πάνω μέρος της εφαρμογής και κάποιες μορφοποιήσεις όπως κυκλική εμφάνιση κάποιον φωτογραφιών δηλώνονται σε αυτό το αρχείο με το κατάλληλο tag.

7.3 Βιβλιοθήκες που χρησιμοποιήθηκαν

Για την υλοποίηση κάποιον αιτημάτων υλοποιήθηκαν βιβλιοθήκες οι οποίες έγιναν implement στο grandle. Κάποιες ήταν είτε για τρόπο εμφάνιση είτε χρειάστηκαν να υλοποιηθούν κλάσεις τους στον κώδικα για το τελικό αποτέλεσμα.

7.3.1 Picasso

Picasso είναι βιβλιοθήκη για εμφάνιση φωτογραφιών στο grandle είναι ως com.squareup.picasso:picasso:2.71828 όπου οι αριθμοί στο τέλος αποτελούν την έκδοση της. Χρησιμοποιείται για να φορτώνονται εικόνες και να εμφανίζονται μέσω url είτε του διαδικτύου είτε σε έναν φάκελο, σε ένα imageView, το οποίο είναι το item που χρησιμοποιείται στα xml layouts για εικόνες. Παράδειγμα υλοποίησης αυτής της βιβλιοθήκης είναι το κομμάτι κώδικα στην εικόνα 6.2.

```
if(!(url.equals(""))) {
    Picasso.get().load(url).placeholder(R.drawable.ic_actor).into(menuItemHolder.imageUrl);
}
```

Εικόνα 6.2: Picasso implementation

Με το get() είναι η εντολή που ‘ζητάμε’ την εικόνα. Συνεχίζοντας στο load(url) όπου url είναι η διεύθυνση σε μορφή string της φωτογραφίας, το οποίο δεν επιτρέπεται να είναι κενό αλλιώς βγαίνει σφάλμα. Σε περίπτωση που δεν βρεθεί η διεύθυνση υπάρχει έρχεται το placeholder() το οποίο δέχεται ως παράμετρο μια οποιαδήποτε εικόνα έχει αποθηκεύσει ο προγραμματιστείς ως default για αυτό που επιθυμεί να εμφανίζει. Στο τέλος ακολουθεί το into(), με παράμετρο το id του imageView ώστε να τοποθετηθεί και να εμφανιστεί η εικόνα. Με αυτή τη βιβλιοθήκη έγινε η εμφάνιση όλων των φωτογραφιών στην εφαρμογή.

7.3.2 Volley vs Retrofit

Όπως έχει αναφερθεί όλα τα στοιχεία της εφαρμογής θεατρικών παραστάσεων φτάνουν σε μορφή Json με url που περιέχει http request. Το android studio διαθέτει δύο γνωστές βιβλιοθήκες οι οποίες διαχειρίζονται τα http requests, ουσιαστικά χρησιμοποιούνται για τη μετάδοση δεδομένων μέσω του δικτύου κάνοντας τη δικτύωση πιο γρήγορη και ευκολότερη για τις εφαρμογές. Οι δύο αυτές βιβλιοθήκες είναι Retrofit και Volley. Στην εφαρμογή που αναπτύχθηκε χρησιμοποιήθηκε η Volley η απόφαση αυτή προήλθε από τη σύγκριση των δύο η οποία εξελίσσεται στις επόμενες παραγράφους.

Όσον αφορά τη βιβλιοθήκη retrofit, σύμφωνα με αρκετούς προγραμματιστές σε forums θεωρείται η πιο γρήγορη λύση, λόγω της εύκολης διαχείρισης της και την ανάγκη για συγγραφή λίγης ποσότητας κώδικα για την υλοποίησή της. Συνιστάται για μικρό αριθμό δεδομένων διότι δεν υποστηρίζει cached data, δηλαδή της πεοσωρινής αποθήκευσης των δεδομένων. Διαθέτει αυτόματη μετατροπή Json δεδομένων σε οπουδήποτε τύπο όπως Boolean, int , string και άλλα. Επιπλέον είναι συμβατή με okHTTP η οποία είναι υπεύθυνη για την αποστολή και λήψη αιτημάτων δικτύου που βασίζονται σε HTTP. Ωστόσο σε περίπτωση αποτυχία του url request, μόνο με παραπάνω κώδικα και τη λογική του προγραμματιστή γίνεται εφικτή η προσπάθεια επανασύνδεσης για επιτυχή αποτέλεσμα. Γενικότερα ως βιβλιοθήκη είναι πιο συντηρήσιμη και δέχεται συνεχώς αναβαθμίσεις.

Πηγαίνοντας στην βιβλιοθήκη volley της google. Αν και δεν υπάρχουν πολλές αναβαθμίσεις κατά καιρούς και ο κώδικας που χρειάζεται για την υλοποίησή της είναι πιο περίπλοκος όσος αφορά την retrofit έχει αρκετά θετικά τα οποία ταίριαζαν με την εφαρμογή που αναπτύχθηκε. Έχει καλύτερα αποτελέσματα σε πολλά δεδομένα από ένα request και σε αντίθεση με την παραπάνω έχει τη δυνατότητα να γίνονται cached τα δεδομένα που φτάνουν σε αυτήν. Με την μέθοδο setRetryPolicy προσαρμόζεται από τον προγραμματιστή το χρονικό περιθώριο για τον αριθμό των προσπαθειών που θα κρίνει απαραίτητους να γίνουν σε περίπτωση που το http request δεν είναι επιτυχής με την πρώτη προσπάθεια, καθώς και την αποχώρηση. Επιπρόσθετα, στη συγκεκριμένη βιβλιοθήκη τα δεδομένα φτάνουν στον Java κώδικα σε μορφή JSONObjects τα οποία μετατρέπονται προγραμματιστικά σε Java Objects με τη βοήθεια της βιβλιοθήκης Gson η οποία περιγράφεται στο κεφάλαιο 6.2.3. Κάτι το οποίο σε πολλούς θεωρείται αρνητικό φαινόμενο της συγκεκριμένης βιβλιοθήκης αλλά στης εφαρμογή ανάκτησης δεδομένων για το θέατρο ήταν θετικό διότι έδινε τη δυνατότητα μα γίνουν αμέσως κάποιες μετατροπές και να περαστούν στον adapter recyclerView όπως έπρεπε με τη σωστή μορφοποίηση. Εν κατακλείδι η εν λόγω βιβλιοθήκη μπορεί να ενσωματωθεί με πολλούς γνωστούς Http Clients και όχι μόνο με okHttp.

Μετά την αιτιολογία χρήσης δεν θα μπορούσε να παρελήφθη ο τρόπος υλοποίησης της βιβλιοθήκης Volley η οποία επιλέχθηκε. Όπως φαίνεται στην εικόνα 6.2 υπάρχουν δύο μεταβλητές την QUEUE και την URLHTTP. Η πρώτη είναι ένα RequestQueue που προσφέρει η volley η οποία είναι μία στοίβα που προθέτονται τα httpRequests που γίνονται και έχει την τιμή της μεθόδου newRequest που ως παράμετρο έχει το context, το πλαίσιο του Fragment των ηθοποιών σε αυτή την περίπτωση. Η δεύτερη

```
QUEUE = Volley.newRequestQueue(getContext());
URLHTTP = getResources().getString(R.string.urlserver);
httpGET(URLHTTP);
setHasOptionsMenu(true);
```

Εικόνα 6.2: το http Request

μεταβλητή είναι αυτή που παίρνει ως τιμή το url που μας ενδιαφέρει. Στην προκειμένη περίπτωση όλα τα url είναι αποθηκευμένα σε ένα xml αρχείο για αυτό το λόγω δεν υπάρχει string παραμόνο η αναφορά του. Μετά καλείται η httpGET η οποία βρίσκεται στη εικόνα 6.3 η οποία έχεις ως παράμετρο το url. Σε αυτή γίνεται το αίτημα http get με StringRequest, το οποίο αν είναι επιτυχής καλείται η parsingData το περιεχόμενο της οποίας είναι στην 6.4 και το αποτέλεσμα αυτής είναι τα δεδομένα. Στο τέλος προστίθεται το αποτέλεσμα στην στοίβα.

```
public void httpGET(String url)
{
    StringRequest stringRequest = new StringRequest(Request.Method.GET, url,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) { parsingData(response); }
            }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            try {
                String responseBody = new String(error.networkResponse.data, charsetName: "utf-8");
                Log.d(TAG, msg: "ERROR "+responseBody);
            }catch (UnsupportedEncodingException errorr){
                Log.d(TAG,errorr.toString());
            }
        }
    }
    );
    QUEUE.add(stringRequest);
}
```

