ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

Ανάπτυξη λογισμικού για την απεικόνιση και αποθήκευση ηλεκτροκαρδιογραφήματος σε περιβάλλον Android

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Παντελής, Ε. Σύριγγας

Επιβλέπων : Γεώργιος Ματσόπουλος

Αναπληρωτής Καθηγητής

Αθήνα, Σεπτέμβριος 2015

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

Επιβλέπων : Γεώργιος Ματσόπουλος

Αναπληρωτής Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 9η Σεπτεμβρίου 2015

.....	.....	.....
Γεώργιος Ματσόπουλος Αναπληρωτής Καθηγητής Ε.Μ.Π.	Νικόλαος Ουζούνογλου Καθηγητής Ε.Μ.Π.	Παντελής Ασβεστάς Επίκουρος Καθηγητής Τ.Ε.Ι Αθήνας

Αθήνα, Σεπτέμβριος 2015

.....

Παντελής Ε. Σύριγγας

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Σκοπός της παρούσης διπλωματικής εργασίας είναι η σχεδίαση και ανάπτυξη ενός λογισμικού για την απεικόνιση και αποθήκευση ενός ηλεκτροκαρδιογραφικού σήματος. Η ανάπτυξη θα γίνει σε περιβάλλον android, χρησιμοποιώντας την γλώσσα προγραμματισμού java. Το περιβάλλον android επιτρέπει μια λειτουργική και γραφική απεικόνιση του βιοσήματος, φιλική προς τον χρήστη γιατρό και κυρίως να τον βοηθά, για να εξάγει εύκολα και γρήγορα συμπεράσματα. Για τον γιατρό είναι σημαντικό να έχει συνεχή εποπτεία των ασθενών του και ενημερώνεται για την πορεία της υγείας τους.

Η λήψη του σήματος θα γίνεται ασύρματα με την χρήση TCP sockets κάτι το οποίο επιτρέπει την μεταφορά και ανταλλαγή δεδομένων, είτε σε τοπικό δίκτυο, Διαδίκτυο ή μέσω δικτύου κινητής τηλεφωνίας.

Τέλος η αποθήκευση του βιοσήματος θα γίνει σε μια βάση δεδομένων SQL που περιέχει τους ασθενείς και το ιστορικό των βιοσημάτων τους για την ευκολία ανάκτησης του. Έτσι ο γιατρός θα μπορεί να μπορεί να ανατρέξει στο ιστορικό του κάθε ασθενή ανά πάσα στιγμή.

Λέξεις κλειδιά: Λειτουργικό Σύστημα Android, ΕΚΓ, Απεικόνιση, Βάση Δεδομένων SQL, TCP, Δικτυακός Προγραμματισμός

Abstract

The goal of this thesis is the design and development of software for visualization and storage of an electrocardiogram signal. The development will be done in the android environment, using java as programming language. The Android environment allows a functional and graphical visualization of the biosignal, easy to use for the doctor and mainly to help him to draw conclusions. It's important for the doctor to have a constant monitoring of his patients and to be informed about their health progress.

The message will be wirelessly transmitted to the application through the use of TCP sockets, that allows, the transferring and, whether on a local network, the Internet or via the mobile communications network.

Finally, the biosignal will be stored inside an SQL Database containing the patients and records of their biosignals designed to be easily retrievable, so the doctor may be able to refer to the history of each patient any moment.

Key words: AndroidOS, ECG, Visualization, SQL Database, TCP, Socket Programming

Ευχαριστίες

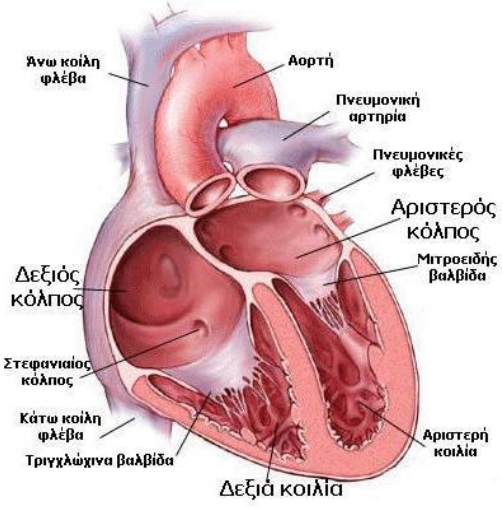
Θα ήθελα να εκφράσω τις ευχαριστίες μου στους καθηγητές Γεώργιο Ματσόπουλο και Παντελή Ασβεστά για την ευκαιρία που μου έδωσαν να ασχοληθώ με το παρόν θέμα καθώς και την καθοδήγηση που μου προσέφεραν για την ολοκλήρωση αυτής της εργασίας.

Πίνακας Περιεχομένων

1 Εισαγωγή

1.1 Η φυσιολογία του μυοκαρδίου

Η καρδιά είναι ένα μυώδες όργανο που συστέλλεται ρυθμικά λειτουργώντας σαν αντλία αίματος με τη βοήθεια του κυκλοφορικού συστήματος. Ο καρδιακός μυς παρουσιάζει σημαντικές αναλογίες με τους γραμμωτούς μύες. Μία σημαντική διαφορά, που τον κάνει να ξεχωρίζει, είναι ότι στον καρδιακό μυ οι κυτταρικές μεμβράνες γειτονικών κυττάρων συγχωνεύονται και δημιουργούν ένα ενιαίο μόρφωμα, γεγονός που έχει σαν αποτέλεσμα τη συστολή μεγάλου αριθμού μυϊκών ινών σαν ένα σύνολο. Εξαιτίας αυτής της ιστολογικής κατασκευής, λέμε ότι ο καρδιακός μυς αποτελεί ένα λειτουργικό σύνολο. Ο ερεθισμός έστω και μίας μυοκαρδιακής ίνας οδηγεί σε εξάπλωση του δυναμικού δράση σε ολόκληρη τη μυϊκή μάζα. Ο καρδιακός μυς περικλείεται από έναν ινώδη σάκο που λέγεται περικάρδιο. Το εσωτερικό της καρδίας καλύπτεται από μία σκληρή μεμβράνη, το ενδοκάρδιο.
Η καρδία αποτελείται από τρεις μείζονες τύπος μυοκαρδίου: το μυοκάρδιο των κόλπων, το μυοκάρδιο των κοιλιών και τις εξειδικευμένες μυϊκές ίνες διέγερσης και αγωγής της διέγερσης. Το μυοκάρδιο των κόλπων και των κοιλίων συστέλλονται με τον ίδιο σχεδόν τρόπο όπως και ο σκελετικός μυς, με τη διαφορά ότι η διάρκεια συστολής τους είναι πολύ μεγαλύτερη. Εξάλλου, οι εξειδικευμένες μυϊκές ίνες διέγερσης και αγωγής της διέγερσης, ελάχιστα μόνο συστέλλονται γιατί περιέχουν ελάχιστα μόνο συσταλτά ινίδια. Αντίθετα, εξαιτίας της ιδιότητας της ρυθμικής τους λειτουργίας και της μεγάλης ταχύτητας αγωγής της διέγερσης, συγκροτούν ένα σύστημα για τη διέγερση της καρδίας, καθώς και ένα σύστημα αγωγής του σήματος για τη διέγερση ολόκληρου του μυοκαρδίου.[1]



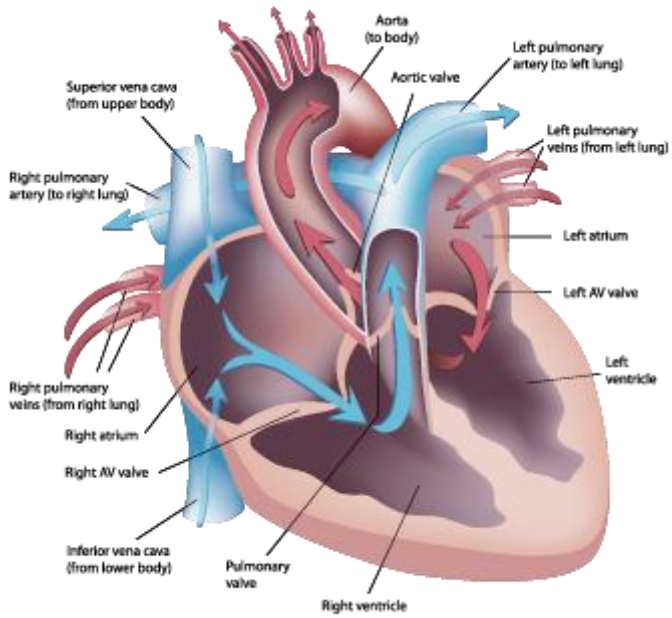
Σχήμα 1.1 Ανατομία της καρδιάς

1.2 Μέγεθος και σχήμα της καρδιάς

Κατά τη γέννηση η καρδιά είναι εγκάρσια (πλατιά) και εμφανίζεται μεγάλη σε αναλογία με τη διάμετρο της κοιλότητας του στήθους. Στα βρέφη η καρδιά καταλαμβάνει το 1/130 του συνολικού βάρους του σώματος, σε σύγκριση με το 1/300 στους ενήλικες. Από την εφηβεία και μέχρι τα 25 χρόνια, η καρδιά παίρνει το κανονικό σχήμα και βάρος περίπου 310g για τους άνδρες και 225g για τις γυναίκες. Στον ενήλικα το σχήμα της καρδιάς τείνει να μοιάσει σε αυτό του στήθους. Στα ψηλά και αδύνατα άτομα η καρδιά συχνά περιγράφεται σαν επιμηκυμένη, σε αντίθεση με τα κοντά και πιο ογκώδη άτομα όπου έχει μεγαλύτερο πλάτος και περιγράφεται σαν εγκάρσια. Στα άτομα που έχουν μέσο ύψος και βάρος η καρδιά μπορεί να μην είναι ούτε μακριά ούτε εγκάρσια αλλά κάτι ενδιάμεσο. Η προσεγγιστικές διαστάσεις είναι: μήκος 12cm, πλάτος 9cm, ύψος 6cm. Το σχήμα 1-2 δείχνει λεπτομέρειες της καρδιάς και των κυριότερων αγγείων από την εμπρόσθια και οπίσθια όψη.

1.3 Περιγραφή καρδιακής λειτουργίας

Η καρδιά αποτελείται από 4 ξεχωριστές αντλίες: δύο προαντλίες, τους κόλπους, και δύο προωθητικές αντλίες, τις κοιλίες (σχήμα 2). Η χρονική περίοδος από το τέλος μιας συστολής της καρδιάς μέχρι το τέλος της επόμενης συστολής, ονομάζεται καρδιακός παλμός (ή καρδιακός κύκλος). Ο κάθε καρδιακός παλμός αρχίζει με την αυτόματη γένεση ενός δυναμικού ενέργειας στον φλεβόκομβο. Αυτός ο κόμβος εντοπίζεται στο πρόσθιο τμήμα του δεξιού κόλπου, κοντά στην εκβολή της άνω κοίλης φλέβας, το δε δυναμικό ενέργειας επεκτείνεται με ταχύτητα και στους δυο κόλπους και από εκεί μέσα από το κολποκοιλιακό δεμάτιο προς τις κοιλίες. Όμως, εξαιτίας της ειδικής διαρρύθμισης του συστήματος αγωγής από τους κόλπους στις κοιλίες, παρατηρείται καθυστέρηση κάπως μεγαλύτερη από 1/10 sec για τη δίοδο της διέγερσης από τους κόλπους στις κοιλίες. Με αυτόν τον τρόπο παρέχεται στους κόλπους η ευκαιρία να συστέλλονται πριν από τις κοιλίες, με αποτέλεσμα την άντληση αίματος προς τις κοιλίες πριν από την έντονη κοιλιακή συστολή. Έτσι οι κόλποι λειτουργούν σαν εναυσματικές αντλίες για τις κοιλίες, οι οποίες με τη σειρά τους παρέχουν την κύρια πηγή της δύναμης για την προώθηση του αίματος μέσα από το αγγειακό σύστημα.



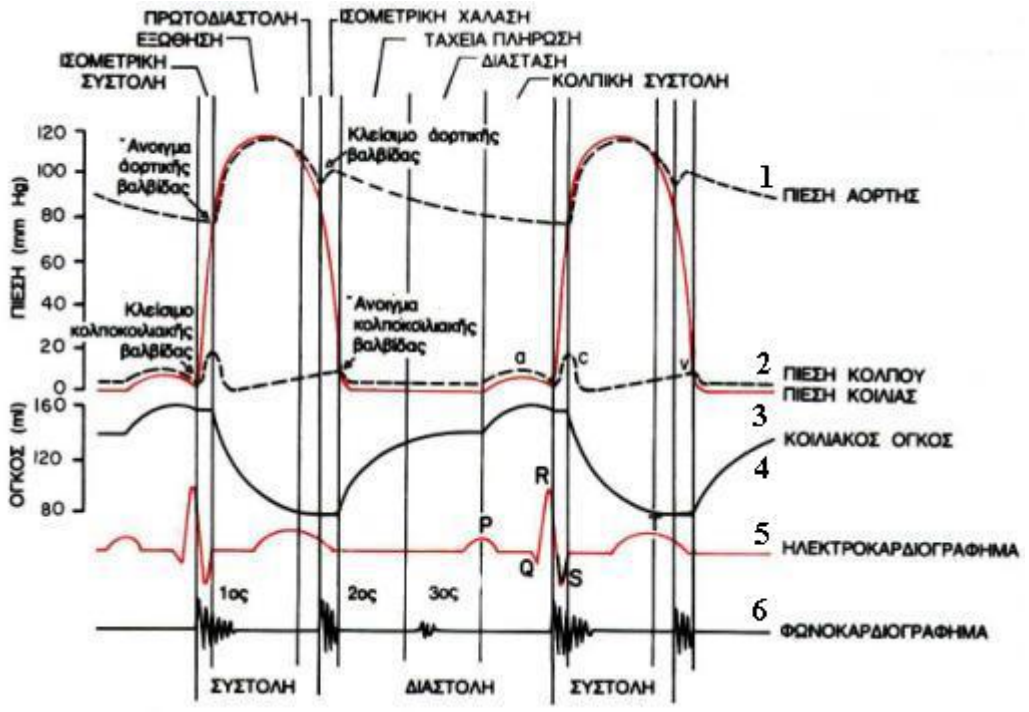
Σχήμα 1.2: Ανατομία της καρδιάς και πορεία του αίματος διάμεσου των καρδιακών κοιλοτήτων.

1.4 Καρδιακός Κύκλος

Κάθε καρδιακός παλμός αρχίζει με την αυτόματη γένεση ενός δυναμικού ενέργειας στο φλεβόκομβο. Αυτός ο κόμβος εντοπίζεται στο πρόσθιο τμήμα του δεξιού κόλπου , κοντά στην εκβολή της άνω κοίλης φλέβας, το δε δυναμικό ενέργειας επεκτείνεται με ταχύτητα και στους δύο κόλπους, και από εκεί, μέσα από το κολποκοιλιακό δεμάτιο, προς τις κοιλίες. Όμως, εξαιτίας της ειδικής διαρρύθμισης του συστήματος αγωγής από τους κόλπους στις κοιλίες, παρατηρείται καθυστέρηση, κάπως μεγαλύτερη από 0.1 sec για τη δίοδο της διέγερσης από τους κόλπους στις κοιλίες. Με αυτόν τον τρόπο παρέχεται στους κόλπους η ευκαιρία να συστέλλονται πριν από τις κοιλίες, με αποτέλεσμα την άντληση αίματος προς τις κοιλίες πριν από την έντονη κοιλιακή συστολή. Έτσι οι κόλποι λειτουργούν σαν εναυσματικές αντλίες για τις κοιλίες, οι οποίες με τη σειρά τους παρέχουν την κύρια πηγή της δύναμης για την προώθηση του αίματος μέσα από το αγγειακό σύστημα.

Ο καρδιακός παλμός αποτελείται από μια περίοδο χαλάρωσης που ονομάζεται διαστολή, κατά τη διάρκεια της οποίας η καρδιά γεμίζει με αίμα, η οποία ακολουθείται από περίοδο συστολής, που ονομάζεται συστολή. Στο σχήμα 2 παριστάνονται τα γεγονότα τα οποία λαμβάνουν χώρα κατά τη διάρκεια του

καρδιακού παλμού. Στις άνω τρεις καμπύλες (1,2,3) παριστάνονται οι μεταβολές της πίεσης μέσα στην αορτή, μέσα στην αριστερή κοιλία και μέσα στον αριστερό κόλπο αντίστοιχα. Η τέταρτη καμπύλη (4) παριστάνει τις μεταβολές του όγκου των κοιλιών, η Πέμπτη (5) είναι το ηλεκτροκαρδιογράφημα και η έκτη (6) είναι το καρδιοφωνογράφημα το οποίο αποτελεί καταγραφή των ήχων που παράγονται από την καρδιά κατά την αντλητική της λειτουργία. Είναι ιδιαίτερα σημαντικό να μελετήσει κανείς με μεγάλη προσοχή και με λεπτομέρεια αυτό το διάγραμμα και να κατανοήσει τα αίτια όλων των φαινομένων που καταγράφονται σ' αυτό. Μετά από 0,16 sec περίπου από την έναρξη του επάρματος P, εμφανίζονται τα επάρματα QRS τα οποία οφείλονται στην εκπόλωση των κοιλιών, η οποία προκαλεί την έναρξη της συστολής των κοιλιών και την ανιούσα φορά της ενδοκοιλιακής πίεσης, όπως απεικονίζεται στο σχήμα 2. Κατά συνέπεια, το σύμπλεγμα QRS αρχίζει ελάχιστο χρόνο πριν από τη συστολή των κοιλιών. Τέλος, παρατηρείται στο ηλεκτροκαρδιογράφημα το κοιλιακό έπαρμα Τ. Αυτό αντιπροσωπεύει την περίοδο επαναπόλωσης των κοιλιών κατά τη διάρκεια της οποίας οι μυϊκές ίνες του μυοκαρδίου των κοιλιών αρχίζουν να χαλαρώνουν. Γι' αυτό και το έπαρμα Τ εμφανίζεται ελάχιστο χρονικό διάστημα πριν από το τέλος της συστολής των κοιλιών.[1]



Σχήμα 1.3 Τα φαινόμενα του καρδιακού κύκλου, με τις μεταβολές της πίεσης στον αριστερό κόλπο, την αριστερή κοιλία και την αορτή (1,2,3), τις μεταβολές του όγκου των κοιλιών (4), το ηλεκτροκαρδιογράφημα (5) και το φωνοκαρδιογράφημα (6).

2 Το ηλεκτροκαρδιογράφημα

2.1 Γενικά

Το ΗΚΓ καταγράφει την ηλεκτρική δραστηριότητα της καρδιάς, όπου κάθε χτύπος της απεικονίζεται ως μια συγκεκριμένη ηλεκτρική κυματομορφή. Ένα ΗΚΓ παρέχει δύο ειδών πληροφορίες: α) τη διάρκεια του ηλεκτρικού κύματος που διαπερνά την καρδιά, η οποία με τη σειρά της καθορίζει το κατά πόσο η ηλεκτρική δραστηριότητα είναι φυσιολογική, αργή ή προβληματική και β) την ισχύ του ηλεκτρικού κύματος που διαπερνά τον καρδιακό μυ, με την οποία μπορεί να αξιολογηθεί η κατάσταση των διαφόρων τμημάτων της καρδιάς αλλά και να διαγνωστεί κατά πόσο η καρδιά είναι καταπονημένη.

Κατά την επέκταση του επάρματος της διέγερσης στα διάφορα τμήματα της καρδιάς, ηλεκτρικά ρεύματα διατρέχουν τους ιστούς γύρω από την καρδιά, ένα μικρό δε μέρος από αυτά φτάνει μέχρι την επιφάνεια του σώματος. Εάν τοποθετηθούν ηλεκτρόδια πάνω στο δέρμα από τη μια και την άλλη πλευρά της καρδιάς, καθίσταται δυνατή η καταγραφή των ηλεκτρικών δυναμικών που παράγονται από αυτήν. Η καμπύλη που λαμβάνεται με αυτόν τον τρόπο ονομάζεται ηλεκτροκαρδιογράφημα. Η επεξεργασία του ΗΚΓ γίνεται με σκοπό την παρουσίαση της καρδιακής δραστηριότητας με όσο το δυνατό λεπτομερέστερο σήμα. Το αντίστροφο πρόβλημα στην καρδιολογία ορίζεται ως η μέγιστη δυνατότητα εντοπισμού της ηλεκτρικής δραστηριότητας του φαινομένου που λαμβάνει χώρα τη συγκεκριμένη χρονική στιγμή. Αν και κάτι τέτοιο είναι πολύ δύσκολο λόγω της υπερβολικής πολυπλοκότητας που παρουσιάζει το νευρικό δένδρο αγωγής της καρδιάς, μέχρι σήμερα έχουν αναπτυχθεί και εφαρμοστεί με επιτυχία αρκετά είδη καρδιογράφων που εξασφαλίζουν ικανοποιητικά την παραπάνω συνθήκη.

Σχήμα 2.1 Ηλεκτροκαρδιογράφημα

2.2 Το Φυσιολογικό Ηλεκτροκαρδιογράφημα

Φυσιολογικά, το εύρος συχνοτήτων του σήματος ενός ΗΚΓ κυμαίνεται από 0.05 έως 100 Hz και το δυναμικό του από 1 έως 10 mV. Το ΗΚΓ χαρακτηρίζεται από 5 κορυφές και κοιλίες, οι οποίες επισημαίνονται με τα γράμματα P, Q, R, S, T. Σε κάποιες περιπτώσεις χρησιμοποιείται μια επιπλέον κορυφή, που ονομάζεται U. Η απόδοση ενός συστήματος ανάλυσης του σήματος ενός ΗΚΓ εξαρτάται κυρίως από την ακρίβεια και την αξιοπιστία στον προσδιορισμό του συμπλέγματος QRS, καθώς και των T και P κυματομορφών. Ο επακριβής προσδιορισμός του συμπλέγματος QRS είναι το σημαντικότερο ζητούμενο στην ανάλυση του σήματος ενός ΗΚΓ. Από τη στιγμή που θα προσδιοριστεί η κυματομορφή QRS, το ΗΚΓ έγκειται σε περεταίρω επεξεργασία για την εξαγωγή των αντίστοιχων συμπερασμάτων.

Η ηλεκτρική τάση των κυμάτων στο φυσιολογικό ηλεκτροκαρδιογράφημα εξαρτάται από τον τρόπο με τον οποίο τα ηλεκτρόδια τοποθετούνται στην επιφάνεια του σώματος. Όταν το ένα ηλεκτρόδιο τοποθετείται αμέσως πάνω από την καρδιά, και το δεύτερο ηλεκτρόδιο τοποθετείται σε κάποιο άλλο σημείο του σώματος, η ηλεκτρική τάση του συμπλέγματος QRS μπορεί να φτάνει τα 3 ή 4 mV. Αλλά ακόμη και αυτή η τάση είναι πολύ μικρή σε σύγκριση με το μονοφασικό δυναμικό ενέργειας των 110 mV, όπως καταγράφεται, με άμεσο τρόπο, από την κυτταρική μεμβράνη μυϊκής ίνας του μυοκαρδίου. Όταν το ηλεκτροκαρδιογράφημα καταγράφεται με ηλεκτρόδια τοποθετημένα στα δυο άνω άκρα, είτε σε ένα άνω και σε ένα κάτω άκρο, η ηλεκτρική τάση του συμπλέγματος QRS είναι συνήθως 1 mV από την κορυφή του επάρματος R μέχρι το κάτω μέρος του επάρματος S. Εξάλλου η ηλεκτρική τάση του επάρματος P είναι 0,1 ως 0,3 mV και του επάρματος T από 0,2 ως 0,3 mV.

Στο φυσιολογικό φλεβοκομβικό ρυθμό (κανονική κατάσταση της καρδιάς) το διάστημα P-R είναι της τάξης των 0.12 έως 0.2 sec. Το διάστημα QRS κυμαίνεται από 0.04 έως 0.12 sec. Το διάστημα Q-T είναι μικρότερο των 0.42 sec, ενώ οι καρδιακοί παλμοί υπό φυσιολογικές συνθήκες κυμαίνονται από 60 έως 100 ανά λεπτό. Έτσι, από τη μορφή του ΗΚΓ μπορούμε με ευκολία να πούμε αν η καρδιακή δραστηριότητα είναι φυσιολογική ή όχι.

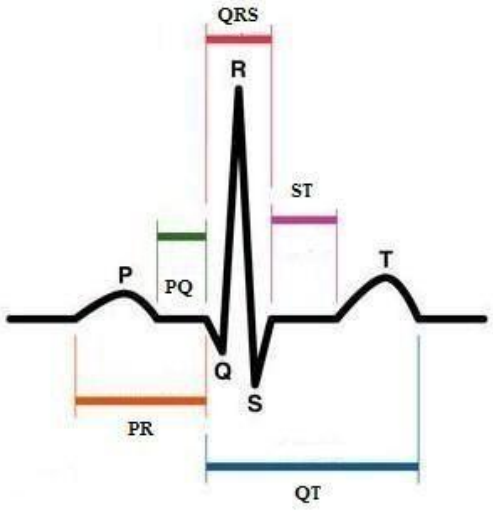
Παρακάτω δίνονται οι ορισμοί για δύο χαρακτηριστικά διαστήματα που απαντώνται στο ΗΚΓ:

Το διάστημα P-Q ή P-R. Το χρονικό διάστημα που μεσολαβεί μεταξύ του επάρματος P και της αρχής του συμπλέγματος QRS είναι ο χρόνος που παρέρχεται από την έναρξη της συστολής των κόλπων μέχρι την έναρξη της συστολής των κοιλιών. Το χρονικό αυτό διάστημα ονομάζεται διάστημα P-Q. Το φυσιολογικό

διάστημα P-Q είναι περίπου 0,16 sec. Αυτό το διάστημα σε μερικές περιπτώσεις ονομάζεται διάστημα P-R γιατί το Q συχνά απουσιάζει.

Το διάστημα Q-T. Η συστολή των κοιλιών πρακτικά διαρκεί από την αρχή του επάρματος Q μέχρι το τέλος του επάρματος T. Το χρονικό αυτό διάστημα ονομάζεται διάστημα Q-T και η φυσιολογική του διάρκεια είναι 0,35 sec.

Η συχνότητα της καρδιακής λειτουργίας μπορεί να καθορισθεί εύκολα από το ηλεκτροκαρδιογράφημα, γιατί το χρονικό διάστημα που παρεμβάλλεται μεταξύ δυο διαδοχικών καρδιακών παλμών είναι το αντίστροφο της καρδιακής συχνότητας. Εάν το χρονικό διάστημα που παρεμβάλλεται μεταξύ δυο διαδοχικών καρδιακών παλμών, όπως καθορίζεται με τις γραμμές βαθμονόμησης, είναι 1 sec, η καρδιακή συχνότητα είναι 60 καρδιακοί παλμοί το λεπτό. Το φυσιολογικό χρονικό διάστημα που παρεμβάλλεται μεταξύ δυο συμπλεγμάτων QRS είναι περίπου 0,83 sec. Αυτό σημαίνει ότι η καρδιακή συχνότητα σ' αυτή την περίπτωση είναι 72 καρδιακοί παλμοί το λεπτό.

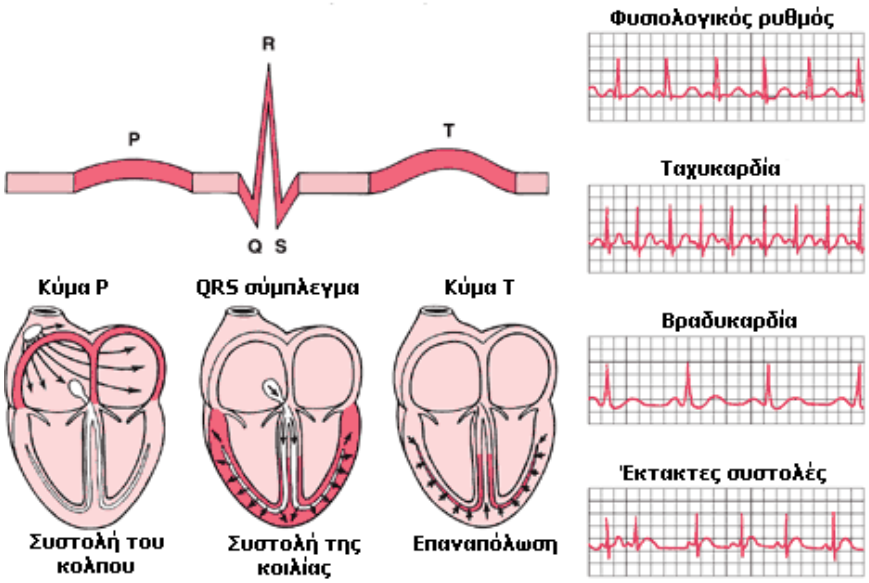


Σχήμα 2.2 Επάρματα και Συμπλέγματα του Ηλεκτροκαρδιογραφήματος

Έπαρμα	Πλάτος
P-wave	0.25mV
R-wave	1.60mV
Q-wave	25% R-wave
T-wave	0.1-0.5mV

Έπαρμα/Σύμπλεγμα	Διάρκεια
P-R	0.12-0.20sec
Q-T	0.35-0.44sec
S-T	0.05-0.15sec
P-wave	0.11sec
QRS	0.09sec

Η φυσιολογική τιμή του καρδιακού παλμού κυμαίνεται μεταξύ 60 και 100 παλμοί ανά λεπτό. Χαμηλότερη τιμή από αυτή ονομάζεται βραδυκαρδία, ενώ υψηλότερη τιμή από αυτή ονομάζεται ταχυκαρδία. Αν η περίοδος του σήματος του καρδιογραφήματος, δηλαδή η χρονική απόσταση μεταξύ δύο καρδιακών παλμών δεν είναι σταθερή, τότε έχουμε το φαινόμενο της αρρυθμίας. Ανάλογα φαινόμενα μπορούν να εντοπιστούν από τη μελέτη ενός ΗΚΓ, και αυτός είναι ο λόγος που το ΗΚΓ χρησιμοποιείται ευρέως στη διάγνωση ποικίλων ανωμαλιών και καταστάσεων που σχετίζονται με την καρδιά.



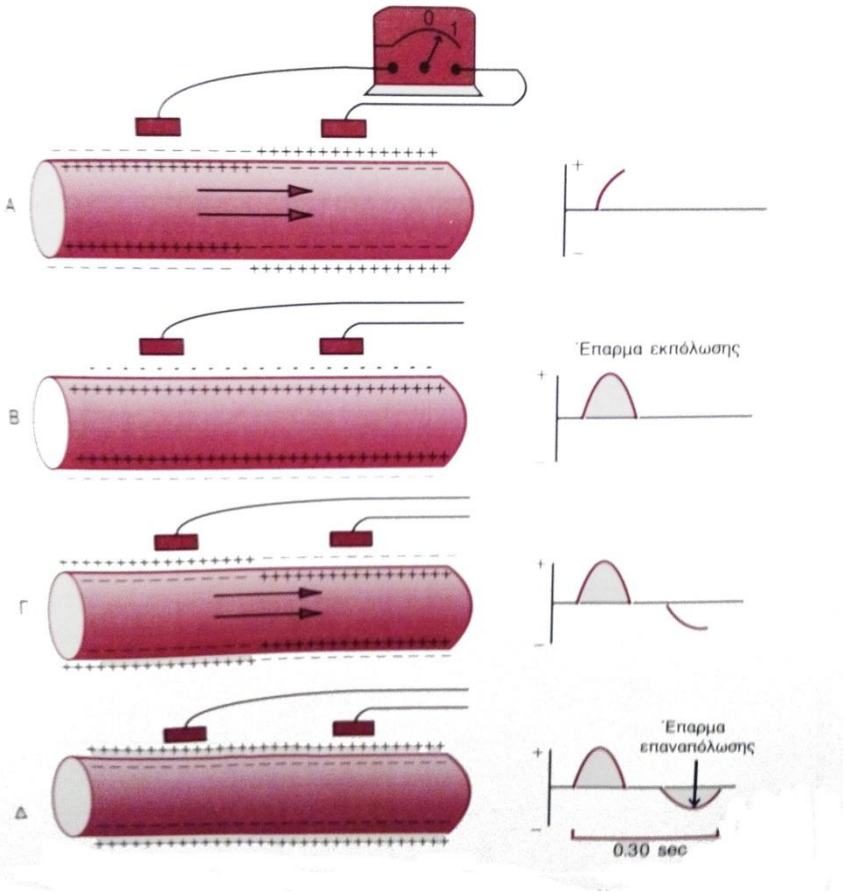
Σχήμα 1.6 Καρδιακός κύκλος και ηλεκτροκαρδιογράφημα

Ορισμένες καρδιακές διαταραχές, συμπεριλαμβανομένων και εκείνων που αφορούν τις καρδιακές βαλβίδες, δεν μπορούν να διαγνωστούν από το ΗΚΓ. Άλλες διαγνωστικές τεχνικές, όπως η αγγειογραφία και το υπερηχοκαρδιογράφημα είναι σε θέση να μας δώσουν τις πληροφορίες που το ΗΚΓ αδυνατεί.