Εικόνα 6.3: httpGET

7.3.3 Gson

Όπως αναφέρθηκε παραπάνω στην βιβλιοθήκη Volley τα δεδομένα έρχονται σε μορφή Json και χρειάζεται να μετατραπούν για να εμφανιστούν και να επεξεργαστούν, αυτό γίνεται με τη βοήθεια της Gson η οποία διαθέτει JSONObjects. Όπως στην εικόνα 6.4 δημιουργείται ένα JSONObject το οποίο περνάει τα data που έχει επιστρέψει το αίτημα http. Συνεχίζει και περνάει τα δεδομένα από το object data μέσα στο οποίο υπάρχει ένα άλλο object content στο οποίο υπάρχουν τα ζητούμενα δεδομένα. Αργότερα ακολουθεί μία δομή επανάληψης, όπου ‘περνάει’ όλα τα δεδομένα δημιουργώντας και άλλα JSONObjects όμως αυτή τη φορά σύμφωνα με τον τίτλο των αντικειμένων όπως για παράδειγμα το fullName τα οποία τα μετατρέπει στην κατάλληλη μορφή και δημιουργεί το αντίστοιχο αντικείμενο και το προσθέτει στο recyclerView για εμφάνιση.

```
JSONObject obj = new JSONObject(jsonData);
JSONObject m_obj = obj.getJSONObject("data");
JSONArray m_jArry = m_obj.getJSONArray( name: "content");

for (int i = 0; i < 100; i++) {
    JSONObject jo_inside = m_jArry.getJSONObject(i);
    String name = jo_inside.getString( name: "fullName");
    String imagUrl = jo_inside.getString( name: "image");
    int id = jo_inside.getInt( name: "id");

    Actor actor= new Actor(name,imagUrl, id);
    mRecyclerViewItems.add(actor);

}
```

Εικόνα 6.4: JSON objects

7.3.4 Intent

Έχει αναφερθεί αρκετές φορές μέχρι στιγμής, είναι μία λειτουργία που βρίσκεται στην βιβλιοθήκη androind.content. Χρησιμοποιείται για μεταφορά δεδομένων από μία οθόνη σε μία άλλη, για παράδειγμα από fragment σε activity ή σε άλλη εφαρμογή όπως google maps. Υπάρχουν δύο ήδη intent Explicit intent και το Implicit intent. Το πρώτο εφαρμόζεται όταν ο προγραμματιστής γνωρίζει που θα σταλθεί η πληροφορία και πυροδοτεί την εντολή καθώς ανοίγει τη σελίδα, activity ή εφαρμογή που απαιτείται. Η δεύτερη περίπτωση είναι δηλώνει τη γενική ενέργεια προς εκτέλεση, η οποία επιτρέπει τον χειρισμό ενός στοιχείου από άλλη εφαρμογή. Για παράδειγμα όταν γίνεται κοινή χρήσης σε οποιαδήποτε εφαρμογή με το πάτημα ενός κουμπιού, δίνεται η δυνατότητα να δει ο χρήστης τις επιλογές της εφαρμογής κοινής χρήσης Gmail, Bluetooth και άλλες. Εδώ ο χρήστης στέλνει ένα αίτημα είναι το σιωπηρό αίτημα που μπορεί να χειριστεί αυτή η εφαρμογή Gmail.

Στην εφαρμογή ανάκτησης δεδομένων για το θέατρο έχουν χρησιμοποιηθεί και οι δύο περιπτώσεις intent. Το explicit με το οποίο γίνονται όλες οι μεταφορές πληροφοριών από fragment σε activity. Για παράδειγμα όταν είναι ο χρήστης στην οθόνη των παραστάσεων και πατάει μία παράσταση από τη λίστα και μεταφέρεται στην ξεχωριστή οθόνη με τις πληροφορίες της παράστασης το οποίο είναι activity, κάποια από τα στοιχεία της παράστασης όπως είναι το κλειδί δηλαδή id της παράστασης μεταφέρονται από το fragment. Έτσι δημιουργείται ένα αντικείμενο τύπου intent το οποίον με την μέθοδο put_Extra και παραμέτρους το όνομα της μεταβλητής και την τιμή της γεμίζει το αντικείμενο με τις πληροφορίες της παράστασης που είναι είδη γνωστές από το fragment και εμφανίζονται στο activity. Το activity γίνεται ορατό στο χρήστης με την μέθοδο startActivity() και παράμετρο το αντικείμενο intent. Ο παρακάτω κώδικας βρίσκεται στο fragment που καλεί το activity.

```
1 Intent int_detail = new Intent(mContext, ProductionInfo.class);
2                 int_detail.putExtra("desc",production.getDesc());
3                 int_detail.putExtra("id",production.getId());
4                 int_detail.putExtra("title", production.getTitle());
```

```
5                 int_detail.putExtra("producer", production.getProducer());
6                 int_detail.putExtra("url", production.getUrl());
7                 mContext.startActivity(int_detail);
```

Όσων αφορά τι γίνεται από την πλευρά του activity που καλείται. Όπως φαίνεται στο παρακάτω απόσπασμα κώδικα δημιουργείται αντικείμενο intent το οποίο είναι ίσο με τη μέθοδο getIntent(), με αυτό το αντικείμενο παίρνει ο προγραμματιστής την μεταβλητή που επιθυμεί εκτελώντας τη μέθοδο ανάλογα με τον τύπο μεταβλητής είτε αυτή είναι String, integer και τα λοιπά, και περνάει ως παράμετρο το όνομα της μεταβλητής το οποίο πρέπει να συμπίπτει με το όνομα που έχει βάλει στο fragment. Έτσι όλες οι απαραίτητες μεταβλητές έχουν μεταφερθεί. Όλες αυτές οι ενέργειες γίνονται στη μέθοδο onCreate() του activity.

```
1 Intent intent = getIntent();
2         String title = intent.getStringExtra("title");
3         String Desc = intent.getStringExtra("desc");
4         String prod = intent.getStringExtra("producer");
5         String url = intent.getStringExtra("url");
```

Ωστόσο δεν έχει αναφερθεί το δεύτερο είδος intent το implicit. Χρησιμοποιήθηκε στη λειτουργία εμφάνισης της τοποθεσίας του θεάτρου στο google maps. Ο κώδικας φαίνεται στο κεφάλαιο 5,6. Η διαφορά με το explicit Intent είναι ότι στο αντικείμενο Intent υπάρχει παράμετρος action και όχι κάποιας συγκεκριμένης κλάσης. Στη συγκεκριμένη περίπτωση είναι το ACTION_VIEW το οποίο αφορά κυρίως προβολή. Επίσης δεν υπάρχει putExtra αλλά setData που σημαίνει ότι δεν περνάμε μεταβλητές αλλά πληροφορία που θέλουμε να ‘διαβαστεί’ από την άλλη εφαρμογή που θα ανοιχτεί.

7.3.5 MPAndroidChart

Στην εφαρμογή ανάκτησης δεδομένων για το θέατρο έχει αναφερθεί ότι υπάρχουν οθόνες με διαγράμματα. Τα διαγράμματα αυτά προήλθαν από νούμερα και στατιστικά που προήλθαν από τη βάση της εφαρμογής. Υπάρχει μία βιβλιοθήκη που συλλέγει αυτά τα νούμερα και τα μετατρέπει σε διαγράμματα είναι MPAndroidChart. Στο grandle είναι ως com.github.PhilJay:MPAndroidChart και η έκδοση που χρησιμοποιείται είναι η 3.1.0.

Για να δημιουργηθεί ένα διάγραμμα αρχικά πρέπει να δημιουργηθεί item στο xml της οθόνης. Εκεί ο προγραμματιστής επιλέγει αν θα είναι ραβδόγραμμα, πίτα ή άλλο διάγραμμα σύμφωνα με το element που θα μπει για παράδειγμα το παρακάτω απόσπασμα είναι ένα ραβδόγραμμα.

```
1 <com.github.mikephil.charting.charts.BarChart
2             android:layout_width="match_parent"
3             android:layout_height="match_parent"
4             android:id="@+id/eventChart"/>
```

Όσον αφορά τον τρόπο που ‘γεμίζει’ αυτό το διάγραμμα περνάμε στον κώδικα java. Στην κλάση του αντίστοιχου fragment ή activity που επιθυμείται να εμφανιστεί το διάγραμμα πρέπει να δημιουργηθεί μία μέθοδος στην οποία παίρνει μορφή. Αφού έχει γίνει η μετατροπή σε αντικείμενο το αντίστοιχο element του xml με το id του, δημιουργείται μία λίστα με τα δεδομένα που θα εμφανιστούν. Η λίστα είναι τύπου BarEntry ένας τύπος που υπάρχει στην βιβλιοθήκη και για να δημιουργηθεί αντικείμενο αυτού του τύπου υπάρχει μέθοδος BarEntry που δέχεται ως παράμετρο τα στοιχεία x και y στο

διάγραμμα. Αμέσως μετά με τη μέθοδο BarDataSet φτιάχνεται αντικείμενο από αυτό με παράμετρο τη λίστα ώστε με τη σειρά του αυτό να έχει την κατάλληλη μορφή και να περαστεί ως παράμετρο στο αντικείμενο BarData. Μετά από αυτή τη διαδικασία ο προγραμματιστής έχει στο χέρι του την κατάλληλη μορφή ώστε η λίστα με τα δεδομένα να περαστεί στο διάγραμμα, αυτό γίνεται με το αντικείμενο του διαγράμματος και τη μέθοδο setData και παράμετρο την τελευταία μορφή δεδομένων. Τα βασικά βήματα γίνανε, μετά είναι στη θέση του προγραμματιστή να δώσει τη μορφή, χρώματα, τίτλους ακόμα και κάποια προσθήκη animation στο διάγραμμα ώστε να το φέρει στη μορφή που επιθυμεί. Όλα αυτά θα γίνουν με μια αλληλουχία από μεθόδους που υπάρχουν μέσω της βιβλιοθήκης. Ένα παράδειγμα της μεθόδου που δίνει σχήμα και δεδομένα σε ένα ραβδόγραμμα είναι αυτός ο κώδικας που υπάρχει στο activity σύγκρισης των παραστάσεων σύμφωνα με τις διοργανώσεις που έχουν.

```
3
4  private  void setUpBarChart()
5      {
6          if(barEntriesArrayList.size() > 0) {
7              barDataSet = new BarDataSet(barEntriesArrayList, "eventNumber");
8              barData = new BarData(barDataSet);
9              allEvents.setData(barData);
10             barDataSet.setColors(ColorTemplate.MATERIAL_COLORS); // χρώματα
11             barDataSet.setValueTextColor(Color.BLACK);// χρώμα αριθμών
12             barDataSet.setValueTextSize(16f);// μέγεθος τίτλου
13             Description description = new Description();
14             description.setText("Events");
15             allEvents.setDescription(description); // περιγραφή διαγράμματος
16             allEvents.animateY(1000); // animation φόρτωσης
17             allEvents.invalidate();
18             //set XAxis formater Διαδικασία δημιουργίας τίτλων ανά ράβδο
19             XAxis xAxis = allEvents.getXAxis();
20             xAxis.setLabelCount(labels.size());
21             xAxis.setValueFormatter(new IndexAxisValueFormatter(labels));
22             //xAxis.setValueFormatter(new formaterLabel(labels));
23             xAxis.setGranularity(1f);
24             xAxis.setPosition(XAxis.XAxisPosition.BOTTOM);
25             xAxis.setDrawGridLines(false);
26             xAxis.setGranularityEnabled(true);
27             xAxis.setDrawAxisLine(false);
28             xAxis.setLabelRotationAngle(35);
29
```

8 Κεφάλαιο : Java vs Kotlin

Όπως έγινε γνωστό σε προηγούμενα κεφάλαια για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το εργαλείο Android studio. Το συγκεκριμένο εργαλείο είναι συμβατό με δύο γλώσσες την Java και τη Kotlin. Η δεύτερη είναι μία σχετικά πρόσφατη προσθήκη, στη συνέχεια του κεφαλαίου θα αναλυθούν οι δυνατότητες της κάθε γλώσσας καθώς και γιατί επιλέχθηκε η μία από αυτές για την ανάπτυξη της εφαρμογής ανάκτησης δεδομένων για το θέατρο.

8.1 Java

Η Java είναι μία γλώσσα προγραμματισμού που υπάρχει από το 1995. Ήταν η πρώτη γλώσσα που χρησιμοποίησε το Android studio για την ανάπτυξη εφαρμογών android. Η περιβαλλοντική ομοιότητα που υποστηρίζεται από την Java καθώς το ίδιο το Android αναπτύσσεται και γράφεται σε αυτό καθιστά τη γλώσσα να είναι η πρώτη επιλογή των προγραμματιστών που ασχολούνται με ανάπτυξη android εφαρμογών.

Το OpenJDK είναι η κύρια εφαρμογή της γλώσσας Java μέχρι σήμερα. Μιλάμε για μια αντικειμενοστραφείς (OOP) γλώσσα με πολυπλατφορμική ικανότητα. λειτουργεί πρακτικά σε οποιαδήποτε συσκευή, διακομιστή ή λειτουργικό σύστημα. Ο κώδικας της είναι επαναχρησιμοποιήσιμος καθιστώντας πιο εύκολη την ανάγνωση και κατανόησή του από τρίτους. Η ανάπτυξη σε Java βασίζεται κυρίως σε δοκιμές και απαιτεί από τον χρήστη να γράψει περισσότερο κώδικα και έχει μεγαλύτερη πιθανότητα σφαλμάτων και σφαλμάτων. Ωστόσο η διαχείριση και η αφαίρεση σφαλμάτων είναι απλή και ίσως πολλές φορές φαίνεται ευκολότερη στους προγραμματιστές σε σύγκριση με την Kotlin η οποία όπως θα σημειωθεί παρακάτω είναι πιο συνοπτική. Λόγω της πολυπλοκότητάς της είναι λίγο πιο αργή από άλλες γλώσσες και απαιτεί μεγάλο μερίδιο της μνήμης της συσκευής.

Ένα μεγάλο πλεονέκτημα της Java είναι αυτό της ασφάλειας της. Έχει ισχυρά μέτρα και πρότυπα ασφαλείας που μειώνουν την πιθανότητα αλλοίωσης της μνήμης. Επίσης, η Java έχει μια καλή βάση κωδικοποίησης που την καθιστά δυνατή.

Σε αντίθεση με την Kotlin όλες οι μέθοδοι και μεταβλητές έχουν τύπο. Δεν μπορεί να χρησιμοποιηθεί στη java μία μεταβλητή χωρίς να έχει δηλωθεί. Διατίθεται σε Java. Επιπλέον υποστηρίζει τη χρήση wildcard, ανώνυμος τύπος δηλαδή τύπος μπαλαντέρ. Διευκολύνει τον προγραμματιστεί όταν σε μία μέθοδο δεν στέλνει ή δεν λαμβάνει τον ίδιο τύπο κάθε φορά, αναπαρίσταται με το σύμβολο (?). Αξιοσημείωτο είναι ότι η κληρονομικότητα στη Java γίνεται σε επίπεδο κλάσης δεν γίνεται σε μεθόδους σε αντίθεση με την Kotlin όπου μπορεί να κληρονομηθεί μία μέθοδος μόνη της.

Η java εφαρμόζεται σε μεγάλες εφαρμογές με πολλές δυνατότητες και λειτουργίες που πρέπει να λειτουργούν σε όλες τις πλατφόρμες, συμπεριλαμβανομένων των Android, iOS, Windows ή Linux. Διαθέτει ώριμες βιβλιοθήκες που υποστηρίζουν καλά αυτόν τον τύπο ανάπτυξης εφαρμογών.

Συμπερασματικά, σύμφωνα με το η Java χρησιμοποιείται για την ανάπτυξη Android επειδή, είναι μια πολύ γνωστή γλώσσα μεταξύ των προγραμματιστών, δεν έχει επιπλοκές στην αριθμητική του δείκτη. Δεδομένου ότι εκτελείται σε εικονική μηχανή, δεν απαιτείται να μεταγλωττιστεί ξανά ο κώδικας για κάθε συσκευή, για τον κωδικό που χρησιμοποίησε. Αν και η ταχύτητα είναι ένα ζήτημα για την JAVA, ωστόσο η δημοτικότητα και τα πλεονεκτήματά της υπερισχύουν της ταχύτητας.

8.2 Kotlin

Από ιστορική σκοπιά, η IDE Jet Brains επινόησε αυτή τη νέα γλώσσα προγραμματισμού που ονομάζεται Kotlin. Σχεδιάστηκε για να χειρίζεται σημαντικά χαρακτηριστικά που δεν ήταν δυνατό να χειριστούν μέσω Java. Ενώ το Kotlin κυκλοφορεί τουλάχιστον από το 2011, η επίσημη κυκλοφορία του ήρθε το 2016. Στις 17 Μαΐου 2017 σύμφωνα με την ανακοίνωση η ομάδα της Android παρουσιάζει την προσθήκη της γλώσσα Kotlin ως γλώσσα ανάπτυξης εφαρμογών. Για πολλά χρόνια υπήρχε ως επιλογή ωστόσο το Μάιο του 2019 η google την καθιέρωσε ως την προτεινόμενη γλώσσα προγραμματισμού για εφαρμογές android. Κάποια από τα μεγαλύτερα ονόματα του κλάδου το Twitter, το Airbnb και το Netflix, μετέτρεψαν τον κώδικά τους σε Kotlin δίνοντας μεγαλύτερη αξία στη συγκεκριμένη γλώσσα.