Ουσιαστικά, το ηλεκτροκαρδιογράφημα είναι μια γραφική απεικόνιση της τάσης που παράγει το μυοκάρδιο κατά τη διάρκεια ενός καρδιακού κύκλου, στο πεδίο του χρόνου. Οι P, QRS και T κυματομορφές αντανακλούν την ρυθμική ηλεκτρική εκπόλωση και επαναπόλωση του μυοκαρδίου, που σχετίζεται με τη συστολή των κόλπων και των κοιλιών.

2.3 Εκπόλωση - Επαναπόλωση

Το ηλεκτροκαρδιογράφημα αποτελείται τόσο από επάρματα εκπόλωσης, όσο και από επάρματα επαναπόλωσης. Η διάκριση μεταξύ των κυμάτων εκπόλωσης και επαναπόλωσης θεωρείται πολύ σημαντική στην ηλεκτροκαρδιογραφία, η περαιτέρω διευκρίνιση εδώ κρίνεται απαραίτητη.



Σχήμα 2.3: Καταγραφή του επάρματος εκπόλωσης και του επάρματος επαναπόλωσης από μια μυϊκή ίνα μυοκαρδίου. (Α) Στάδιο επέκτασης εκπόλωσης στο μισό τμήμα της ίνας. (Β) Στάδιο επέκτασης εκπόλωσης σε όλο το μήκος της ίνας. (Γ) Στάδιο επαναπόλωσης στο μισό τμήμα της ίνας. (Δ) Στάδιο επαναπόλωσης σε όλο το μήκος της ίνας.

Στο σχήμα 2.3 απεικονίζεται μια μυϊκή ίνα σε τέσσερα διαφορετικά στάδια εκπόλωσης και επαναπόλωσης. Κατά τη διεργασία της «εκπόλωσης» το φυσιολογικό αρνητικό δυναμικό στο εσωτερικό της ίνας παύει να υπάρχει, το δε δυναμικό της μεμβράνης στην πραγματικότητα αντιστρέφεται, δηλαδή γίνεται ελαφρά θετικό στο εσωτερικό της ίνας, και αρνητικό στην εξωτερική της επιφάνεια

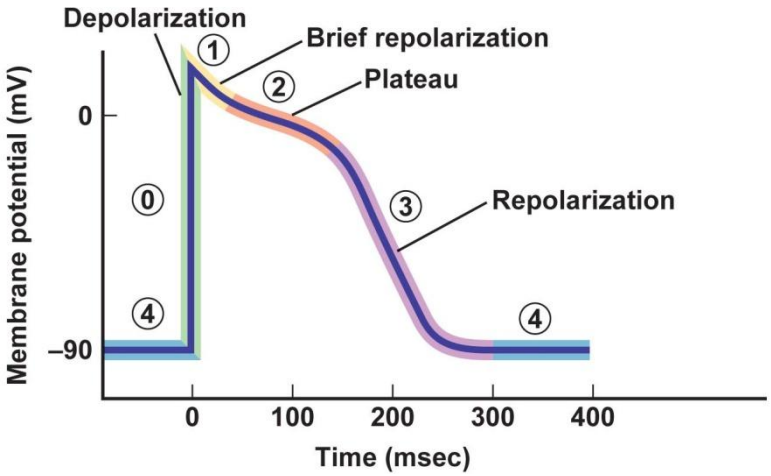
Στο σχήμα 2.3-Α, η διεργασία της εκπόλωσης, που απεικονίζεται με τα θετικά φορτία στο εσωτερικό και τα αρνητικά φορτία στην εξωτερική επιφάνεια, επεκτείνεται από τα αριστερά στα δεξιά, το δε πρώτο μισό τμήμα της ίνας έχει ήδη υποστεί εκπόλωση, ενώ το υπόλοιπο μισό διατηρεί ακόμη την πόλωσή του. Κατά συνέπεια, το αριστερό ηλεκτρόδιο πάνω στην ίνα βρίσκεται σε περιοχή αρνητικότητας όταν έρχεται σε επαφή με το εξωτερικό της ίνας, ενώ το δεξιό ηλεκτρόδιο βρίσκεται σε περιοχή θετικότητας. Αυτό καταγράφεται ως θετική απόκλιση. Στο δεξιό άκρο του σχήματος απεικονίζεται η καμπύλη του δυναμικού, όπως καταγράφεται από όργανο με μεγάλη ταχύτητα καταγραφής, όπως είναι σε αυτό το συγκεκριμένο στάδιο εκπόλωσης. Σημειώνεται ότι όταν η εκπόλωση φτάσει στο μέσο της ίνας, το παρατηρούμενο δυναμικό βρίσκεται στη μέγιστη θετική του τιμή.

Στο σχήμα 2.3-Β η εκπόλωση έχει επεκταθεί σε ολόκληρη τη μυϊκή ίνα, η δε καμπύλη προς τα δεξιά έχει επανέλθει στη μηδενική βασική γραμμή, επειδή και τα δυο ηλεκτρόδια βρίσκονται τώρα σε περιοχές ίσης αρνητικότητας μεταξύ τους. Το συμπληρωμένο αυτό έπαρμα είναι έπαρμα εκπόλωσης, γιατί προκαλείται από την επέκταση της διέγερσης σε ολόκληρο το μήκος της μυϊκής ίνας.

Στο σχήμα 2.3-Γ απεικονίζεται η διεργασία επαναπόλωσης της μυϊκής ίνας, η οποία έχει ήδη προχωρήσει ως το μέσο της ίνας, από αριστερά προς τα δεξιά. Στο σημείο αυτό, το αριστερό ηλεκτρόδιο βρίσκεται σε επαφή με περιοχή θετικότητας, ενώ το δεξιό ηλεκτρόδιο βρίσκεται σε περιοχή αρνητικότητας. Η πολικότητα των δυο ηλεκτροδίων είναι ακριβώς η αντίθετη από εκείνη του σχήματος 2.3-Α. Γι’ αυτό και η καμπύλη του δυναμικού, όπως απεικονίζεται στα δεξιά, γίνεται αρνητική.

Τελικά στο σχήμα 2.3-Δ, η μυϊκή ίνα έχει πλήρως επαναπολωθεί, και τα δυο ηλεκτρόδια βρίσκονται σε επαφή με περιοχές θετικότητας, με αποτέλεσμα να μην υφίσταται πια διαφορά δυναμικού μεταξύ τους. Έτσι, στην καμπύλη προς τα δεξιά του σχήματος, το δυναμικό επανέρχεται και πάλι σε μηδενικό επίπεδο. Το συμπληρωμένο αυτό αρνητικό έπαρμα χαρακτηρίζεται ως έπαρμα επαναπόλωσης, επειδή προκαλείται από την επέκταση της διεργασίας επαναπόλωσης πάνω στη μυϊκή ίνα.

Το μονοφασικό δυναμικό ενέργειας του μυοκαρδίου των κοιλιών διαρκεί φυσιολογικά από 0.25 ως 0.35 sec. Στο άνω μέρος του σχήματος 2.4 απεικονίζεται μονοφασικό δυναμικό ενέργειας, το οποίο έχει καταγραφεί με μικροηλεκτρόδιο που έχει εισαχθεί μέσα σε μυϊκή ίνα του μυοκαρδίου των κοιλιών. Η σχεδόν κάθετη προς τα άνω γραμμή του δυναμικού ενέργειας προκαλείται από την εκπόλωση, η δε επάνοδος του δυναμικού στη βασική γραμμή προκαλείται από την επαναπόλωση. [1]



Σχήμα 2.4: Η εκπόλωση (depolarization) είναι γρήγορη, ενώ η επαναπόλωση (repolarization) είναι αργή κατά τη φάση του επιπέδου (plateau) της καμπύλης αλλά πολύ γρήγορη προς το τέλος του.

Σημειώνεται ιδιαίτερα το γεγονός ότι στο ηλεκτροκαρδιογράφημα δεν αναγράφεται καθόλου δυναμικό όταν το μυοκάρδιο των κοιλιών διατηρεί πλήρως την πόλωσή του είτε όταν βρίσκεται σε πλήρη εκπόλωση. Μόνο όταν το μυοκάρδιο είναι κατά ένα μέρος μόνο σε κατάσταση πόλωσης και κατά το άλλο μέρος σε κατάσταση εκπόλωσης παρατηρείται ροή ηλεκτρικού ρεύματος από ένα μέρος των κοιλιών σε άλλο και κατά συνέπεια παρατηρείται και ροή του στην επιφάνεια του σώματος, οπότε και προκαλείται η γένεση των κυμάτων του ηλεκτροκαρδιογραφήματος.

2.4 Σχέση ΗΚΓ με συστολή – διαστολή κόλπων και κοιλίων

Πριν να είναι δυνατή η συστολή του μυός, είναι απαραίτητη η επέκταση της εκπόλωσης στο μυοκάρδιο, για την έναρξη των χημικών διεργασιών της συστολής. Το έπαρμα P προκαλείται από την επέκταση της εκπόλωσης στους κόλπους, τα δε επάρματα QRS από την επέκταση της εκπόλωσης στις κοιλίες. Κατά συνέπεια, το έπαρμα P εμφανίζεται αμέσως πριν από την έναρξη της συστολής των κόλπων, ενώ το σύμπλεγμα QRS αμέσως πριν από την έναρξη της συστολής των κοιλιών. Οι κοιλίες παραμένουν σε κατάσταση συστολής για μερικά χιλιοστά του sec μετά την επαναπόλωση, δηλαδή μετά το τέλος του επάρματος T.

Το έπαρμα επαναπόλωσης των κοιλιών, στο φυσιολογικό ηλεκτροκαρδιογράφημα, είναι το έπαρμα T. Φυσιολογικά, ορισμένες μυϊκές ίνες του

μυοκαρδίου των κοιλιών αρχίζουν να επαναπολώνονται 0,2 sec περίπου μετά την έναρξη του επάρματος εκπόλωσης, πολλές όμως άλλες ίνες αρχίζουν να επαναπολώνονται βραδύτερα, μέχρι και 0,35 sec. Έτσι, η διεργασία της επαναπόλωσης επεκτείνεται μέσα σε σχετικά μεγάλο χρονικό διάστημα, περίπου 0,15 sec. Κατά συνέπεια, το έπαρμα Τ στο φυσιολογικό ηλεκτροκαρδιογράφημα συχνά είναι παρατεταμένο, η ηλεκτρική του όμως τάση είναι σημαντικά μικρότερη από την τάση του συμπλέγματος QRS, αυτό δε μερικώς οφείλεται στη μεγάλη του διάρκεια.

Οι κόλποι επαναπολούνται περίπου 0,15 ως 0,20 sec μετά το έπαρμα εκπόλωσης. Ο χρόνος όμως αυτός συμπίπτει με την εμφάνιση του συμπλέγματος QRS στο ηλεκτροκαρδιογράφημα. Κατά συνέπεια, το έπαρμα επαναπόλωσης των κόλπων, γνωστό ως Τ των κόλπων, συνήθως επικαλύπτεται από το πολύ μεγαλύτερο σύμπλεγμα QRS. Για αυτό το λόγο, σπάνια μόνο είναι δυνατόν να παρατηρηθεί κολπικό έπαρμα Τ στο ηλεκτροκαρδιογράφημα. [1]

2.5 Τηλεϊατρική

Οι τεχνολογίες επικοινωνίας μας προσφέρουν τα μέσα για να μεσολαβήσει η ανθρώπινη επικοινωνία όταν η πρόσωπο με πρόσωπο αλληλεπίδραση δεν είναι δυνατή. Η τεχνολογία γεφυρώνει τα κενά της επικοινωνίας που καθιστούν την περίθαλψη ανεπαρκής και επικίνδυνη.

Οι κλινικές τεχνολογίας επικοινωνίας συχνά αναγνωρίζονται με τον όρο τηλεϊατρική. Όροι όπως τηλεϋγεία ή τηλεφροντίδα χρησιμοποιούνται σαν εναλλακτικές με σκοπό να ξεκαθαρίσουν ότι δεν εμπλέκονται μόνο γιατροί αλλά και παραϊατρικό προσωπικό. Για πολλούς η τηλεϋγεία έχει έναν πιο συγκεκριμένο ορισμό και περιέχει την χρήση βιντεοδιάσκεψης για να παρέχει διαβουλεύσεις και περίθαλψη από απόσταση. Αρχικά η τηλεϋγεία είχε σκοπό να παρέχει τέτοιου είδους συνδέσεων μέσω βίντεο μεταξύ ιατρικών εμπειρογνωμόνων και ασθενείς σε απομακρυσμένες περιοχές. Δυστυχώς, αυτοί οι τεχνολογικοί ορισμοί περιορίζουν την σκέψη μας και δεν σκεφτόμαστε για τα πιθανά προβλήματα που χρειάζονται επίλυση στον χώρο της επικοινωνίας. Η υγειονομική περίθαλψη έχει πολλές και ποικίλες επικοινωνιακές ανάγκες και η χρήση της τεχνολογίας βίντεο είναι μία μόνο λύση που πρέπει να διερευνηθεί. Κάποιες λύσεις στα προβλήματα επικοινωνίας δεν είναι τεχνολογικές αλλά απαιτούν αλλαγές στον άνθρωπο ή στις διαδικασίες μέσα στην οργάνωση.

Η τηλεϋγεία γίνεται καλύτερα κατανοητή ως οποιαδήποτε επικοινωνία με χρήση της τεχνολογίας η οποία διευκολύνει την κλινική φροντίδα όπου η ανταλλαγή της πληροφορίας συμβαίνει όλο τον χρόνο και ανεξαρτήτως της απόστασης. Οι πληροφορίες αυτές μπορεί να είναι φωνή, εικόνα στοιχεία ιατρικού φακέλους ή εντολές σε ένα χειρουργικό ρομπότ. Δεν είναι μία νέα επιχείρηση – ο Einthoven πειραματίστηκε με την τηλεφωνική μετάδοση χρησιμοποιώντας την εφεύρεση του, τον ηλεκτροκαρδιογράφο στην αρχή του 20^{ου} αιώνα.

Η τηλεϊατρική με την χρήση κινητών συσκευών (mobile health ή m-health) ασχολείται ειδικά με την χρήση της ασύρματης επικοινωνίας, υπολογιστές και αισθητήρες για να παρέχει περίθαλψη. Τα smart phones (έξυπνα κινητά), οι φορητοί υπολογιστές και οι αισθητήρες κάνουν την παρακολούθηση του ασθενή πολύ εύκολη και συνηθισμένη όταν κάποτε διαδικασίες, όπως η παρακολούθηση ηλεκτροκαρδιογραφήματος Holter ήταν εξειδικευμένες υπηρεσίες. Κατά βάθος η πληροφορική συνδυάζεται φυσικά με την επικοινωνία σε κινητές πλατφόρμες επικοινωνίας και επιτρέπει την εύκολη αλληλεπίδραση μεταξύ κλινικών ιατρών. Άρα η m-health εντάσσεται στην ομπρέλα της τηλεϊατρικής και μπορεί να θεωρηθεί σαν εξέλιξη της τηλεϋγείας.

Η ανάπτυξη νέων τεχνολογιών επικοινωνίας που επιτρέπουν την πιο προχωρημένη ανταλλαγή πληροφοριών, όπως τα δίκτυα 4G και 5G οδηγούν την m-health στην γρήγορη ανάπτυξη. [2]

2.6 Τηλεπαρακολούθηση

Η τηλεπαρακολούθηση είναι μία τεχνική για την εξ' αποστάσεως παρακολούθηση των ασθενών, οι οποίοι δεν βρίσκονται στον ίδιο χώρο με τον υπεύθυνο παροχής ιατρικής περίθαλψης.

Σε γενικές γραμμές ο ασθενής, θα έχει μία σειρά από συσκευές (αισθητήρες) παρακολούθησης στο σπίτι και τα αποτελέσματα αυτών των συσκευών διαβιβάζονται μέσω τηλεφώνου, μέσω διαδικτύου, ή μέσω κινητής συσκευής, όπως γίνεται στη παρούσα εργασία στον υπεύθυνο παροχής ιατρικής περίθαλψης. [3]

Μερικές από τις πιο συνήθεις μετρήσεις που λαμβάνω μέσω της τηλεπαρακολούθησης αφορούν τα φυσιολογικά δεδομένα ενός ατόμου όπως πίεση αίματος, καρδιακό σφυγμό κλπ.