Ένας από τους πρωταρχικούς στόχους της Kotlin ήταν να επιτύχει προσβασιμότητα σε όλες τις πλατφόρμες. Μια απλή έννοια της ονομαστικής αξίας, η αποστολή του Kotlin πηγαίνει πολύ πιο βαθιά. Στόχος του είναι να καταστήσει δυνατή και προσβάσιμη την κοινή χρήση κώδικα μεταξύ όλων των πλατφορμών. Η κυκλοφορία του Kotlin 1.3 ήταν ένα βήμα προς τη σωστή κατεύθυνση με βελτιώσεις στο Kotlin/Native που προάγουν την ευκολία πολλαπλών πλατφορμών.

Η Kotlin είναι ένας συνδυασμός λειτουργικού και αντικειμενοστραφούς προγραμματισμού. Μπορούν να χρησιμοποιηθούν σε αυτή πολλές δυνατότητες της Java, όπως, η ενσωμάτωση στοιβών Java με το Kotlin μπορεί επίσης να γίνει απρόσκοπτα με την εγκατάσταση μόνο ενός πρόσθετου. Αυτό το πρόσθετο θα σας επιτρέψει να μεταβείτε από Java σε Kotlin. Στην περίπτωση του android πολλές βιβλιοθήκες της Java μπορούν να χρησιμοποιηθούν σε Kotlin. Είναι μια γλώσσα προγραμματισμού ανοιχτού κώδικα που επιτρέπει στους προγραμματιστές να εκτελούν Java Virtual Machine (JVM). Σε αντίθεση με την Java η ύπαρξη ανοιχτού κώδικα δίνει στην Kotlin ένα σημαντικό πλεονέκτημα ασφάλειας και ανάπτυξης.

Μεταξύ πολλών πραγμάτων για τα οποία φημίζεται η Java, η συνοπτικότητα του κώδικά της δεν είναι ένα από αυτά. Αυτό είναι ένα πραγματικό αγκάθι για τους προγραμματιστές. Η Kotlin είναι πολύ πιο συνοπτική, αλλά και η αναγνωσιμότητα που παρέχει είναι επίσης πιο ξεκάθαρη. Ο λιγότερος εντατικός περιορισμός κειμένου του Kotlin συνοδεύεται από τη δυνατότητα καλύτερης ανάγνωσης και κατανόησης της κωδικοποίησης. Αυτό επιτρέπει στους προγραμματιστές να αλλάζουν, να επεξεργάζονται ή να βελτιώνουν ένα προϊόν με σχετική ευκολία σε αντίθεση με την Java. Αυτή η μείωση του κώδικα λύνει ένα μεγάλο πρόβλημα για τους προγραμματιστές, για τους οποίους ο έλεγχος για σφάλματα είναι απίστευτα δύσκολος. Με λιγότερες γραμμές κώδικα η διαδικασία αυτή είναι πιο σύντομη. Ωστόσο για κάποιον νέο σε αυτή τη γλώσσα είναι δύσκολη η ανάγνωση, η κατανόηση και η συντήρηση του κώδικα λόγω των αδύναμων μοτίβων της. Σε αντίθεση με τη Java η οποία είναι πιο ‘περιγραφική’

Όσον αφορά κάποια τεχνικά χαρακτηριστικά τα οποία αξίζει να σημειωθούν. Οι πίνακες δεν είχαν αμετάβλητο στην Java, αλλά το έχουν στο Kotlin. Τόσο στην Java όσο και την Kotlin υπάρχουν interfaces τα οποία διαθέτουν μία abstract μέθοδο αυτά ονομάζονται Single Abstract Method Interfaces (SAM Interfaces), Οι μετατροπές αυτών στην Java δεν είχαν τύπους συναρτήσεων, αλλά η Kotlin έχει. Επιπλέον οι εξαιρέσεις (exceptions) στον κώδικα δεν δηλώνονται σε συγκεκριμένη μέθοδο, για παράδειγμα η εντολή throws γνωστή η χρήση της στην java για τα exceptions στην Kotlin δεν υπάρχει. Ενώ σημαντικό να σημειωθεί είναι η διαχείριση της Kotlin στις null (μηδενικές) τιμές. Πολλές είναι οι γλώσσες προγραμματισμού που διαχειρίζονται τις μηδενικές τιμές ως λάθη και εμφανίζουν exceptions στην περίπτωση της Kotllin απλά θα εμφανιστεί μηδενική ή κενή τιμή.

Εκτός από κάποιες τεχνικές λειτουργίες της γλώσσας, υπάρχουν περιπτώσεις που συνιστάται να χρησιμοποιείται. Όπως Εφαρμογές όπου η απόδοση έχει μεγάλη σημασία, όπως αυτές που πρέπει να λειτουργούν ομαλά σε παλαιότερα τηλέφωνα Android ή σε αυτά που χρησιμοποιούνται για επεξεργασία φωτογραφιών. Έχει πιο βελτιωμένο και αποδοτικό σχεδιασμό από την Java, επομένως θα έχει καλύτερη απόδοση σε τέτοιου είδους καταστάσεις, ειδικά όταν πρόκειται για κλιμάκωση. Εφαρμογές που πρέπει να διατηρήσουν την ανεξαρτησία της πλατφόρμας και να διασταυρωθούν για πολλαπλές πλατφόρμες καθώς και για Android. Η Kotlin μπορεί να εκτελέσει αυτές τις λειτουργίες ενώ η Java δεν μπορεί λόγω της χρήσης bytecode που μπορεί να μεταγλωττίσει κώδικα μόνο για μια συγκεκριμένη πλατφόρμα τη φορά.

Μερικές από τις πιο πρόσφατες καινοτομίες έχουν κάνει την κλιμάκωση των εφαρμογών για κινητά πιο εφικτή, είναι η δυνατότητα επαναχρησιμοποίησης κώδικα για ουσιαστική εξοικονόμηση χρόνου και προσπάθειας. Αυτό επιτρέπει στους προγραμματιστές να εστιάσουν και να αφιερώσουν επιπλέον χρόνο σε πολύ πιο απαιτητικές εργασίες.

8.3 Ποια χρησιμοποιήθηκε και γιατί

Για την υλοποίηση της εφαρμογής ανάκτησης δεδομένων για το θέατρο χρησιμοποιήθηκε η γλώσσα προγραμματισμού java. Δεδομένου της παλαιότητάς της υπήρχε αρκετό υλικό και παραδείγματα για κάποιον που χρησιμοποιεί πρώτη φορά android studio. Υπάρχουν περισσότερες υλοποιήσεις και βιβλιοθήκες για διάφορες λειτουργίες. Αν και κάποιες από αυτές είναι συμβατές με Kotlin υπήρχαν αρκετές διαφορές στην πράξη. Έτσι με τον μεγάλο αριθμό έγκυρων βιβλιοθηκών και λειτουργιών κάποιες τεχνολογίες γίνονταν πιο εύκολα με τη χρήση Java.

Είναι μία δομημένη γλώσσα, υπάρχουν αρκετές οδηγίες σε περίπτωση λαθών και εξαιρέσεων από τον ίδιο των compiler ώστε ο τελικός κώδικας να έχει το ορθό αποτέλεσμα. Αν και χρειάζεται περισσότερες γραμμές κώδικα η Java για μία λειτουργία από ότι η Kotlin, είναι πιο ‘περιγραφική’ και μπορούν να εντοπιστούν λάθη πιο εύκολα και να γίνουν μετατροπές σε κώδικα τρίτων διότι είναι πιο κατανοητός.

Γενικότερα ο βασικός λόγος της επιλογής αυτής ήταν η διαθεσιμότητα εργαλείων και της εκτενής τεκμηρίωσης αυτών λόγω της παλαιότητάς της. Ωστόσο είναι στη θέση του κάθε προγραμματιστή, των αναγκών και των γνώσεων του ποια από τις παραπάνω γλώσσες θα επιλέξει, αφού και οι δύο έχουν εγκριθεί και έχουν υλοποιηθεί από μεγάλες εφαρμογές android.

9 Κεφάλαιο : Χρονοδιάγραμμα υλοποίησης εφαρμογής

Όπως έχει αναφερθεί η ανάγκη για την υλοποίηση της εφαρμογής είναι αποτέλεσμα που δημιουργήθηκε από άλλες πτυχιακές με αποτέλεσμα η συνεργασία με την ευρύτερη ομάδα να αποτελεί κομμάτι για την επίτευξη του τελικού αποτελέσματος, χωρίς ωστόσο να υπάρχει πλήρη εξάρτηση από άλλες τις εργασίες. Σε αυτό το κεφάλαιο θα αναλυθούν τα βήματα που έγιναν ώστε να υπάρχει επιθυμητό αποτέλεσμα.

9.1 Απαιτήσεις και πως ξεκίνησαν

Αρχικά πριν υπάρξει ανάγκη για υλοποίηση εφαρμογής με τη βοήθεια της τεχνολογίας scrapping συλλέχθηκαν δεδομένα για θεατρικές παραστάσεις τα οποία περάστηκαν σε βάση. Με κάποιον τρόπο έπρεπε αυτά τα δεδομένα να παρουσιαστούν με τρόπο φιλικό σε έναν χρήστη έτσι έπρεπε να γίνει μία web εφαρμογή όπως και μία εφαρμογή για android κινητά και για IOS. Για να μπορέσουν και οι τρεις να έχουν τα δεδομένα από τη βάση έπρεπε να δημιουργηθεί API το οποίο παίρνει τα δεδομένα από τη βάση και τα μετατρέπει σε JSON.

Οι προγραμματιστές των τριών προϊόντων-εφαρμογών κλήθηκαν να δώσουν την καλύτερη παρουσίαση των δεδομένων της βάσης ώστε ο χρήστης να έχει πλήρη και εύκολη πρόσβαση και γνώση για τις παραστάσεις, τους ηθοποιούς, τα θέατρα και τις διοργανώσεις. Έτσι έγιναν συναντήσεις και συλλέχθηκαν αιτήματα για τον διαχειριστή του API ώστε τα δεδομένα JSON να έρχονται κατάλληλα. Κάποια από τα αιτήματα που έγιναν ήταν τα εξής:

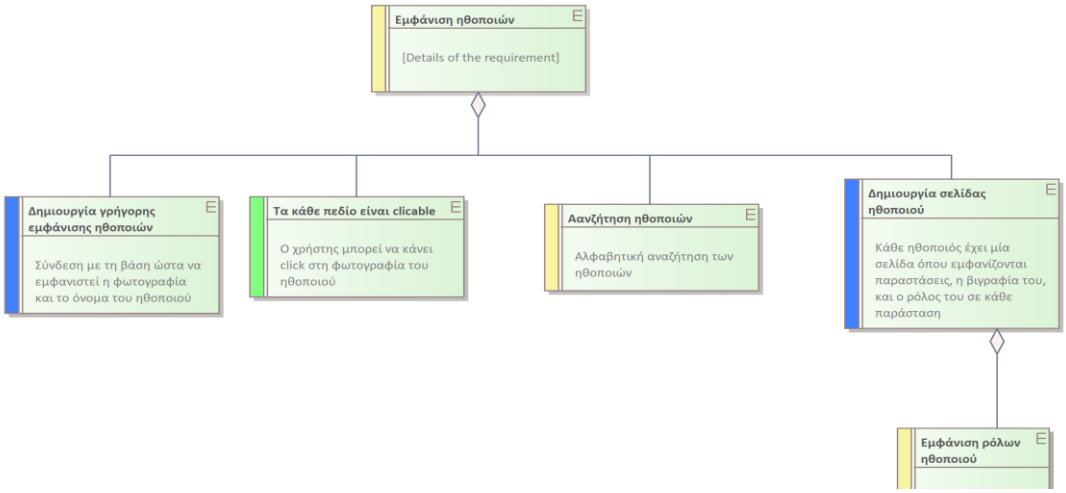
- Με το id του ηθοποιού να εμφανίζονται οι παραστάσεις που έχει συμμετάσχει.
- Να εμφανίζονται οι πιο πρόσφατες ημερολογιακά παραστάσεις
- Με το id του ηθοποιού να υπάρχει η λίστα των φωτογραφιών του
- Με το id της παράστασης να έρχονται οι διοργανώσεις που έγιναν με αυτήν και σε ποια θέατρα
- Όταν είναι γνωστό το id παράστασης, ηθοποιού ή θεάτρου να έρχονται πληροφορίες μόνο για αυτό το id

Αφού υλοποιήθηκαν αυτά τα αιτήματα τότε συνεχίστηκαν τα επόμενα βήματα υλοποίησης η διαδικασία αυτή διήρκησε περίπου 3 μήνες διότι έπρεπε να προηγηθεί το βήμα του brainstorming. Η διαδικασία αυτή είναι όταν όλη η ομάδα αναφέρει ιδέες για το ποιος είναι ο γενικός σκοπός, πως θα γίνει η διαχείριση και η εμφάνιση των δεδομένων καθώς και πολλά αιτήματα όπως τα παραπάνω από τα οποία βγήκαν και τα τελικά που δόθηκαν στον υπεύθυνο του API.

9.2 Σχεδίαση οθονών

Στην αρχή κάθε εφαρμογή πρέπει να γίνει καταγραφή των λειτουργιών που θα έχει κάθε οθόνη , με αυτή την καταγραφή δημιουργήθηκε η ανάγκη είτε να επαναπροσδιοριστούν είτε διατυπωθούν επιπλέον αιτήματα που αφορούσαν το ακριβώς προηγούμενο βήμα κεφάλαιο 8.1. Έγινε σχεδίασε των αιτημάτων για κάθε οθόνη όπως φαίνεται στην εικόνα 8,1 στην οποία είναι η σχεδίαση των λειτουργιών που θα υπήρχαν στην οθόνη των ηθοποιών.

9 Κεφάλαιο



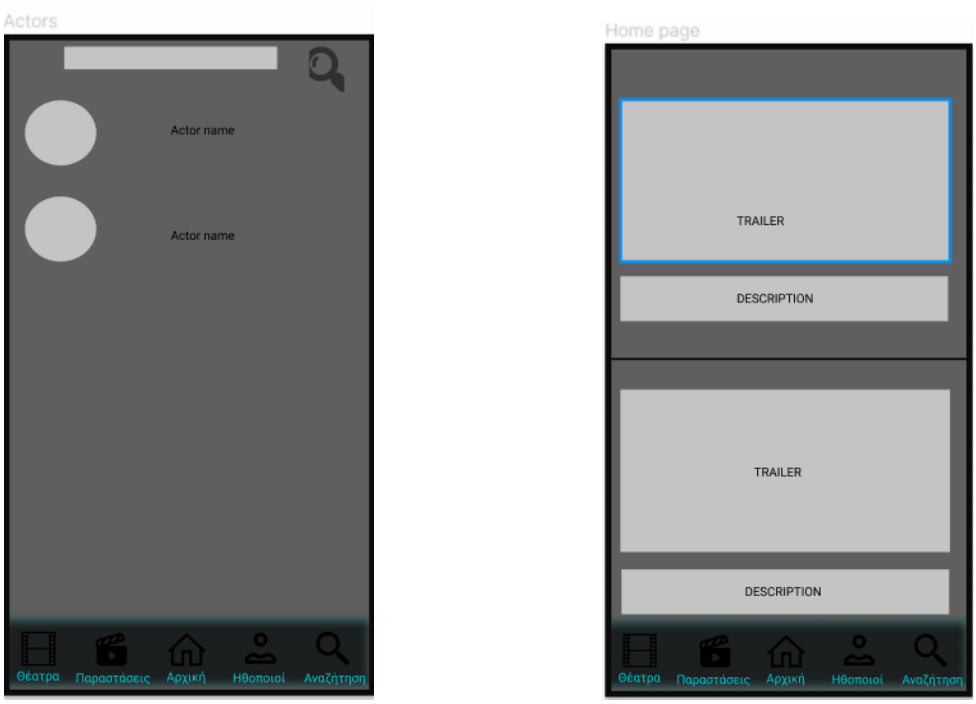
Εικόνα 8.1 : Λειτουργίες εφαρμογής

Εκτός από την καταγραφή των λειτουργιών της κάθε οθόνης, σε κάθε project χρειάζεται η ομάδα να οργανώσει τον τρόπο εμφάνισης των πληροφοριών, δηλαδή πως περίπου θα είναι οι οθόνες. Σε αυτή την διαδικασία πήραν μέρος οι δύο πανομοιότυπες εφαρμογές δηλαδή αυτή που αφορά το λειτουργικό android και αυτή του IOS. Η διαδικασία διήρκησε περίπου ένα μήνα καθώς γινόταν συνέχεια αλλαγές ανάλογα με τις ανάγκες που προ έκυπταν. Για παράδειγμα στην αριστερή οθόνη της εικόνας 8,1 έγινε ένα σχέδιο οθόνης το οποίο κρίθηκε από την ομάδα ότι δεν ήταν η κατάλληλη μορφή αρχικής οθόνης και θα δημιουργούσε σύγχυση στον χρήστη έτσι απορρίφθηκε.