Τα φυσιολογικά δεδομένα που συγκεντρώνονται από ένα δίκτυο αισθητήρων μπορούν να αποθηκευτούν για ένα μεγάλο χρονικό διάστημα και μπορούν να χρησιμοποιηθούν για ιατρική έρευνα στο κεντρικό σύστημα του νοσοκομείου.

Αυτοί οι μικροί κόμβοι αισθητήρων επιτρέπουν στο άτομο μεγαλύτερη ελευθερία κινήσεων και παρέχουν μία καλύτερη ποιότητα ζωής για τα άτομα σε σύγκριση με τα κέντρα παροχής θεραπείας. Όλα αυτά γίνονται δυνατά με ένα απλό ασύρματο δίκτυο επικοινωνίας. [3] [4]

2.7 Τηλεματική

Ο όρος τηλεματική (telematique) δημιουργήθηκε από τους Γάλλους Simon Nora και Alain Minc το 1976 και υπονοεί τη σύζευξη των τηλεπικοινωνιών (telecommunication) και της πληροφορικής (informatique). Η τηλεματική (ιατρική και νοσηλευτική) σημαίνει τη χρήση τεχνολογιών τηλεπικοινωνιών και πληροφοριών για την παροχή υπηρεσιών υγείας, ανεξάρτητα από το που βρίσκονται οι επαγγελματίες υγείας, οι ασθενείς, οι φάκελοι πληροφοριών υγείας και ο εξοπλισμός. Π.χ. : Μεταφορά ιατρικών εικόνων, ανάμεσα σε κέντρα υγείας για να γίνει απομακρυσμένη διάγνωση, παροχή φροντίδας στο σπίτι του ασθενή (ηλικιωμένοι, διαβητικοί ασθενείς – κατ' οίκον φροντίδα). Συνδυάζει την τεχνολογία με τις επιστήμες υγείας θέτοντας την πρώτη στη διάθεση των δεύτερων.
Τα συστατικά της τηλεματικής είναι η ιατρική και νοσηλευτική γνώση, οι επικοινωνίες, το υλικό (σαρωτές, κάμερες, οπτικά και μαγνητικά μέσα, κάρτες ήχου, video, εκτυπωτές) και λογισμικό (πλοηγοί, λογισμικό βιντεοδιάσκεψης, ηλεκτρονική αλληλογραφία, συστήματα διαχείρισης βάσεων δεδομένων κλπ) [2] [4]

Τι οδήγησε στην ανάπτυξη της τηλεματικής;
Α) Κοινωνικές ανάγκες

1) Βελτίωση της ποιότητας των παρεχομένων ιατρικών και νοσηλευτικών υπηρεσιών υγείας.
2) Δημογραφικές αλλαγές: α. Γήρανση πληθυσμού, β. Αύξηση ατόμων με χρόνιες ασθένειες, όπως άσθμα, διαβήτης και καρδιακές παθήσεις, γ. Αύξηση ατόμων που δεν μπορούν να μετακινηθούν εύκολα (ηλικιωμένοι).
3) Έλλειψη ιατρικού και νοσηλευτικού προσωπικού σε απομονωμένες – απομακρυσμένες περιοχές.
4) Ισότιμη παροχή υπηρεσιών υγείας σε απομονωμένες – απομακρυσμένες περιοχές, σε σχέση με τα αστικά κέντρα.
5) Πρωτοβάθμια φροντίδα, κοινοτική φροντίδα και κατ' οίκον φροντίδα

Β) Οικονομικές ανάγκες

1) Αυξημένο κόστος παροχής υπηρεσιών υγείας (π.χ. χρόνιοι ασθενείς, άτομα με ειδικές ανάγκες και ηλικιωμένοι).
2) Κόστος εξελιγμένου ιατρικού / νοσηλευτικού εξοπλισμού.

Γ) Δυνατότητες

Η Τηλεματική εκμεταλλεύεται τόσο τις δυνατότητες της πληροφορικής όσο και τις δυνατότητες των τηλεπικοινωνιών.

1) Πληροφορική: α. Ψηφιακή επεξεργασία – ανάλυση εικόνων και σημάτων β. Πληροφοριακά συστήματα νοσοκομείων γ. Ηλεκτρονικός φάκελος ασθενών δ. Συστήματα υποστήριξης λήψης αποφάσεων.

2) Τηλεπικοινωνίες: α. Ανάπτυξη τηλεπικοινωνιακές υποδομής σε απομακρυσμένα σημεία β. Δορυφορικές τηλεπικοινωνίες γ. Δίκτυα υπολογιστών – διαδίκτυο δ. Ασύρματα δίκτυα. [5]

3 Λογισμικό και Τεχνολογίες

3.1 Android

Το Android είναι ένα λειτουργικό σύστημα για κινητές συσκευές (που κατά κύριο λόγο διαθέτουν οθόνη αφής), όπως έξυπνα τηλέφωνα (Smartphones) και tablets, το οποίο είναι βασισμένο στον πυρήνα του Linux (Linux kernel). Επιτρέπει τη δημιουργία εφαρμογών με τη χρήση της γλώσσας Java (διαθέτοντας και έτοιμες βιβλιοθήκες). Ιδρύθηκε τον Οκτώβρη του 2003 στο Palo Alto της California και αναπτύχθηκε από τους Andy Rubin,Rich Miner, ears και Chris White. Τον Αύγουστο του 2005 το αγόρασε η Google. Η πλατφόρμα Android ανακοινώθηκε στις 5 Νοεμβρίου του 2007, παράλληλα με την ίδρυση της Open Handset Alliance (κοινοπραξία εταιριών υλικού, λογισμικού και τηλεπικοινωνιών με στόχο την προώθηση ανοιχτών προτύπων για κινητές συσκευές). Το πρώτο τηλέφωνο με λειτουργικό Android πωλήθηκε τον Νοέμβρη του 2008. [6]

3.2 Διεπαφή

Η διεπαφή χρήστη του Android βασίζεται στην αλληλεπίδραση μέσω αφής, όπως σύρσιμο, απλό άγγιγμα κλπ για το χειρισμό αντικειμένων στην οθόνη, καθώς και σε ένα εικονικό πληκτρολόγιο. Η απάντηση στην «είσοδο» από το χρήστη είναι σχεδιασμένη να είναι άμεση και να παρέχει μία ρευστή διεπαφή αγγίγματος, χρησιμοποιώντας συχνά δυνατότητες δόνησης, με σκοπό να προσφέρει απτική ανάδραση στο χρήστη. Το υλικό (hardware) της συσκευής περιέχει επιταχυνσιόμετρα, γυροσκόπια, αισθητήρες εγγύτητας κλπ, τα οποία χρησιμοποιούνται από κάποιες εφαρμογές με σκοπό να ανταποκρίνονται σε επιπρόσθετες ενέργειες του χρήστη, όπως το να προσαρμόζεται η οθόνη από «πορτραίτο» σε «τοπίο» (από κάθετα σε οριζόντια), ανάλογα με το πώς είναι προσανατολισμένη η συσκευή, ή να επιτρέπει στο χρήστη να κατευθύνει ένα όχημα σε παιχνίδι αγώνων, περιστρέφοντας την οθόνη, προσομοιώνοντας με αυτόν τον τρόπο τον έλεγχο ενός τιμονιού. [6]

3.3 Εφαρμογές

Οι εφαρμογές, οι οποίες επεκτείνουν τη λειτουργικότητα των συσκευών είναι γραμμένες πρωτίστως σε Java, χρησιμοποιώντας το Android Software Development Kit (SDK). Το SDK περιέχει ένα ολοκληρωμένο σετ από προγραμματιστικά εργαλεία, συμπεριλαμβανομένων προγράμματος εντοπισμού σφαλμάτων (debugger), βιβλιοθηκών λογισμικού (software libraries), εξομοιωτή συσκευής βασισμένο στο QEMU (Quick EMUlator), τεκμηρίωση (documentation), δειγμάτων κώδικα/ενδεικτικού κώδικα (sample code) και «φροντιστηριακού» τύπου οδηγιών (tutorials).

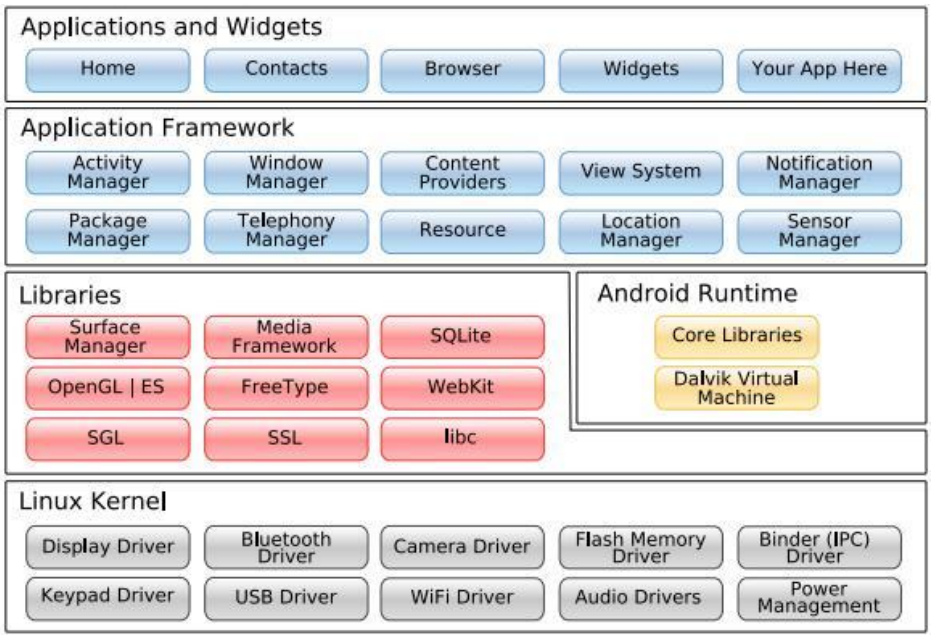
Αρχικά, το ολοκληρωμένο περιβάλλον ανάπτυξης (integrated development environment-IDE) που χρησιμοποιούνταν ήταν το Eclipse, που συμπεριλάμβανε το Android Development Tools (ADT) plugin (επιπρόσθετο πρόγραμμα). Το Δεκέμβριο του 2014 η Google κυκλοφόρησε την πρώτη σταθερή έκδοση Android Studio (1.0), βασισμένο στο IntelliJ IDEA ως βασικό IDE για την ανάπτυξη Android εφαρμογών. Επίσης, να σημειωθεί ότι είναι διαθέσιμα και άλλα εργαλεία ανάπτυξης προγραμμάτων, όπως το Native Development Kit για εφαρμογές ή επεκτάσεις σε γλώσσα C ή C++ και ποικίλα πλαίσια για εφαρμογές μέσω διαδικτύου για κινητά.

Το Android διαθέτει μεγάλη ποικιλία από εφαρμογές άλλων κατασκευαστών, οι οποίες μπορούν να αποκτηθούν από τους χρήστες «κατεβάζοντας» (download) και εγκαθιστώντας το APK (Android Application Package) αρχείο της ή «κατεβάζοντάς» τις χρησιμοποιώντας ένα πρόγραμμα «καταστήματος εφαρμογών» (application store) που επιτρέπει στους χρήστες να εγκαθιστούν, να ενημερώνουν και να διαγράφουν τις εφαρμογές από τις συσκευές τους. Το Google Play Store είναι το βασικό κατάστημα εφαρμογών και είναι εγκατεστημένο στις συσκευές που χρησιμοποιούν Android, πληρούν τις απαιτήσεις συμβατότητας της Google και χορηγούν άδεια χρήσης στο λογισμικό Google Mobile Services Software. [6]

3.4 Αρχιτεκτονική του Συστήματος

Το Android δεν είναι μόνο ένα λειτουργικό σύστημα. Είναι μια στοίβα λογισμικού η οποία αποτελείται από το λειτουργικό σύστημα, τις υπηρεσίες διασύνδεσης με τις εφαρμογές (middleware) και τέλος από τις κύριες (core) εφαρμογές, μεταξύ αυτών, ενός email client, μιας εφαρμογής διαχείρισης SMS, ενός ημερολογίου, ενός browser, μιας εφαρμογής διαχείρισης επαφών, και άλλες οι οποίες έρχονται προεγκατεστημένες με την υπόλοιπη στοιβάδα λογισμικού του Android.

Για να καταλάβει κάποιος με ποιον ακριβώς τρόπο δουλεύει το Android θα ήταν καλό να κοιτάξει και να μελετήσει την εικόνα στην οποία φαίνονται τα διαφορετικά στρώματα από τα οποία αποτελείται το λειτουργικό σύστημα του Android. Η εικόνα φαίνεται παρακάτω και έπειτα αναλύονται οι βασικοί τομείς στους οποίου χωρίζεται το λειτουργικό σύστημα. [7][8]



Σχήμα 3.1 Αρχιτεκτονική του Android

3.4.1 Linux Kernel

Ο πυρήνας (kernel) του Android είναι βασισμένος στον πυρήνα των Linux (Linux kernel) - όχι στο λειτουργικό σύστημα Linux (Linux OS). Από τον Απρίλιο του 2014, οι συσκευές Android χρησιμοποιούν κατά κύριο λόγο τις εκδόσεις 3.4 ή 3.10 του Linux kernel. Η έκδοσή του εξαρτάται από τη συσκευή Android και την πλατφόρμα υλικού της. Είναι υπεύθυνος για την ασφάλεια, τη διαχείριση μνήμης και διαχείριση διαδικασιών, τη στοίβα δικτύου και το μοντέλο προγράμματος οδήγησης. Λειτουργεί αφαιρετικά ως στρώμα μεταξύ του υλικού και της υπόλοιπης στοίβας λογισμικού.

Η αποθήκευση σε μνήμη τύπου flash (flash storage) στις συσκευές Android χωρίζεται σε διαφορετικά τμήματα, όπως /system για το λειτουργικό από μόνο του και /data για τα δεδομένα χρήστη και την εγκατάσταση εφαρμογών. Σε αντίθεση με τις εκδόσεις Linux για επιτραπέζιους/φορητούς υπολογιστές, οι χρήστες των συσκευών Android δεν έχουν πρόσβαση διαχειριστή (δικαίωμα υπερχρήστη - root access) στο λειτουργικό σύστημα, καθώς ευαίσθητες κατατμήσεις (partitions) όπως /system είναι προσπελάσιμες μόνο για ανάγνωση (read-only). Ωστόσο τα δικαιώματα υπερχρήστη μπορούν να επιτευχθούν με την αξιοποίηση κενών ασφαλείας, το οποίο χρησιμοποιείται συχνά από τις κοινότητες ανοιχτού κώδικα για να ενισχύσει τις δυνατότητες των συσκευών τους, αλλά και από κακόβουλα μέρη για να εγκαταστήσουν ιούς και κακόβουλο λογισμικό. [6][7]

3.4.2 Βιβλιοθήκες (Libraries)

Οι βιβλιοθήκες τρέχουν στο παρασκήνιο του συστήματος και χρησιμοποιούν γλώσσες C/C++. Υπάρχουν 4 τύποι βιβλιοθηκών:

- Bionic libc, , «παράγωγο» της πρότυπης βιβλιοθήκης C (standard C library) , C βιβλιοθήκες συστήματος
- Βιβλιοθήκες λειτουργιών, υποστήριξη πολυμέσων, πρόγραμμα περιήγησης ιστού, SQLite κλπ
- Native Servers
- Βιβλιοθήκες του στρώματος αφαίρεσης υλικού (Hardware Abstraction Layer)

3.4.3 Εκτέλεση (Android Runtime)

Η έκδοση Android 4.4 εισήγαγε το Android Runtime (ART), ένα καινούριο περιβάλλον εκτέλεσης (σε πειραματικό στάδιο και όχι ενεργοποιημένο από προεπιλογή), το οποίο χρησιμοποιεί μεταγλωττιστή AOT (Ahead-Of-Time compiler), με τον οποίο γίνεται μεταγλώττιση μόνο μία φορά, όταν εγκαθίσταται η εφαρμογή. Αντίθετα, με τον JIT μεταγλωττιστή (Just-In-Time) (Dalvik Virtual Machine) όταν εγκαθίσταται η εφαρμογή ο κώδικας μετατρέπεται μερικώς σε μια γλώσσα που μπορεί να μετατραπεί ξανά, πιο εύκολα, σε γλώσσα μηχανής σε σχέση με την αρχική γλώσσα. Κάθε φορά που ανοίγει μια εφαρμογή, εκείνη τη στιγμή μετατρέπει πλήρως τον κώδικα σε γλώσσα μηχανής, χρησιμοποιώντας λιγότερο χώρο αποθήκευσης αλλά απαιτώντας περισσότερη επεξεργαστική ισχύ. Η χρήση λοιπόν της εικονικής μηχανής ART (αντί της Dalvik), η οποία είναι υποχρεωτική για το Android 5.0, απαιτεί λιγότερες ενέργειες από τον επεξεργαστή προκειμένου να τρέξει μία εφαρμογή**,** και έτσι, οι εφαρμογές ανοίγουν πιο γρήγορα (ειδικά σε συσκευές με πιο αδύναμους επεξεργαστές), αφού γίνεται καλύτερη διαχείριση των πόρων της συσκευής.