Υπάρχουν πολλά εργαλεία που βοηθάνε στην σχεδίαση οθονών στην συγκεκριμένη περίπτωση χρησιμοποιήθηκε ένα online που ονομάζεται Figma. Η διαδικασία που χρησιμοποιεί το θυμίζει το γνωστό σε αρκετούς drive της google. Δηλαδή υπάρχει ένα αρχείο στο οποίο αυτό που το δημιουργεί δίνει δικαίωμα προβολής ή και συγγραφής σε άλλους χρήστες. Άρα πολλοί χρήστες μπορούν να έχουν ένα κοινό αρχείο όπου κάνουν αλλαγές ταυτόχρονα και ενημερώνουν ο ένας τον άλλον με αυτό τον τρόπο για τις προτάσεις που έχουν ώστε να φτάσουν στο τελικό αποτέλεσμα.

Εκτός από την ευκολία στην συνεργασία η χρήση του συγκεκριμένου εργαλείου έγινε εξαιτίας της εύκολης χρήσης του και του αποτελέσματος που προσφέρει. Υπάρχει η ικανότητα επιλογής οθόνης από μια μεγάλη γκάμα επιλογών καθώς και μπορεί να γίνει προσαρμογή σύμφωνα με τις διαστάσεις. Επιπλέον στην οθόνη που σχεδιάζεται μπορούν να προστεθούν και να τροποποιηθούν φωτογραφίες και σχήματα από το διαδίκτυο κάνοντας ακόμα πιο εύκολο και αποδοτικό το έργο του σχεδιασμού.

Στην εικόνα 8.2 φαίνεται ένας πρώτος σχεδιασμός που έγινε για την οθόνη των ηθοποιών και της αρχικής οθόνης όλα τα σχήματα και χρώματα τροποποιήθηκα και προστέθηκαν ξεχωριστά καθώς και οι σκιάσεις.



Εικόνα 8.2: οθόνες Figma

9.3 Η αρχή του κώδικα

Εκτός από τα βήματα που αφορούσαν συνεργασία με την υπόλοιπη ομάδα σημαντική είναι και η προεργασία που έγινε πριν την συγγραφή κώδικα.

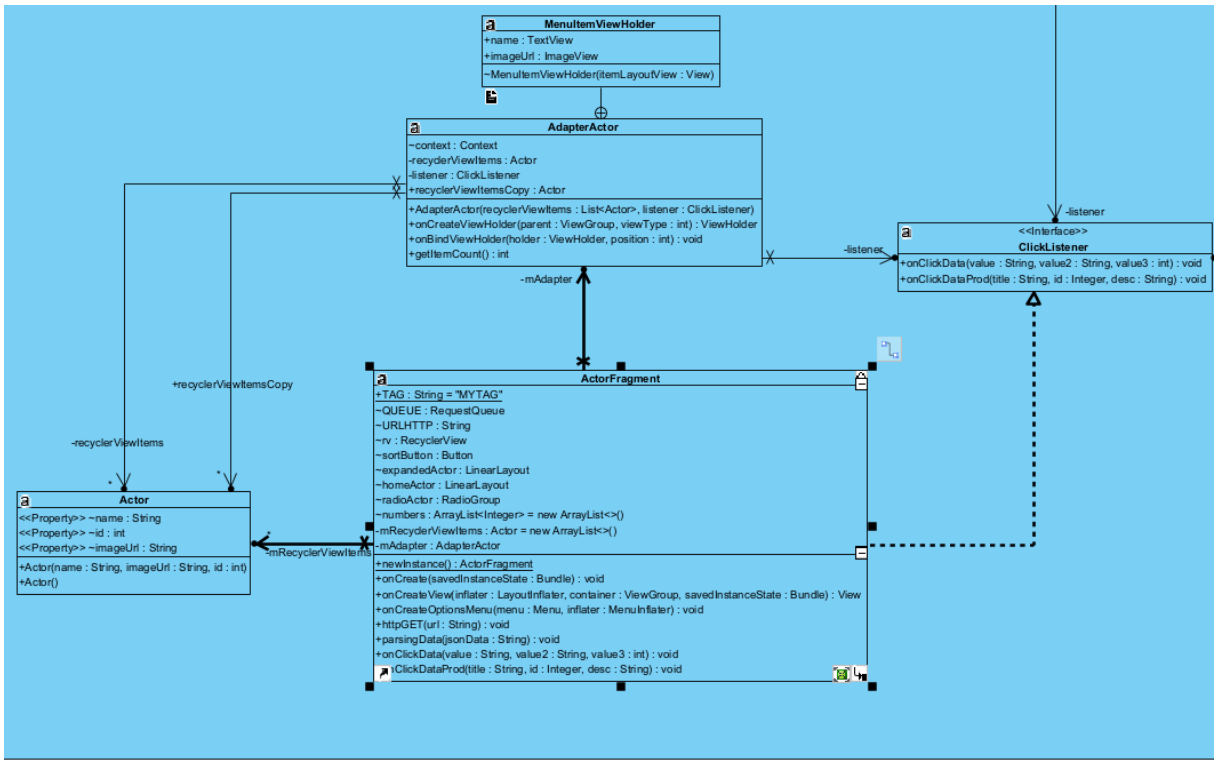
Αφού έγιναν οι απαραίτητες διορθώσεις και οι οθόνες που αναφέρθηκαν στο κεφάλαιο 8.2 πήραν την τελική τους μορφή, έγινε η σχεδίασή τους στο android studio. Ουσιαστικά έγιναν οι οθόνες της εφαρμογές και γράφτηκαν σε xml, η οποία όπως αναφέρθηκε στο κεφάλαιο 2.3 είναι η γλώσσα που χρησιμοποιεί το android studio για την σχεδίαση layout της εφαρμογής. Η διαδικασία δημιουργίας των xml αρχείων διήρκησε περίπου 3 εβδομάδες λόγω ότι δεν χρησιμοποιήθηκαν έτοιμα template και όλα έγιναν με κώδικα και μορφοποίηση με τη βοήθεια των εργαλείων που διαθέτει το android studio για τις οθόνες.

Αυτές οι οθόνες έπρεπε να είναι λειτουργικές, για να επιτευχθεί αυτό χρειάζεται κώδικας οποίος επιλέχθηκε να είναι σε γλώσσα Java όπως αναφέρθηκε στο κεφάλαιο 8. Έτσι έπρεπε να γίνει σχεδίαση των κλάσεων ώστε να υπάρχει μία εικόνα τις εφαρμογής πριν την συγγραφή του. Ένας διαδεδομένος τρόπος για να γίνει αυτό είναι η δημιουργία διαγραμμάτων UML. Με τη βοήθεια του εργαλείου visual Paradigm, είναι ένα γνωστό εργαλείο το οποίο έχει ένα περιβάλλον κατάλληλο για την ανάπτυξη UML διαγραμμάτων.

Η ενοποιημένη γλώσσα σχεδιασμού (unified modeling language) όπως λέγεται UML είναι μια γραφική γλώσσα για την οπτική αναπαράσταση και τη διαμόρφωση προδιαγραφών ενός λειτουργικού συστήματος. Είναι κατάλληλη για το σχεδιασμό αντικειμενοστραφών συστημάτων όπως πρόκειται για μία γλώσσα όπως η Java. Το σχέδιο είναι μια απλοποιημένη παράσταση της πραγματικότητας. Είναι πολλά τα διαγράμματα στη γλώσσα uml αφορούν είτε απαιτήσεις ενός συστήματος, τις κλάσεις ακόμα και την ακολουθία μιας ενέργειας του χρήστη. Το είδος διαγράμματος που χρησιμοποιήθηκε είναι αυτό των κλάσεων το οποίο ονομάζεται class diagram. Με αυτό δεδομένη τη γνώση ότι κάθε activity και fragment έχει τη δική του κλάση εξελίχθηκε το διάγραμμα σύμφωνα με τις ανάγκες των

λειτουργιών. Στην εικόνα 8.2 φαίνεται μέρος από τα διαγράμματα. Αφορά ένα μέρος του κώδικα των ηθοποιών. Όπως γίνεται αντιληπτό υπάρχουν όλες οι συνδέσεις και τα περιεχόμενα της κάθε κλάσης. Με αυτή τη σχεδίαση έγινε πιο απλή η συγγραφή του κώδικα διότι όταν κάτι φαινόταν ότι δεν θα λειτουργήσει είτε δεν μπορούσε μία σύνδεση να γίνει σωστά στο σχήμα απλά άλλαζε το σχήμα δεν χρειάστηκε να σβηστεί και να αλλάξει κώδικας. Η όλη διαδικασία είχε και ως αποτέλεσμα να γίνουν πιο εύκολα αντιληπτές κάποιες διορθώσεις και refactor στον κώδικα που μπορούν να γίνουν στο μέλλον για την καλύτερη λειτουργία της εφαρμογής.

Αφού τα σχήματα ήρθαν σε μία ικανοποιητική μορφή ώστε να είναι οι συνδέσεις σωστές τότε προχώρησε η διαδικασία στην υλοποίηση. Η διαδικασία αυτή είχε διάρκεια 3 μήνες. Το βήμα αυτό συνδυάστηκε με αναζήτηση τεχνολογιών. Διότι οι απαιτήσεις και οι υλοποιήσεις ήταν γνωστές από πριν έγινε έρευνα βιβλιοθηκών που θα έπρεπε να χρησιμοποιηθούν και ο τρόπος χρήσης τους ώστε να υπάρχει μία μικρή ιδέα για την μορφή του κώδικα πριν τη σχεδίαση του class diagram. Ουσιαστικά πρώτα έγινε έρευνα, γέμισε το αρχείο grandle με τα απαραίτητα, αργότερα σχεδιάστηκαν οι οθόνες σε xml και τέλος τα διαγράμματα uml.



Εικόνα 8.2: class diagram Actors

9.4 Η υλοποίηση και εκτέλεση

Το τελικό βήμα της δημιουργίας της εφαρμογής θεατρικών παραστάσεων είναι αυτό της υλοποίησης και της δοκιμής και του τελικό αποτελέσματος. Αρχικά με βάση τα διαγράμματα έγιναν οι κλάσεις με τις αντίστοιχες μεθόδους τους. Με κάθε καινούρια λειτουργία που υλοποιούταν δοκιμαζόταν η λειτουργία της. Η εφαρμογή εκτελούταν με 3 τρόπους:

- Με τον ενσωματωμένο emulator του android studio. Υπάρχουν είτε αίτημα μοντέλα συσκευών είτε μπορεί να φτιαχτεί ένα custom μοντέλο ανάλογα τι χαρακτηριστικά θέλει ο

προγραμματιστής να έχει η συσκευή που θα τρέξει την εφαρμογή. Ωστόσο ο emulator αυτός απαιτεί μηχάνημα με δυνατά χαρακτηριστικά, όπως ram περίπου 16 GB, αλλιώς η διαδικασία οδηγείται σε επανωτά κολλήματα ακόμα και σε διακοπή, με αποτέλεσμα να μην μπορεί η εκτέλεση της εφαρμογής να γίνει.

- Με εξωτερικό emulator. Υπάρχουν πολλές εφαρμογές για windows οι οποίες προσφέρουν εικονικές συσκευές που μπορεί ο χρήστης να επιλέξει τα χαρακτηριστικά της συκευής που επιθυμεί να χειριστεί τα οποία είναι συμβατά με το android studio και ταυτόχρονα δεν απαιτούν ‘δυνατά ’ χαρακτηριστικά μηχανήματος. Η εφαρμογή που ονομάζεται bluestacks είναι αυτή που χρησιμοποιήθηκε στην συγκεκριμένη περίπτωση η οποία ενδείκνυται για υπολογιστές με ram 6gb.
- Με φυσική συσκευή, δηλαδή με ένα οποιοδήποτε κινητό βρίσκεται στην κατοχή του προγραμματιστή, το μόνο που αρκεί είναι να υπάρχει σύνδεση της συσκευής μέσω usb ή να βρίσκεται στο ίδιο δίκτυο με τον υπολογιστή που βρίσκεται ο κώδικας της εφαρμογής.

Έτσι με την επιλογή που προσφέρει το android studio να μπορεί η εφαρμογή να τρέχει ταυτόχρονα σε πολλές συσκευές μπορεί να εκλεχθεί ταυτόχρονα σε τουλάχιστον 3 συσκευές όπως αναφέρθηκε αν η λειτουργικότητα της εφαρμογής είναι σωστή. Επιπλέον κατά τη διάρκεια της εκτέλεσης διατίθεται ένας profiler, είναι ένα παράθυρο το οποίο δείχνει τι επίπτωση έχει η εφαρμογή σε κάθε συσκευή που τρέχει. Δηλαδή πόση CPU χρησιμοποιείται, πόσο μνήμη και ενέργεια από τη συσκευή καταλαμβάνει ώστε να έχει ο προγραμματιστής μία εικόνα της εφαρμογής του και κατά πόσο αυτή μπορεί να λειτουργήσει σε μία συσκευή ταυτόχρονα με άλλες. Το βήμα συγγραφής κώδικα και εκτέλεσης του διήρκησε άλλον 1 μήνα.

10 Κεφάλαιο : Συμπεράσματα και προτάσεις Βελτίωσης

10.1 Προτάσεις Βελτίωσης εφαρμογής

Σε αυτή την ενότητα θα συζητηθούν οι βελτιώσεις και προσθήκες που θα μπορούσαν να γίνουν στην εφαρμογή στο μέλλον. Μία εφαρμογή δεν μένει ποτέ σταθερή, πάντα μπορούν να προστεθούν και να βελτιωθούν πράγματα για να εξάπτει το ενδιαφέρον των χρηστών, καθώς και τη χρησιμότητα της για αυτό οι προτάσεις βελτίωσης και οι ενημερώσεις δεν σταματάνε. Ακόμα και μεγάλες εφαρμογές όπως είναι το Facebook, Instagram έχουν συνεχείς ενημερώσεις και προστίθενται καινούριες δυνατότητες.

Όσον αφορά την όψη της εφαρμογής θα μπορούσαν να γίνουν βελτιώσεις εμφάνισης κάποιων πληροφοριών και προσθήκες animation. Ένα από αυτά είναι η προσθήκη μίας οθόνης η οποία λέγεται splash screen. Είναι σύντομη η εμφάνιση της και θα γίνεται εμφανή κατά τη διάρκεια φόρτωσης των στοιχείων. Για παράδειγμα αν κατά την εντολή του χρήστη θα εμφανίζεται η λίστα των ηθοποιών μέχρι να φορτωθούν από την εφαρμογή οι ηθοποιοί θα εμφανίζεται ένα παράθυρο που θα γράφει loading το οποίο θα εξαφανίζεται όταν τα στοιχεία θα είναι αίτημα για εμφάνιση. Επιπλέον με την ίδια λογική μπορεί να προστεθεί μία splash οθόνη με το άνοιγμα της εφαρμογής η οποία θα έχει το logo της.

Εκτός από κάποιες εμφανισιακές προσθήκες μπορούν να γίνουν και τεχνικές προσθήκες. Μία ιδέα θα μπορούσε να είναι, στην οθόνη των πληροφοριών θεάτρου, όπου είναι το διάγραμμα πίτας για το ποσοστό πληρότητας του θεάτρου σύμφωνα με το μέσο όρο των θέσεων που ήταν κλεισμένες σε κάθε παράσταση, μπορεί να αντικατασταθεί από μία η οπτική εμφάνιση των θέσεων σε μορφή θεάτρου και να χρωματιστούν οι θέσεις σύμφωνα με το ποσό του μέσου όρου των θέσεων. Συνεχίζοντας μία ακόμα λειτουργεία είναι στην οθόνη πληροφοριών της κάθε παράστασης. Αντί για τη λίστα με τις διοργανώσεις της παράστασης μπορεί να ανοίγει το ημερολόγιο του κινητού του χρήστη μετά από συναίνεσή του για τη χρήση αυτού του ημερολογίου, να μαρκαριστούν όλες οι ημερομηνίες που ήταν οι διοργανώσεις και με σημείωση των λεπτομερειών αυτής. Ενώ εκτός από πληροφορία θα μπορούσε να αλλάξει η οπτική της εφαρμογής και να καλείται ο χρήστης αν επιθυμεί να γράψει κάποιο σχόλιο για την κάθε οντότητα(ηθοποιό, θέατρο, παράσταση) ώστε να έχει και πιο ενεργό ρόλο στην εφαρμογή.

Ωστόσο όταν μιλάμε για βελτίωση μιλάμε και στο κομμάτι του ήδη υπάρχων κώδικα. Μπορεί να γίνει μια καλύτερη διαχείριση και να χρησιμοποιηθεί πιο αποτελεσματικά η ικανότητα της retrofit κεφάλαιο 6,3,2 για την αποθήκευση προσωρινών δεδομένων ώστε τα αποτέλεσμα να εμφανίζονται πιο σύντομα στον χρήστη. Επιπλέον έχει αναφερθεί ότι για κάθε recyclerView υπάρχει ένας adapter θα μπορούσε να γίνει ένας κοινός adapter για όλα τα recyclerView ώστε να μειωθεί η ποσότητα του κώδικα και στο μέλλον να υπάρξει καλύτερη συντήρηση του κώδικα διότι είναι καλύτερο σε κάθε αλλαγή να επηρεάζεται ένα αρχείο παρά πολλά διαφορετικά. Αυτά θα βοηθήσουν και στην απόδοση της εφαρμογής. Ενώ δεν πρέπει να παραληφθεί η προσθήκη unit test στην εφαρμογή στα οποία το android studio έχει εύκολή διαχείριση και εκτενή τεκμηρίωση.