1) **Βιβλιοθήκες πυρήνα (Core Libraries)**

- System C Library - μια BSD υλοποίηση που προέρχεται απο την εφαρμογή της πρότυπης C βιβλιοθήκης συστήματος (libc), συντονισμένη για συσκευές με ενσωματωμένο Linux-based σύστημα.
- Media Libraries – αυτές οι βιβλιοθήκες υποστηρίζουν την αναπαραγωγή και την καταγραφή ήχων, εικόνων και βίντεο μορφών οπως MPEG4, H.264, MP3, AAC, JPG, και PNG.
- Surface Manager - χειρίζεται τη διαχείριση της οθόνης, που συγκροτείται από 2D και 3D γραφικά στρώματα.
- SGL - σημαίνει " Scalable Graphics Library" και παρέχει 2D γραφικά υποσυστήματα για το Android. Συνεργάζεται με τον Surface Manager και με τον Window Manager για την υλοποίηση του συνολικής σωλήνωσης των Android γραφικών.
- 3D Libraries - μια υλοποίηση της OpenGL ES 1.0 η οποία βασίζεται στην OpenGL 1.3. OpenGL ES 2.0 -βασίζεται σε OpenGL 2.0 - υποστηρίζεται από το Android 2.0 και το R3 NDK. Οι βιβλιοθήκες χρησιμοποιούν είτε επιτάχυνση 3D υλικού (όπου είναι διαθέσιμο) ή το ιδιαίτερα βελτιωμένο 3D rasterizer λογισμικού.
- LibWebCore - μια μηχανή περιήγησης ιστού που βασίζεται στο Webkit το οποίο δίνει δυνατότητα λειτουργίας και σε προγράμματα περιήγησης και ενσωματώσιμες web εφαρμογές. Επίσης, υποστηρίζει SSL για παροχή ασφάλειας Διαδικτύου.
- FreeType - μια bitmap και vector-based βιβλιοθήκη απεικόνισης γραμματοσειρών.
- SQLite – μια ελαφριά, σχεσιακή βάση δεδομένων, διαθέσιμη σε όλες τις εφαρμογές.

2) Εικονική Μηχανή Dalvik

Οι λειτουργίες των Java Core libraries βασίζονται στην εικονική μηχανή Dalvik και στον υποκείμενο πυρήνα των Linux. Πολλαπλά εικονικά μηχανήματα Dalvik μπορούν να τρέχουν ταυτόχρονα. Κάθε εφαρμογή Android τρέχει τη δική της διαδικασία, με το δικό της «περίπτωση» (instance) εικονικής μηχανής Dalvik. Ο μεταγλωττισμένος τύπος .class της java μετατρέπεται σε τύπο .dex όταν «χτίζεται» η εφαρμογή. Κάθε εφαρμογή (.apk) λαμβάνει το δικό της μοναδικό αναγνωριστικό (ID) χρήστη Linux και ομαδικό αναγνωριστικό. Η εικονική αυτή μηχανή είναι σχεδιασμένη έτσι ώστε ο διερμηνέας (interpreter) να προσφέρει βελτιστοποίηση της χρήσης της κεντρικής μονάδας επεξεργασίας (CPU) και να επιτυγχάνεται αποτελεσματική χρήση της μνήμης κατά τη διάρκεια εκτέλεσης. [8]

3.4.4 Πλαίσιο εφαρμογών (Application Framework)

Το πλαίσιο εφαρμογών απλοποιεί την επαναχρησιμοποίηση των στοιχείων, καθώς οι εφαρμογές μπορούν να δημοσιεύσουν τις δυνατότητές τους και οποιαδήποτε άλλη εφαρμογή μπορεί να χρησιμοποιήσει αυτές τις δυνατότητες. Περιλαμβάνει:

- Διαχειριστή Εφαρμογών (Activity Manager), που ελέγχει τη διάρκεια ζωής τους
- Διαχειριστή Ειδοποιήσεων (Notification Manager), που επιτρέπει στις εφαρμογές να εμφανίζουν προσαρμοσμένες ειδοποιήσεις στη γραμμή κατάστασης (status bar)
- Διαχειριστή Πηγών (Resource Manager), ο οποίος παρέχει πρόσβαση σε πηγές εκτός κώδικα όπως γραφικά και αρχεία διαρρύθμισης
- Παρόχους Περιεχομένου (Content Providers), που επιτρέπουν την πρόσβαση σε δεδομένα άλλων εφαρμογών ή μοιράζονται τα δικά τους δεδομένα
- Όψεις (Views), οι οποίες χρησιμοποιούνται για να χτιστεί μία εφαρμογή, συμπεριλαμβανομένων λιστών, πλεγμάτων, παραθύρων κειμένου, κουμπιών κλπ

3.4.5 Εφαρμογές (Applications)

Το Android περιλαμβάνει ένα σύνολο από βασικές εφαρμογές γραμμένες σε Java, όπως πελάτης ηλεκτρονικού ταχυδρομείου (email client), πρόγραμμα σύντομων μηνυμάτων (SMS), ημερολόγιο, χάρτες, φυλλομετρητή, επαφές κ.ά.

3.5 Συστατικά στοιχεία της εφαρμογής (App Components)

Τα συστατική στοιχεία μιας εφαρμογής είναι τα δομικά εκείνα μέρη που είναι απαραίτητα για την κατασκευή της. Κάθε στοιχείο είναι και ένας διαφορετικός τρόπος με τον οποίο το λειτουργικό μπορεί να εισέλθει στην εφαρμογή. Τα στοιχεία αυτά μπορούν να διακριθούν σε τέσσερα βασικά είδη. Κάθε είδος επιτελεί τον δικό του συγκεκριμένο ρόλο και έχει ένα ξεχωριστό κύκλο ζωής σύμφωνα με τον οποίο δημιουργείται και καταστρέφεται.
Τα τέσσερα αυτά είδη είναι:

1. **Activities**
2. **Services**
3. **Content Providers**
4. **Broadcast Receivers**

3.5.1 Activities

Οι activities αποτελούν το βασικότερο στοιχείο μιας εφαρμογής. Κάθε activity ουσιαστικά αντιπροσωπεύει μια οθόνη μαζί με την διεπιφάνεια χρήστη. Μια εφαρμογή συνήθως αποτελείται από πολλές activities οι οποίες είναι χαλαρά δεμένες μεταξύ τους. Συνηθέστερα, μια activity αποτελεί την main (βασική) activity η οποία είναι και η οθόνη που βλέπει ο χρήστης όταν ανοίγει την εφαρμογή για πρώτη φορά. Κάθε activity μπορεί να καλέσει μια νέα activity για να εκτελέσει διαφορετικές λειτουργίες. Κάθε φορά που μια νέα activity ξεκινάει, η προηγούμενη activity σταματάει. Το σύστημα όμως την διατηρεί και την αποθηκεύει σε μια στοίβα (LIFO), το λεγόμενο "back stack". Όταν λοιπόν μια νέα activity ξεκινάει, τότε εμφανίζεται στην οθόνη και αποθηκεύεται τελευταία στο back stack. Έτσι, όταν ο χρήστης τελειώσει με τη χρήση της και πατήσει το πλήκτρο Back, τότε αυτή γίνεται pop από την στοίβα, καταστρέφεται και η οθόνη επανέρχεται στην προηγούμενη activity. Αν πατηθεί το πλήκτρο Back ενώ ο χρήστης βρίσκεται στην main activity τότε εξέρχεται από την εφαρμογή.

Για να κατασκευαστεί μια activity, πρέπει να δημιουργηθεί μια υποκλάση της βασικής κλάσης Activity. Σε αυτή την υποκλάση πρέπει να εφαρμοστούν συγκεκριμένες μέθοδοι τις οποίες καλεί το σύστημα όταν η activity μεταβαίνει σε διαφορετικά στάδια του κύκλου ζωής της. Τέτοια στάδια είναι όταν η activity δημιουργείται (create), σταματάει (stop), επανέρχεται (resume) ή καταστρέφεται (destroy). [9]

Οι δύο πιο βασικές τέτοιες μέθοδοι είναι:

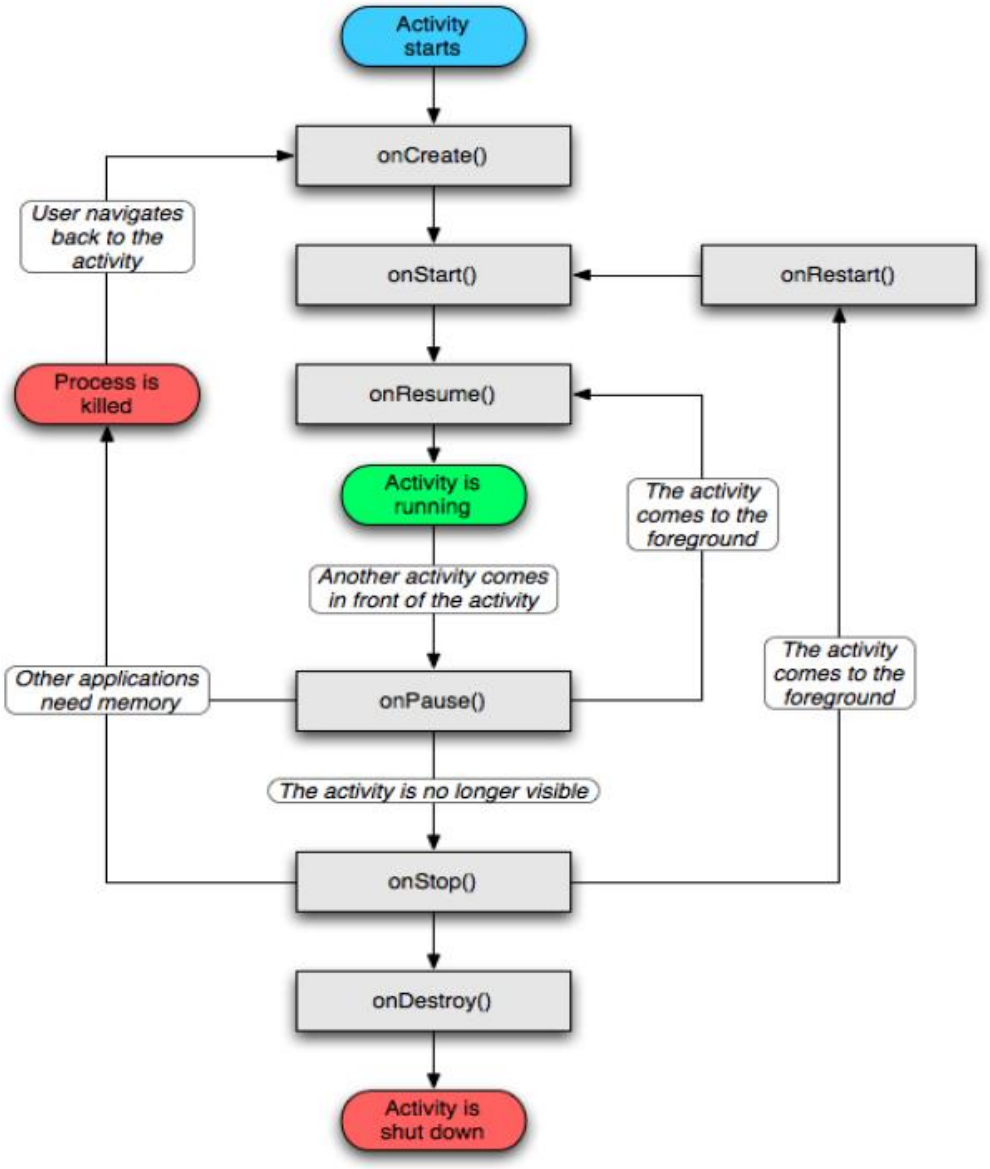
1. **onCreate( )**

Πρέπει αναγκαστικά να εφαρμοστεί αυτή η μέθοδος. Το σύστημα την καλεί όταν δημιουργεί την activity. Εδώ πρέπει να αρχικοποιηθούν τα απαραίτητα στοιχεία της activity. Το πιο σημαντικό είναι να γίνει κλήση σε αυτό το σημείο στην μέθοδο setContentView( ) με την οποία ορίζεται η διάταξη (layout) της διεπιφάνειας χρήστη (user interface) αυτής της activity.

2. **onPause( )**

Το σύστημα καλεί αυτή τη μέθοδο όταν έχει ενδείξεις ότι ο χρήστης φεύγει από την activity (χωρίς να σημαίνει η activity καταστρέφεται). Εδώ συνήθως ο προγραμματιστής πρέπει να κάνει όποιες αλλαγές θα χρειαστεί να παραμείνουν και μετά το πέρας αυτής της περιόδου γιατί ο χρήστης μπορεί να μην επιστρέψει.

Υπάρχουν και άλλες βασικές μέθοδοι πέρα από αυτές, οι οποίες χρησιμοποιούνται από το σύστημα και που ο προγραμματιστής θα πρέπει να χρησιμοποιεί ώστε να προσφέρει στον χρήστη μια ομαλή εμπειρία χρήσης. Ο προγραμματιστής θα πρέπει να έχει χρησιμοποιήσει όλες τις κατάλληλες μεθόδους ώστε καθώς ο χρήστης θα πλοηγείται ανάμεσα στις διάφορες activities, η εφαρμογή να μπορεί να χειριστεί απρόσμενες διακοπές οι οποίες θα αναγκάζουν τα activities να σταματούν ή ακόμα και να καταστρέφονται. Ο βασικός κύκλος ζωής μιας activity φαίνεται στο Σχήμα 3.2.



Σχημα 3.2 Κύκλος ζωής Activity

3.5.1.1 Fragments

Τα Fragments παρά το ότι είναι ένα εξειδικευμένο στοιχείο στην ανάπτυξη εφαρμογών για Android, είναι αναγκαίο να αναφερθούν στην παρούσα διπλωματική διότι έχουν χρησιμοποιηθεί κατά κόρον στην παρούσα εφαρμογή. Επίσης, είναι ένα αρκετά νέο στοιχείο στον κόσμο του Android καθ' ότι έγινε διαθέσιμο από το API 11 και μετά, δηλαδή από την έκδοση 3.0. Από τότε όμως χρησιμοποιείται ευρύτατα από τους προγραμματιστές μιας και είναι πάρα πολύ χρήσιμο και ευέλικτο.
Τα Fragments αναπαριστούν τη συμπεριφορά ή ένα μέρος της διεπιφάνειας χρήστη μέσα σε μια Activity. Μπορούν να συνδυαστούν πολλά fragments σε μια μόνο activity ώστε να κατασκευαστεί ένα UI με πολλά παράθυρα και επίσης μπορεί να επαναχρησιμοποιηθεί το ίδιο fragment σε πολλές activities. Ουσιαστικά, ένα fragment είναι κατασκευαστικό κομμάτι μιας activity, με το δικό του κύκλο ζωής, την δική του είσοδο δεδομένων και το οποίο μπορεί να προστεθεί ή να αφαιρεθεί στη διάρκεια που μια εφαρμογή εκτελείται.

Ένα fragment πρέπει πάντα να είναι ενσωματωμένο μέσα σε μια activity και ο κύκλος ζωής του επηρεάζεται άμεσα από τον κύκλο ζωής αυτής της activity. Για παράδειγμα, όταν η activity σταματάει, το ίδιο συμβαίνει και σε όλα τα fragments μέσα σε αυτή. Όταν η activity καταστρέφεται, καταστρέφονται και όλα τα fragments σε αυτήν. Όμως, το σημαντικό κομμάτι είναι ότι, καθώς η activity εκτελείται, ο προγραμματιστής μπορεί να χειριστεί κάθε fragment ξεχωριστά, να προσθέσει ή να αφαιρέσει διαφορετικά fragments, ακόμα και να αντικαταστήσει fragments με άλλα fragments – αυτό ονομάζεται fragment transaction. Όπως και με τις activities, έτσι και τα fragments, μπορούν να τα προστεθούν στο back stack. Το back stack αυτό χρησιμοποιείται από την activity και δίνει τη δυνατότητα στο χρήστη να αντιστρέψει μια fragment transaction, να πλοηγηθεί δηλαδή προς τα πίσω, με τη χρήση του πλήκτρου Back. [9]

3.5.2 Services

Ένα service είναι ένα στοιχείο το οποίο τρέχει στο παρασκήνιο (background) και εκτελεί χρονοβόρες διαδικασίες. Ένα service δεν παρέχει διεπιφάνεια χρήστη (user interface). Για παράδειγμα, ένα service μπορεί να παίζει μουσική στο παρασκήνιο, ενώ ο χρήστης βρίσκεται σε μια άλλη εφαρμογή. Ένα service εφαρμόζεται ως υποκλάση της βασικής κλάσης Service.

3.5.3 Content Providers

Ένας content provider διαχειρίζεται ένα μοιραζόμενο σύνολο από δεδομένα εφαρμογών. Μέσω αυτών μπορούν να αποθηκευθούν δεδομένα στο σύστημα

αρχείων, σε μια βάση δεδομένων SQLite, στο δίκτυο, ή σε οποιαδήποτε άλλο αποθηκευτικό χώρο έχει πρόσβαση η εφαρμογή.

3.5.4 Broadcast Receivers

Ένας broadcast receiver είναι ένα στοιχείο το οποίο αποκρίνεται σε γεγονότα τα οποία «ανακοινώνονται» από το ίδιο το λειτουργικό σύστημα. Τέτοια γεγονότα μπορεί να είναι το ότι έσβησε η οθόνη, το ότι η στάθμη της μπαταρίας είναι χαμηλή ή ότι ο χρήστης τράβηξε μια φωτογραφία. Για παράδειγμα, αρκετές εφαρμογές λήψης φωτογραφιών που χρησιμοποιούν flash, όταν η στάθμη της μπαταρίας είναι χαμηλή το απενεργοποιούν για να εξοικονομήσουν ενέργεια. Ενημερώνονται δηλαδή μέσω ενός broadcast receiver για το γεγονός και αντιδρούν ανάλογα.

3.5.5 Processes and Threads (Διαδικασίες και Νήματα)

Όταν ένα συστατικό στοιχείο της εφαρμογής ξεκινάει να εκτελείται και η εφαρμογή δεν έχει κάποιο άλλο στοιχείο να τρέχει, τότε το Android ξεκινάει μια νέα Linux process (διαδικασία) για την εφαρμογή, με ένα μόνο thread (νήμα) εκτέλεσης. Όλα τα στοιχεία της ίδιας εφαρμογής τρέχουν στην ίδια διαδικασία και στο ίδιο νήμα. Αν δηλαδή ένα στοιχείο μιας εφαρμογής ξεκινήσει να εκτελείται και υπάρχει ήδη μια διαδικασία για αυτή την εφαρμογή, τότε το νέο στοιχείο ξεκινάει να εκτελείται μέσα σε αυτή την διαδικασία και χρησιμοποιεί το ίδιο thread εκτέλεσης. Παρ' όλα αυτά, ο προγραμματιστής μπορεί να επέμβει έτσι ώστε διαφορετικά στοιχεία να τρέχουν σε διαφορετικές διαδικασίες και να δημιουργήσει νέα thread για κάθε διαδικασία.

3.5.6 Intents

Intents :
● Χρησιμοποιούνται όταν θέλουμε να μεταβούμε από το ένα Activity σε άλλο Activity.
● περιγραφεί τι θέλει η εφαρμογή.
● Τα Activities, Services και Broadcast receivers
● επικοινωνούν μεταξύ τους με Intents.
● Μηχανισμός μεταφοράς δεδομένων
Services:
- Δεν αλληλεπιδρούν με το χρήστη
- τρέχουν με το main thread της προόδου της εφαρμογής .

3.5.7 Notifications (Ειδοποιήσεις)

Notifications:

- Ειδοποιήσεις προς τον χρηστή για events.
- Ελέγχονται από τον NotificationManager

Σχήμα 3.3 Ειδοποήσεις

3.5.8 AndroidManifest.xml

Το AndroidManifest.xml, που παρουσιάζει τις απαραίτητες πληροφορίες για την εφαρμογή στο σύστημα Android, οι οποίες χρειάζονται ώστε να μπορέσει το σύστημα να «τρέξει» οποιοδήποτε κώδικα της εφαρμογής. Κάθε εφαρμογή πρέπει να έχει το ακριβές όνομα αυτού του αρχείου. [9]

Μερικές λειτουργίες του αρχείου AndroidManifest:

- Καθορίζει το όνομα του πακέτου της εφαρμογής και είναι μοναδικό
- Περιγράφει τα συστατικά (components) που συνθέτουν την εφαρμογή.

- Καθορίζει ποιες άδειες (permissions) πρέπει να έχει η εφαρμογή για να μπορέσει να αλληλεπιδράσει με άλλες εφαρμογές καθώς και αντίστροφα (πρόσβαση σε κάρτα SD, χρήση της φωτογραφικής, κλπ).
- Καθορίζει την ελάχιστη έκδοση της διεπαφής του προγράμματος (API) που χρειάζεται η εφαρμογή για να λειτουργήσει.
- Παρουσιάζει την λίστα με τις βιβλιοθήκες που χρησιμοποιεί και συνδέεται η εφαρμογή.

3.6 Βάση Δεδομένων

Τα διαγράμματα οντοτήτων-σχέσεων, έχουν σαν βασικό στοιχείο τους την οντότητα. Η οντότητα, είναι μια αναπαράσταση κάποιας αυτόνομης ύπαρξης με υλική (στον πραγματικό κόσμο) ή θεωρητική υπόσταση (συμβατική ύπαρξη). Για παράδειγμα, οντότητα μπορεί να είναι ένας φοιτητής (ένας άνθρωπος με ονοματεπώνυμο, χαρακτηριστικά, κ.α) αλλά και ένα μάθημα σε μια σχολή (κάτι άυλο αλλά με συμβατική υπόσταση).

Τα δεδομένα τα οποία θα αποθηκευτούν σε κάποια οντότητα του μοντέλου οντοτήτων-σχέσεων, αντιστοιχούν στις εγγραφές στο φυσικό επίπεδο μίας Βάσης Δεδομένων. Έτσι η οντότητα, θα πρέπει να έχει κάποιο όνομα, και κάποια στοιχεία που να καθορίζουν τα χαρακτηριστικά της συγκεκριμένης οντότητας[17]

Όπως και στις περισσότερες πλατφόρμες το Android δίνει τη δυνατότητα να αποθηκεύσουμε δεδομένα ώστε να διατηρούνται μετά το τέλος της εφαρμογής. Υπάρχουν πολλοί τρόποι μπορεί να γίνει αυτό, αρχεία κειμένου (text) – αποθηκευμένα στην εξωτερική κάρτα SD του κινητού, αποθηκευμένα αρχεία (assets) κατά τη μεταγλώττιση της εφαρμογής τα οποία χρησιμοποιούνται μόνο για ανάγνωση. Κάποιες φορές όμως πρέπει να πραγματοποιήσουμε σύνθετους χειρισμούς σε δεδομένα και απαιτείται πιο αποδοτικός τρόπος από την διαχείριση ενός αρχείο κειμένου. Σε αυτή την περίπτωση χρησιμοποιείται μια βάση δεδομένων. Το Android χρησιμοποιεί την βάση δεδομένων SQLite.

Η SQLite DataBase είναι μία βάση ανοιχτού κώδικα που υποστηρίζει όλα τα χαρακτηριστικά σχεσιακών βάσεων δεδομένων, όπως η SQL. Τέτοιου τύπου βάση είναι ενσωματωμένη σε όλες τις συσκευές που χρησιμοποιούν Android και εάν η εφαρμογή δημιουργεί μία βάση δεδομένων, τότε αυτή αποθηκεύεται αυτόματα στο directory DATA/data/APP_NAME/databases/FILENAME. Για τη δημιουργία της βάσης αρκεί να φτιάξουμε μία υποκλάση της κλάσης SQLiteOpenHelper. [10][11][12]

3.6 Συναρμογές Δικτύου (Network Sockets)

Ένα network socket είναι το ένα άκρο μιας επικοινωνίας μεταξύ δύο διεργασιών πάνω από ένα δίκτυο υπολογιστών.
Το Socket API είναι ένα Application Programming Interface (API), συνήθως, παρέχεται από το λειτουργικό σύστημα, που επιτρέπει σε προγράμματα εφαρμογής να ελέγχουν και να χρησιμοποιούν Network Sockets. Σκοπός του Socket API είναι η γενική επικοινωνία μεταξύ διεργασιών (που τρέχουν στον ίδιο ή σε διαφορετικούς υπολογιστές).
Μία διεύθυνση socket είναι ένας συνδυασμός μια διεύθυνσης IP και ενός αριθμού θύρας (port number). Βασιζόμενο σε αυτή τη διεύθυνση, το internet socket στέλνουν δεδομένα σε μορφή πακέτων στην αντίστοιχη εργασία ή νήμα.
Ένα Internet socket εκτός από την τοπική του διεύθυνση χαρακτηρίζεται και από το πρωτόκολλο μεταφοράς. Τα κυριότερα είναι το TCP και το UDP. Δύο sockets με ίδια τοπική διεύθυνση αλλά διαφορετικό πρωτόκολλο είναι διακριτά μεταξύ τους. Ένα socket που έχει συνδεθεί σε ένα άλλο χαρακτηρίζεται από το εξής:

Απομακρυσμένη διεύθυνση socket (Remote Sockets Address)

Ένας εξυπηρετητής TCP μπορεί να εξυπηρετήσει πολλούς πελάτες ταυτόχρονα. Ο εξυπηρετητής δημιουργεί ένα socket για κάθε πελάτη με την ίδια τοπική διεύθυνση από την μεριά του server αλλά με διαφορετική απομακρυσμένη διεύθυνση socket.[13]

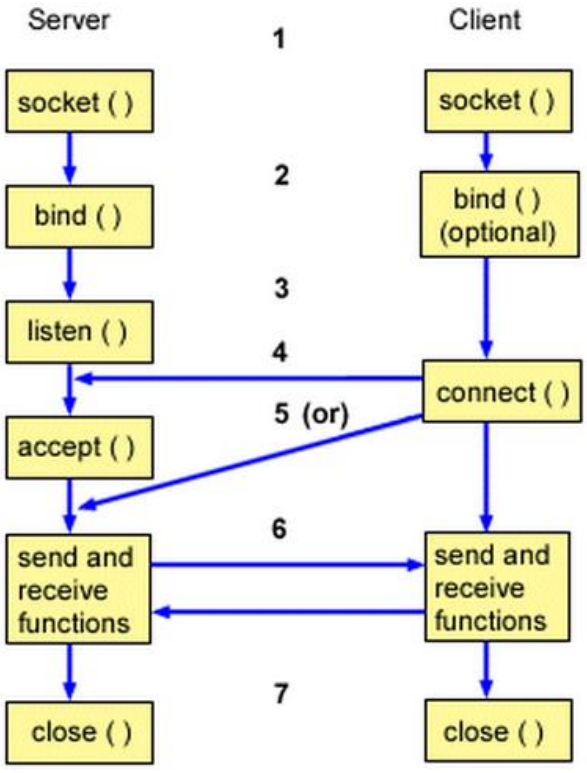
3.6.1 Προγραμματισμός με sockets

Τα sockets παρέχουν τον μηχανισμό επικοινωνίας μεταξύ δύο υπολογιστών χρησιμοποιώντας το πρωτόκολλο TPC. Ένα πρόγραμμα πελάτης δημιουργεί ένα socket στην δική του μεριά του δίαυλου επικοινωνίας και προσπαθεί να συνδέσει

αυτό το socket σε έναν server. Όταν δημιουργηθεί η σύνδεση, ο server δημιουργεί ένα socket και στην συνέχεια μπορεί να επικοινωνήσει και να διαβάσει από αυτό. Το TCP είναι ένα πρωτόκολλο επικοινωνίας, ώστε τα δεδομένα να μπορούν να σταλθούν και να ληφθούν ταυτόχρονα. [14]

Ένα socket έχει μια τυπική ροή γεγονότων. Σε μια σύνδεση πελάτης-προς-εξυπηρετητή το socket στην μεριά του server περιμένει αιτήματα από τον πελάτη. Για να γίνει αυτό ο server καθορίζει την διεύθυνση (binds) που θα χρησιμοποιούν οι clients για να συνδεθούν σε αυτόν. Όταν η διεύθυνση καθιερωθεί, ο server περιμένει τους clients να ζητήσουν μια υπηρεσία. Η ανταλλαγή δεδομένων client-προς-server γίνεται όταν ο client συνδεθεί στον server μέσω ενός socket. Ο server επεξεργάζεται το αίτημα του client και στέλνει την απάντηση πίσω.

Παρακάτω εμφανίζεται το διάγραμμα σύνδεσης και ανταλλαγής δεδομένων μεταξύ ενός πελάτη και ενός εξυπηρετητή.



Σχήμα 3.4 Διάγραμμα σύνδεσης και ανταλλαγής δεδομένων μεταξύ server – client

1) Η εντολή socket()δημιουργεί το Socket το χειρίζεται μέσω ενός θετικού ακεραίου - socket descriptor το οποίο το ένα άκρο της επικοινωνίας. Αρχικά το socket δεν έχει IP διεύθυνση ή port. Συνδέει το αποτελέσμα που έχει επιστραφεί από την κλήση socket(), με μία τοπική διεύθυνση και θύρα (IP address, port number), γνωστοποιώντας στο σύστημα ότι τα μηνύματα που έρχονται στα συγκεκριμένα: (interface - port) απευθύνονται στη συγκεκριμένη διεργασία

2) Η συνάρτηση bind() απλά δεσμεύει το socket descriptor (που έχει επιστρέψει η κλήση socket() με μια τοπική διεύθυνση και μια θύρα "λέγοντας" στο σύστημα ότι τα μηνύματα που έρχονται στα συγκεκριμένα port-interface απευθύνονται στην συγκεκριμένη διαδικασία.

3) Η μέθοδος listen() δηλώνει ότι ο client είναι έτοιμος να δεχτεί συνδέσεις από τον server. Η κλήση listen() είναι προαιρετική. Η συνάρτηση αυτή θα δέχεται εισερχόμενες συνδέσεις στην socket του συστήματος που ήδη έχει δεσμευτεί από την bind μόλις γίνει μια επιτυχής σύνδεση ένας νέος περιγραφέας υποδοχής επιστρέφεται και μπορεί να χρησιμοποιηθεί για την επικοινωνία του προγράμματος.

4) Ο client πρέπει να συνδεθεί στο socket με την χρήση της μεθόδου connect().

5) Η κλήση accept() κάνει ουσιαστικά αποδοχή μιας αίτησης σύνδεσης . Από την στιγμή αυτή, το κανάλι επικοινωνίας με τον client έχει εγκατασταθεί . Μέχρι να ξανακληθεί η accept(), ο server δεν μπορεί να δεχτεί άλλες κλήσεις για σύνδεση . Οι κλήσεις που γίνονται όσο ο server βρίσκεται εκτός της accept() τοποθετούνται σε μια ουρά ή οποία μπορεί να αλλάξει το μέγεθος της με την κλήση listen())

6) Όταν πραγματοποιηθεί η σύνδεση μεταξύ των client και server sockets μπορούν να χρησιομοποιηθούν οι μέθοδοι για την επικοινωνία μεταξύ τους όπως οι read() και write().