10.2 Συμπεράσματα χρήσης android studio

Όπως έχει γίνει αντιληπτό και έχει ειπωθεί από τα πρώτα κεφάλαια η android εφαρμογή ανάκτησης δεδομένων για το θέατρο δημιουργήθηκε με το εργαλείο android studio όπου στο κεφάλαιο 2,4 αναλύεται και η αιτιολογία της επιλογής του. Υπήρχαν όμως κάποια προβλήματα όσον αφορά τη λειτουργικότητά του τα οποία έχουν αναφερθεί και από άλλους χρήστες. Αρχικά μία προσωπική

παρατήρηση δεδομένου των πολλών αναβαθμίσεων έπρεπε συνεχώς να γίνονται αλλαγές στο gradle αρχείο ώστε οι βιβλιοθήκες να έχουν την τελευταία τους έκδοση. Επίσης αρκετές φορές έπρεπε να γίνουν αλλαγές και στον κώδικα java διότι κάποιοι μέθοδοι από τις βιβλιοθήκες καταργούνταν με τις ενημερώσεις και χρησιμοποιούνταν άλλες με αποτελέσματα να χάνεται χρόνος για να γίνουν σωστά οι λειτουργίες που είχαν ήδη υλοποιηθεί.

Όσον αφορά την επίδοση του εργαλείου, χρειάζεται υπολογιστή με καλές επιδόσεις διότι απαιτεί αρκετή ram για να λειτουργήσει. Επομένως είναι δύσκολο έως ακατόρθωτο σε πολλά μηχανήματα να γίνει multitasking κατά τη χρήση του. Όσες προσπάθειες και να έχουν γίνει με τις αναβαθμίσεις ακόμα υπάρχει καθυστέρηση στο build και run του κώδικα. Ενώ από πολλούς απλούς χρήστες και εταιρείες έχουν γίνει παράπονα για την επίδοση του emulator το οποίο σε πολλές περιπτώσεις έχει καθυστέρηση.

Εκτός από κάποια αρνητικά γενικά το android studio είναι ένα από τα καλύτερα εργαλεία που μπορεί να χρησιμοποιηθεί για ανάπτυξη εφαρμογών και το documentation, δηλαδή οι πληροφορίες που παρέχει η ίδια η android για τις λειτουργίες του είναι αρκετά λεπτομερής και κατανοητές και έχει καλυφθεί μεγάλη γκάμα πληροφοριών. Επιπλέον είναι σημαντική διευκόλυνση η παρουσίαση της προεπισκόπησης της οθόνης κατά τη συγγραφή κώδικα xml για να δοθεί μορφή και να προστεθούν στοιχεία και πληροφορίες στην οθόνη της εφαρμογής.

10.3 Γενικά Συμπεράσματα

Κατά τη διάρκεια εκπόνησης της εργασίας οι πληροφορίες που συλλέχτηκαν ήταν αρχικά η εκμάθηση του τρόπου χρήσης ενός εργαλείου όπως είναι τι android studio. Η δυνατότητα ανάπτυξης μία εφαρμογής ανάλογα με τις ανάγκες που ήταν επιθυμητό να καλύψει και ταυτόχρονα να μπορεί η εφαρμογή να γίνει αντιληπτή από τους χρήστες. Ουσιαστικά η αντίληψή σημαίνει το στήσιμο της εφαρμογής, οι οθόνες πως θα παρουσιαστεί μία πληροφορία ώστε να γίνει κατανοητή από το χρήστη μέχρι και τα χρώματα που χρησιμοποιήθηκαν ώστε να είναι ευανάγνωστο και ευχάριστο το αποτέλεσμα προς τρίτους.

Εκτός από τις τεχνικές γνώσεις που συγκεντρώθηκαν , όπως έχει αναφερθεί η εφαρμογή ανάκτησης δεδομένων για το θέατρο είναι προϊών ευρύτερης συνεργασίας άλλων εργασιών. Επομένως σημαντική ήταν εκμάθηση λειτουργίας μία ομάδας, η ανάγκη αναζήτησης και χρήσης εργαλείων που θα έκαναν αυτή τη συνεργασία ευκολότερη και πιο αποτελεσματική, καθώς και θα έπρεπε να γίνει αντιληπτό το ανεξάρτητο κομμάτι της κάθε εργασίας και ο διαχωρισμός της από τις υπόλοιπες. Ενώ στο πιλοτικό αυτό σύστημα συνεργασίες δεν έλλειψαν κάποια προβλήματα για τα οποία κλήθηκαν να αλλαχθούν κάποιες απαιτήσεις και κάποιες λειτουργείς είτε χρειάστηκε να αλλάξουν είτε να βρεθεί κάποια εναλλακτική για την υλοποίησή τους. Για παράδειγμα υπήρξαν κάποια προβλήματα στα ονόματα των ηθοποιών τα οποία σε κάποιες περιπτώσεις είχαν και χαρακτήρες οι οποίες δεν έπρεπε να εμφανίζονται στην εφαρμογή. Ωστόσο ήταν μία προσομοίωση πραγματικής συνεργασίας η οποία σίγουρα θα κληθεί κάποιος να συναντήσει και στο μέλλον.

Βιβλιογραφία

Internet Site

[1] https://www.britannica.com/technology/Android-operating-system

[2] https://developer.android.com/studio/build#kts

[3] https://source.android.com/docs/core/architecture

[4] https://android-developers.googleblog.com/2017/05/android-announces-support-for-kotlin.html

[5] https://www.securecoding.com/blog/kotlin-vs-java-nullpointerexception/

[6] https://www.figma.com/

[7] https://www.visual-paradigm.com/

[8] https://www.bluestacks.com/

[9] https://www.gartner.com/reviews/market/application-development-integration-and-management-others/vendor/google/product/android-studio

[10] https://www.trustradius.com/products/android-studio/reviews

[11] http://developer.android.com/index.html

[12] https://clearbridgemobile.com/java-vs-kotlin-which-is-the-better-option-for-android- app-development/

[13] https://developer.android.com/guide/topics/ui/layout/recyclerview?gclid=Cj0KCQjw9ZGYBhCEARIsAEUXITWpgfR_vziN8ZikfTxHAqxPB49JWtppL-qmEuJmZywXxzw2miGHF7waAtqcEALw_wcB&gclsrc=aw.ds

[14] https://developer.android.com/studio/releases/past-releases

[15] https://www.techopedia.com/definition/17056/cross-platform

[16] https://developer.android.com/guide/topics/ui/layout/cardview

[17] https://www.javainhand.com/2020/03/difference-between-retrofit-and-volley-in-android.html

[18] https://www.quora.com/What-is-the-difference-between-retrofit-and-volley-in-Android

[19] https://medium.com/@sudhakarprajapati7/retrofit-vs-volley-c6cf74b3c8e4

Papers

[20] Martinez, Matias & Mateus, Bruno. (2020). How and Why did developers migrate Android Applications from Java to Kotlin? A study based on code analysis and interviews with developers.

[21] Luca Ardito, Riccardo Coppola, Giovanni Malnati, Marco Torchiano, “Effectiveness of Kotlin vs. Java in android app development tasks, Information and Software Technology”, Volume 127, 2020, 106374, ISSN0950-5849, https://doi.org/10.1016/j.infsof.2020.106374. (https://www.sciencedirect.com/science/article/pii/S0950584920301439)

[22] Rajinder Singh, "An Overview of Android Operating System and Its Security Features", International Journal of Engineering Research and Applications, Volume 4, Number 2, 2014, pp. 519-521(3), Publisher**:** Directory of Open Access Journals

Βιβλία

[23] Uzayr, S.B. (2022). Mastering Android Studio: A Beginner's Guide (1st ed.). CRC Press. https://doi.org/10.1201/9781003229070

[24] Jie Lin, Chuanyi Liu, and Binxing Fang. 2019. Out-of-Domain Characteristic Based Hierarchical Emulator Detection for Mobile. In Proceedings of the 2nd International Conference on Information Technologies and Electrical Engineering (ICITEE-2019). Association for Computing Machinery, New York, NY, USA, Article 144, 1–5. https://doi.org/10.1145/3386415.3387091