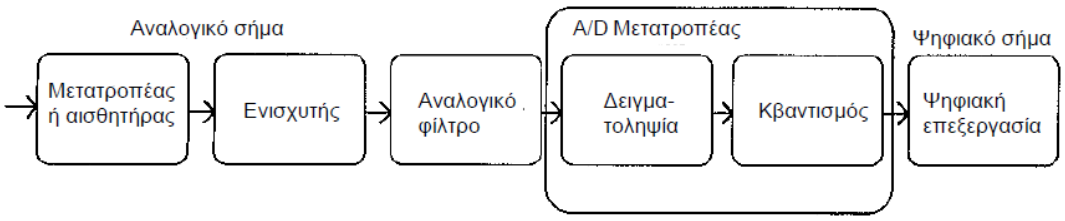
7) Όταν oserver θέλει να τερματήσει την επικοινωνία καλεί την μέθοδο close() και απελευθερώνει τους πόρους του συστήματος. [15]

4 Σχεδιασμός

4.1 Εισαγωγή

Στο παρόν κεφαλαίο θα περιγραφεί αναλυτικά η εφαρμογή που αναπτύχθηκε σε περιβάλλον Android. Η εφαρμογή που περιγράφεται απεικονίζει το ηλεκτροκαρδιογράφημα σε πραγματικό χρόνο στην οθόνη ενός smartphone που έχει το android ως λειτουργικό σύστημα. Η σύνδεση με την συσκευή καταγραφής ή τον server γίνεται χρησιμοποιώντας sockets μέσω πρωτοκόλλου TCP. Το περιβάλλον που επιλέχτηκε για την ανάπτυξη την εφαρμογής ήταν το Android. Ένας από τους κύριους λόγους που επιλέχτηκε είναι η δυνατότητα να τρέξει μία εφαρμογή σαν «service» δηλαδή τρέχει συνέχεια στο παρασκήνιο κάτι το οποίο το iOS δεν επιτρέπει.

Στην εικόνα βλέπουμε ένα διάγραμμα λήψης ενός σήματος. Η εφαρμογή που υλοποιήθηκε αναλαμβάνει το κομμάτι μετά την καταγραφή του σήματος δηλαδή μετά την μετατροπή του από αναλογικό σε ψηφιακό.



Σχήμα 4.1 Γενικό Διάγραμμα λήψης ενός ψηφιακού σήματος

Σχήμα 4.2 Αρχιτεκτονική συστήματος

4.2 Καταγραφή Σήματος

Η καταγραφή είναι μία ενέργεια η οποία προσεγγίζεται θεωρητικά δεδομένου ότι δεν υπάρχει σύστημα διαθέσιμο έτσι ώστε να γίνουν τα ανάλογα πειράματα.

Γι' αυτό χρησιμοποιήθηκε η βάση δεδομένων PhysioBank (http://www.physionet.org), για να προσομοιώσουμε την απεικόνιση και την αποθήκευση του ηλεκτροκαρδιογραφικού σήματος.

Η Physiobank, η οποία είναι μέρος της Physionet, είναι μια μεγάλη και συνεχώς αναπτυσσόμενη τράπεζα ψηφιακών καταγραφών φυσιολογικών σημάτων και σχετικών δεδομένων, διαθέσιμων προς χρήση από την βιοϊατρική ερευνητική κοινότητα (http://www.physionet.org/physiobank). Η Physiobank περιέχει βάσεις δεδομένων καρδιακών, νευρικών και άλλων βιοσημάτων από υγιή υποκείμενα, καθώς και από υποκείμενα-ασθενείς με ποικιλία παθολογικών καταστάσεων συμπεριλαμβανομένων του αιφνίδιου καρδιακού θανάτου, καρδιακής ανεπάρκειας, επιληψίας, υπνικής άπνοιας, στηθάγχης κλπ. Η Physionet παρέχει ελεύθερη πρόσβαση μέσω του διαδικτύου σε μεγάλες συλλογές καταγεγραμμένων βιοσημάτων, καθώς και σχετικό λογισμικό ανοικτού κώδικα. Η ιστοσελίδα της Physionet επιχορηγείται από τους παρακάτω φορείς: National Institutes of Health (http://www.nih.gov), NIBIB (http://www.nibib.nih.gov) και NIGMS (http://www.nigms.nih.gov).

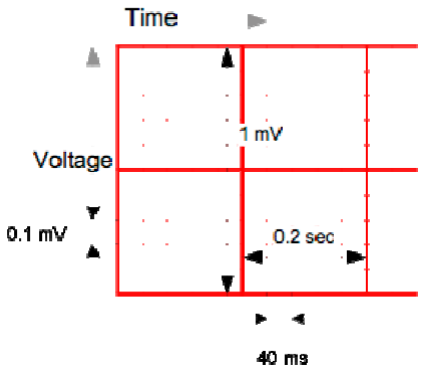
4.3 Λήψη Σήματος

Η εφαρμογή που αναπτύχθηκε αναλύει το κομμάτι μετά την καταγραφή του σήματος. Θεωρήθηκε ότι το σύστημα ηλεκτροκαρδιογράφου έχει δυνατότητα αποστολής δεδομένων ασύρματα. Για την ασύρματη μεταφορά των δεδομένων επιλέχθηκε η χρήση των sockets. Με την βοήθεια των sockets το κινητό τηλέφωνο (client) μπορεί να συνδεθεί στον εξυπηρετητή (server) χρησιμοποιώντας μία διεύθυνση IP και μια θύρα. Ο server μπορεί να είναι ο κεντρικός υπολογιστής του νοσοκομείο ο οποίος λαμβάνει όλα τα ηλεκτροκαρδιογραφήματα από τους ασθενείς είτε ένα σύστημα ηλεκτροκαρδιογράφου συνδεδεμένο σε κάποιον ασθενή το οποίο έχει δυνατότητα να στείλει μεμονωμένα το καταγραφόμενο ECG.

4.4 Απεικόνιση Σήματος

Όπως αναφέρθηκε η εφαρμογή αναπτύχθηκε σε android άρα η απεικόνιση του σήματος θα γίνεται σε μία οθόνη ενός smart phone ή tablet που έχουν το android ως λειτουργικό σύστημα. Υπάρχουν πολλά μεγέθη οθονών γι'αυτό τον λόγο υπήρχε η απαίτηση, η εφαρμογή να λειτουργεί για κάθε διαφορετικό μέγεθος. Αυτό επιτυγχάνεται χρησιμοποιώντας σαν παραμέτρους το πλάτος και το ύψος των οθονών.

Βασική ιδιότητα όλων των ηλεκτροκαρδιογράφων είναι το ότι η επιλογή ταχύτητας καταγραφής του χαρτιού αλλά και τα τετράγωνα που είναι ζωγραφισμένα σε αυτό είναι τυποποιημένα. Το ηλεκτροκαρδιογραφικό χαρτί κινείται με ταχύτητα 25mm/sec και αποτελείται από τετράγωνα μιλιμετρέ. Συγκεκριμένα, κάθε μικρό τετράγωνο το οποίο έχει μήκος 1mm αντιστοιχεί σε 0,04 sec, ενώ ένα μεγάλο τετράγωνο μήκους 5mm αντιστοιχεί σε 0,2 sec αντίστοιχα.



Σχήμα 4.3 Ηλεκτροκαρδιογραφικό χαρτί

Για λόγους ευκρίνειας επιλέξαμε να εμφανίζεται στην οθόνη δύο δευτερόλεπτα. Υποθέσαμε ότι η το σύστημα του ηλεκτροκαρδιογράφου στέλνει 256 δείγματα για 1 δευτερόλεπτο. Κάθε δείγμα είναι ένας float αριθμός 32bit ή 4 byte. Άρα 8192 bits/sec είναι ο ρυθμός μεταφοράς δεδομένων.

Μια μέση ανάλυση κινητών είναι 1280χ720 pixels. Θέλουμε 256X2=512 τιμές στην οθόνη άρα αυτές οι τιμές θα μετατραπούν σε 720 pixels. Θέλουμε κάθε δευτερόλεπτο να λαμβάνονται οι επόμενες 256 τιμές από τον ηλεκτροκαρδιογράφο άρα το ηλεκτροκαρδιογράφημα θα μετακινείται 360pixels/sec αριστερά. Έτσι δημιουργήθηκε ένα νήμα (thread) που πραγματοποιεί την ανανέωση της οθόνης κάθε 1 sec. Αναφέρεται ότι ο ρυθμός εμφάνισης του ECG προσαρμόζεται κατάλληλα σε κάθε οθόνη κινητού για να μετακινείται 1 sec αριστερά.

Όπως αναφέρθηκε η καταγραφή του σήματος προσεγγίζεται θεωρητικά και τα δεδομένα λήφθηκαν από την βάση δεδομένων Physiobank. Το αρχείο που

χρησιμοποιήθηκε είναι αρχείο κειμένου που έχει δύο στήλες. Η πρώτη στήλη περιέχει τιμές του χρόνου και η δεύτερη το πλάτος του ECG. Δημιουργήθηκε μία συνάρτηση η οποία διαβάζει γραμμή – γραμμή το αρχείο και αποθηκεύει σε λίστες float αριθμών τις τιμές του ECG. Στην συνέχεια για να προσομοιωθεί η καταγραφή του σε ECG σε πραγματικό χρόνο το αρχείο text διαβάζει 256 τιμές κάθε δευτερόλεπτο γεμίζοντας έτσι τις λίστες.

4.5 Αποθήκευση

Η εφαρμογή επιτρέπει στον χρήστη να αποθηκεύσει το ηλεκτροκαρδιογράφημα που λαμβάνεται σε ψηφιακή μορφή σε αρχείο κειμένου. Το αρχείο αποθηκεύεται στον προκαθορισμένο φάκελο «ecg data» στην κάρτα sd του κινητού.

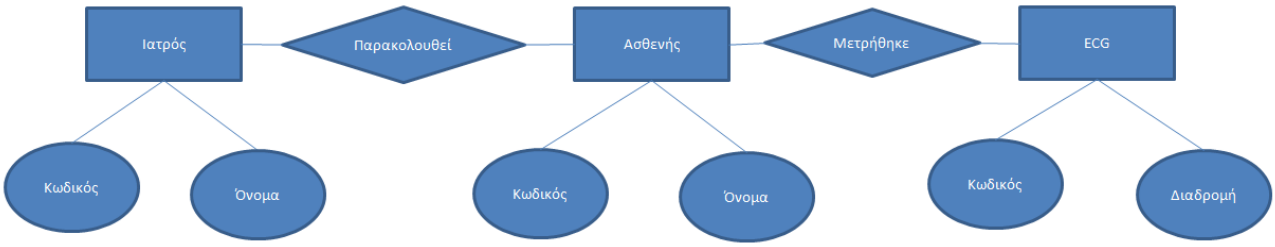
Το ηλεκτροκαρδιογράφημα που αποθηκεύτηκε θα πρέπει να συνδεθεί στην συνέχεια με τον αντίστοιχο ασθενή. Για να επιτευχθεί αυτό χρησιμοποιήθηκε μια βάση δεδομένων SQLite η οποία βρίσκεται τοπικά στο Smartphone. Ο χρήστης επιλέγοντας την σύνδεση με κάποιον ασθενή, το αρχείο κειμένου παίρνει συγκεκριμένη ονομασία. Ανάλογα τον κωδικό του ασθενή. Για να γίνει η σύνδεση που αναφέρθηκε δημιουργήθηκε ένα πεδίο που κρατάει το μέρος που είναι αποθηκευμένο.

4.6 Ανάκτηση σήματος

Ο χρήστης έχει τη δυνατότητα να ανακτήσει τα αποθηκευμένα σήματα που έχει αποθηκεύσει στην τοπική βάση δεδομένων. Το ECG συνδέεται με τον ασθενή από τον οποίο καταγράφθηκε και με αυτό τον τρόπο διατηρούμε ένα πλήρες ιστορικό για τον κάθε ασθενή. Έτσι ο γιατρός μπορεί να βγάλει γρήγορα συμπεράσματα καθώς μπορεί να ανατρέξει σε παλιότερα βιοσήματα πολύ εύκολα.

4.7 Βάση Δεδομένων

Παρακάτω εμφανίζεται μια πολύ απλή μορφή της βάσης δεδομένων της εφαρμογής. Για πρακτικούς λόγους αυτή η βάση δημιουργήθηκε τοπικά στο κινητό τηλέφωνο αλλά η φύση της εφαρμογής καθιστά απαραίτητο η βάση να υπάρχει σε έναν server στον οποίο θα συνδέεται ο χρήστης (ιατρός) και θα αντλεί τα δεδομένα. Οι οντότητες δημιουργήθηκαν με τα βασικά πεδία που απαιτούνται για να είναι το σύστημα λειτουργικό.

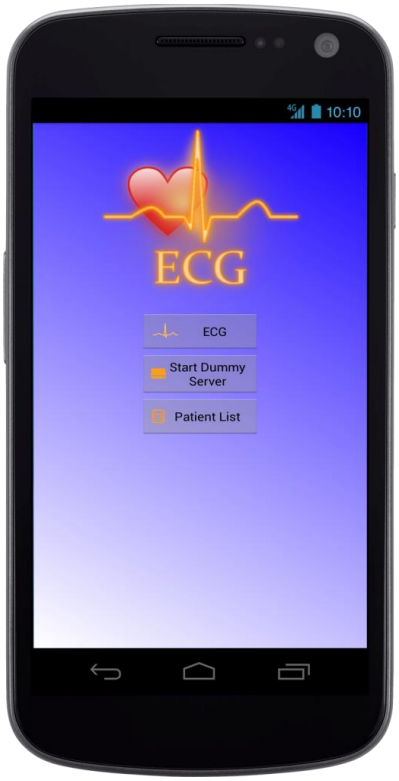


Σχήμα 4.4 Συσχέτιση οντοτήτων Γιατρού – Ασθενή - Ηλεκτροκαρδιογραφήματος

5 Εφαρμογή ECG και Λειτουργίες

5.1 Λειτουργίες

5.1.1 Κεντρικό Μενού



Οθόνη Κεντρικού Μενού

Όταν εκτελεστεί η εφαρμογή η πρώτη κλάση που έχει ρυθμιστεί να καλείται στο AndroidManifest.xml είναι η Menu.java. Περιέχει όλες τις απαραίτητες πληροφορίες για να αρχικοποιήσει το κεντρικό μενού της εφαρμογής. Παρακάτω βλέπουμε τον κώδικα της Menu.java και παρατηρούμε τις 3 μεθόδους setOnClickListener οι οποίες καλούν 3 κλάσεις και αντιστοιχούν σε 3 λειτουργίες της εφαρμογής όπως θα αναλυθούν παρακάτω.

```
        Startapp.setOnClickListener(new View.OnClickListener() {
            @Override

            public void onClick(View v) {
                Context context = getApplicationContext();
                int duration = Toast.LENGTH_SHORT;
                Toast toast = Toast.makeText(context, "Get Data",
duration);
                toast.show();
                Intent startapp = new
Intent(Menu.this,MainActivity.class);
                Menu.this.startActivity(startapp);
            }
        });

        patientlist.setOnClickListener(new View.OnClickListener() {
            @Override

            public void onClick(View v) {
                Context context = getApplicationContext();
                int duration = Toast.LENGTH_SHORT;
                Toast toast = Toast.makeText(context, "Database
Opened", duration);
                toast.show();
                Intent startapp = new
Intent(Menu.this,AndroidDatabase.class);
                Menu.this.startActivity(startapp);
            }
        });

        dserver.setOnClickListener(new View.OnClickListener() {
            @Override

            public void onClick(View v) {
                Intent serverActivity = new
Intent(Menu.this,ServerActivity.class);
                Menu.this.startActivity(serverActivity);

            }
        });
    }

    }
}
```

Ο χρήστης έχει 3 δυνατότητες οι οποίες καθορίζονται από τα 3 κουμπιά.

Το πλήκτρο ECG ξεκινάει την MainActivity διεργασία. Είναι η σημαντικότερη διεργασία της εφαρμογής διότι είναι καλεί τις κλάσεις που είναι υπεύθυνες για την απεικόνιση και αποθήκευση του ηλεκτροκαρδιογραφήματος, την Υπηρεσία (Service) που λαμβάνει τα δεδομένα του server, καθώς και τη διαχείριση όλων των νημάτων της εφαρμογής που ανανεώνουν την οθόνη όπως θα παρουσιαστεί παρακάτω.

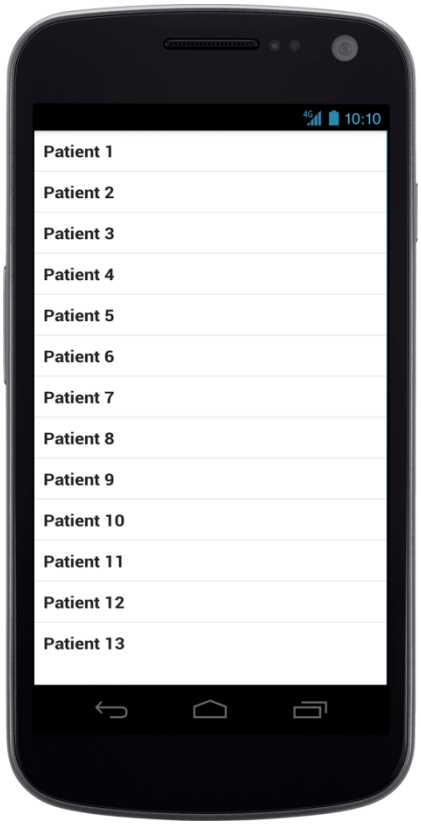
Δεύτερο πλήκτρο που παρατηρούμε είναι το «Start Dummy Server». Όπως αναφερθήκαμε λόγω της έλλειψης συστήματος το οποίο στέλνει ασύρματα τα δεδομένα θα χρησιμοποιηθεί ένα αρχείο κειμένου. Αυτό το αρχείο κειμένου διαβάζεται γραμμή - γραμμή και στέλνεται με ρυθμό 256 δείγματα/sec μέσω socket σε μια δεύτερη συσκευή android. Άρα με την χρήση του πλήκτρου δημιουργούμε έναν εικονικό server ο οποίος στέλνει τα δεδομένα σε πραγματικό χρόνο με την χρήση sockets.

Τέλος, το τρίτο κουμπί «Patient List» ανοίγει την τοπική βάση δεδομένων σε SQLite η οποία περιέχει ασθενείς, ιατρούς και αποθηκευμένα ηλεκτροκαρδιογραφήματα.

Πατώντας κάθε πλήκτρο δημιουργείται το κατάλληλο Intent το οποίο ξεκινάει την αντίστοιχη Διεργασία όπως φαίνεται και στον κώδικα.

5.1.2 Λίστα Ασθενών

Όπως αναφέρθηκε πατώντας το πλήκτρο «Patient List» ανοίγει η τοπική βάση δεδομένων. Η κλάση που καλείται είναι η AndroidDatabase.java η οποία αρχικοποιεί ένα στιγμιότυπο της κλάσης newSQLiteHelper.java, η οποία είναι μία κλάση που επεκτείνει (extends) την SQLiteHelper. Η newSQLiteHelper περιέχει τις μεθόδους για εισαγωγή, διαγραφή, προσπέλαση και ανανέωση στοιχείων των οντοτήτων.



Οθόνη Λίστα Ασθενών

Η AndroidDatabase.java είναι μία Activity και όπως έχουμε περιγράψει όταν καλείται η μέθοδος onCreate(). Με αυτή τη μέθοδο έχουμε φορτώνουμε δεδομένα στην βάση δεδομένων χρησιμοποιώντας τις μεθόδους CreatePatient και CreateECG της newSQLiteHelper.

```
Public class AndroidDatabase extends ListActivity implements
OnItemClickListener{

newSQLiteHelper db = new newSQLiteHelper(this);
List<Patient> list;
ArrayAdapter myAdapter;


    @Override
    public void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);

        setContentView(R.layout.database);

        // drop this database if already exists
        db.onUpgrade(db.getWritableDatabase(), 1, 2);

        db.createPatient(new Patient("Patient 1", "Doctor 1"));
        db.createPatient(new Patient("Patient 2", "Doctor 1"));
        db.createPatient(new Patient("Patient 3", "Doctor 1"));
        db.createPatient(new Patient("Patient 4", "Doctor 4"));
        db.createPatient(new Patient("Patient 5", "Doctor 4"));
        db.createPatient(new Patient("Patient 6", "Doctor 4"));
        db.createPatient(new Patient("Patient 7", "Doctor 4"));
        db.createPatient(new Patient("Patient 8", "Doctor 4"));
        db.createPatient(new Patient("Patient 9", "Doctor 3"));
        db.createPatient(new Patient("Patient 10", "Doctor 3"));
        db.createPatient(new Patient("Patient 11", "Doctor 2"));
        db.createPatient(new Patient("Patient 12", "Doctor 2"));
        db.createPatient(new Patient("Patient 13", "Doctor 5"));
        db.createECG(1,1);
        db.createECG(1,2);
        db.createECG(2,1);
        db.createECG(2,2);


        // get all patients
        list = db.getAllPatients();
        List listName = new ArrayList();

        for (int i = 0; i < list.size(); i++) {
          listName.add(i, list.get(i).getName());
        }

        myAdapter = new ArrayAdapter(this, R.layout.row_layout,
R.id.listText, listName);
        getListView().setOnItemClickListener(this);
        setListAdapter(myAdapter);
    }

    @Override
    public void onItemClick(AdapterView arg0, View arg1, int arg2,
long arg3) {
        // start PatientActivity with extras the patient id
        Intent intent = new Intent(this, PatientActivity.class);
        intent.putExtra("patient", list.get(arg2).getId());
        startActivityForResult(intent, 1);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        // get all patients again, because something changed
        list = db.getAllPatients();

        List<String> listName = new ArrayList<String>();
```

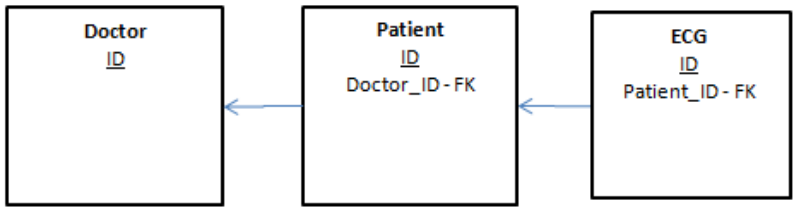
```
        for (int i = 0; i < list.size(); i++) {
            listName.add(i, list.get(i).getName());
        }

        myAdapter = new ArrayAdapter(this, R.layout.row_layout,
R.id.listText, listName);
        getListView().setOnItemClickListener(this);
        setListAdapter(myAdapter);
    }

    @Override
    public void onBackPressed() {
        db.close();
        finish(); // finish activity
    }
}
```

Σημαντική μέθοδος της newSQLite είναι η getAllPatients η οποία χρησιμοποιεί ένα ερώτημα (query) για να προσπελάσει τα στοιχεία του πίνακα ασθενών της βάσης και να τα αποθηκεύσει σε μεταβλητές για να προβληθούν και να επεξεργαστούν περεταίρω. Επίσης παρατηρούμε τις μεθόδους onItemClick και onActivityResult.

Η πρώτη καλείται όταν πατηθεί ένας ασθενής από την λίστα που φαίνεται στην οθόνη και στέλνει μέσω Intent το id του ασθενή στην activity PatientActivity στην οποία εμφανίζονται αναλυτικότερα οι πληροφορίες του ασθενή.

Η onActivityResult καλείται όταν εκτελεστεί μία άλλη activity το αποτέλεσμα της επιστρέφεται μέσω intent. Στην συγκεκριμένη περίπτωση αν υπάρξει αλλαγή στην λίστα με τους ασθενής θα καλεστεί η onActivityResult και θα ανανεωθεί η λίστα.

Στην συνέχεια παρουσιάζεται μια απλή σχεσιακή βάση δεδομένων όπως σχεδιάστηκε για την υποστήριξη της εφαρμογής.



Σχήμα 5.1 - ERD Διάγραμμα Οντοτήτων - Συσχετίσεων

Η οντότητα Ιατρός αναπαριστά τους ιατρούς όπως αυτοί αποθηκεύονται στο σύστημα με τις παρακάτω ιδιότητες – πεδία.

Οντότητα Doctor (Ιατρός)

Πεδίο	**Περιγραφή**
ID – PK	Κωδικός Ιατρού
Name	Όνομα Ιατρού
Phone Number	Τηλέφωνο Ιατρού
Address	Διεύθυνση Ιατρού
Email	Email Ιατρού

Το πεδίο κωδικός ιατρού αφορά έναν αριθμό κλειδί που επιτρέπει να αναγνωρίζεται κάθε γιατρός. Ο κωδικός αυτός πρέπει να είναι μοναδικός σε κάθε γιατρό . Στην παρούσα διπλωματική επιλέχθηκε ο αύξοντας αριθμός με τον οποίο καταχωρήθηκαν στην βάση δεδομένων οι γιατροί.

Η οντότητα Ασθενής αναπαριστά τους ασθενείς όπως αυτοί αποθηκεύονται στο σύστημα με τις παρακάτω ιδιότητες – πεδία.

Οντότητα Patient (Ασθενής)

Πεδίο	**Περιγραφή**
ID – PK	Κωδικός Ασθενή
Name	Όνομα Ασθενή
Phone Number	Τηλέφωνο Ασθενή
Address	Διεύθυνση Ασθενή
Email	Email Ασθενή
AMKA	AMKA Ασθενή
Doctor_ID - FK	Συσχέτιση με Ιατρό

Όμοια όπως ο κωδικός του γιατρού ένας ασθενής χρειάζεται έναν μοναδικό αναγνωριστικό κωδικό. Για λόγους ευκολίας επιλέχθηκε πάλι ο αύξοντας αριθμός καταχώρησης. Ο πεδίο Doctor_ID είναι εξωτερικό κλειδί (foreign key) και

χρησιμοποιείται για να δείξει την συσχέτιση ενός ασθενή με έναν γιατρό. Με τον τρόπο που σχεδιάστηκε το σύστημα ένας ασθενής επιβλέπεται από έναν γιατρό, αλλά ένας γιατρός μπορεί να επιβλέπει πολλούς ασθενείς. Αυτή η σχέση γιατρός – ασθενής) λέγεται σχέση 1-n. Σε ένα νοσοκομείο αυτή η σχέση θα μπορούσε να είναι n-n δηλαδή να υπάρχουν πολλοί γιατροί για κάθε ασθενή αλλά αυτό ξεφεύγει από τον σκοπό αυτής της εργασίας.

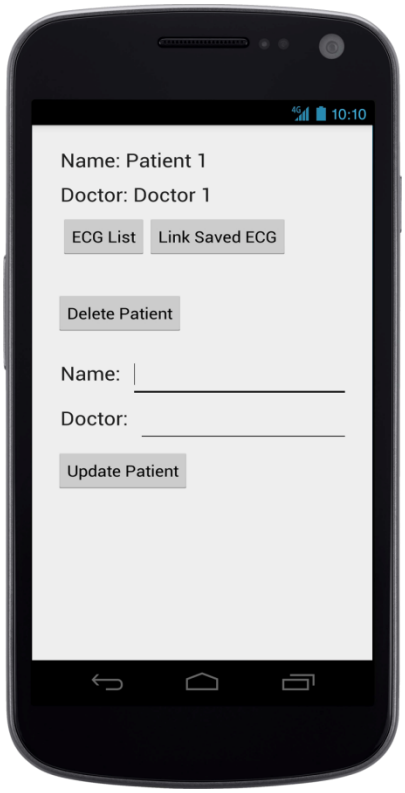
Οντότητα ECG

Πεδίο	**Περιγραφή**
ID – PK	Κωδικός ECG
Path	Διαδρομή Αποθήκευσης
Date	Ημερομηνία Καταγραφής
Patient_ID – FK	Συσχέτιση με Ασθενή

Η συσχέτιση με τον ασθενή καθορίζεται από το πεδίο Patient_ID το οποίο είναι εξωτερικό κλειδί και δηλώνει ότι κάθε ασθενής έχει πολλά καταγεγραμμένα ηλεκτροκαρδιογραφήματα και κάθε ηλεκτροκαρδιογράφημα προέρχεται από έναν συγκεκριμένο ασθενή. Το πεδίο path περιέχει την διαδρομή που είναι αποθηκευμένο το ηλεκτροκαρδιογράφημα, ώστε να ξέρει η εφαρμογή από πού θα αντλήσει τα δεδομένα. Μία εναλλακτική προσέγγιση θα ήταν να αποθηκευόταν το ηλεκτροκαρδιογράφημα σε μορφή δεδομένων κατευθείαν στην βάση. Ο κωδικός ECG είναι μοναδικός ώστε να γίνεται αναγνωρίσιμο ένα ηλεκτροκαρδιογράφημα από το σύστημα και να ξέρουμε κάθε φορά σε ποιον ασθενή ανήκει.

5.1.3 Στοιχεία Ασθενή

Αυτή η οθόνη παρέχει επιπλέον πληροφορίες για τον ασθενή. Όπως αναφέρθηκε με την επιλογή ενός ασθενή με απλό πάτημα από την λίστα ασθενών ο χρήστης καταλήγει στην παρακάτω οθόνη.



Οθόνη Ασθενής

Παρουσιάζονται κάποιες βασικές πληροφορίες όπως το όνομα του ασθενή και ο γιατρός που τον παρατηρεί. Για την ανάκτηση ενός ασθενής πραγματοποιείται ένα ερώτημα (query) στην βάση όπως φαίνεται παρακάτω:

```
public Patient readPatient(int id) {
    // get reference of the PatientDB database
    SQLiteDatabase db = this.getReadableDatabase();

    // get patient query
    Cursor cursor = db.query(table_PATIENTS, // a. table
```

```
            COLUMNS, " id = ?", new String[] { String.valueOf(id) },
null, null, null, null);

    // if results !=null, parse the first one
    if (cursor != null)
        cursor.moveToFirst();

    Patient patient = new Patient();
    patient.setId(Integer.parseInt(cursor.getString(0)));
    patient.setName(cursor.getString(1));
    patient.setDoctor(cursor.getString(2));

    return patient;
```

5.1.4 Λειτουργία Ενημέρωσης και Διαγραφής Ασθενή

Παρατηρούμε επίσης 4 κουμπιά καθώς και 2 πεδία κειμένου. Ο χρήστης μπορεί να διορθώσει τα στοιχεία του ασθενή και με το πλήκτρο Update Patient ενημερώνεται στην βάση δεδομένων η αντίστοιχη καταχώριση ασθενής. Η εφαρμογή εμφανίζει ειδοποιητικό μήνυμα τύπου «Toast» ότι ο ασθενής ενημερώθηκε.

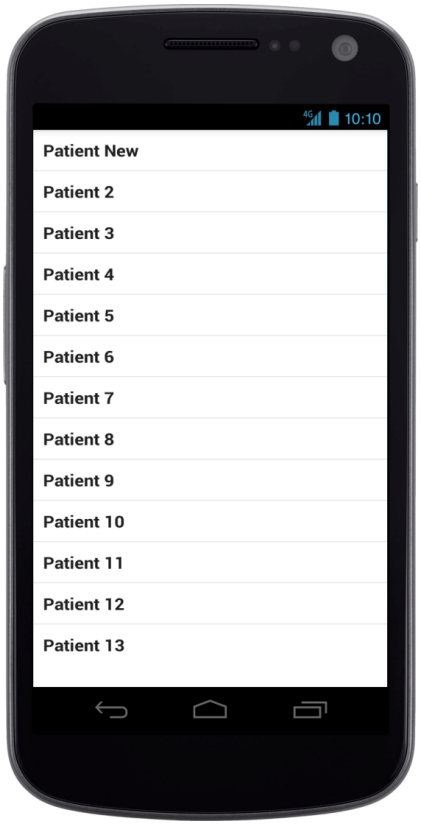
Το πλήκτρο Delete Patient προσφέρει στον χρήστη την ικανότητα να διαγράψει τον επιλεγμένο ασθενή από την βάση δεδομένων. Κάθε αλλαγή που πραγματοποιείται από τον χρήστη ενημερώνει την λίστα με τους ασθενής στην προηγούμενη οθόνη. Η εφαρμογή εμφανίζει ειδοποιητικό μήνυμα τύπου «Toast» ότι ο ασθενής διαγράφτηκε επιτυχώς.

Η υλοποίηση παρουσιάζεται παρακάτω:

```
public void update(View v) {
    Toast.makeText(getApplicationContext(), "This patient is
updated.", Toast.LENGTH_SHORT).show();
    selectedPatient.setName(((EditText)
findViewById(R.id.nameEdit)).getText().toString());
    selectedPatient.setDoctor(((EditText)
findViewById(R.id.doctorEdit)).getText().toString());

    // update patient with changes
    db.updatePatient(selectedPatient);
    finish();
}

public void delete(View v) {
    Toast.makeText(getApplicationContext(), "This patient is
deleted.", Toast.LENGTH_SHORT).show();
```

```
        // delete selected patient
        db.deletePatient(selectedPatient);
        finish();
    }
```

Γράφοντας στο πεδίο Name το νέο όνομα του ασθενή και επιστρέφοντας στην λίστα με τους ασθενής παρατηρούμε ότι η λίστα ενημερώθηκε επιτυχώς.



Οθόνη μετά την Ενημέρωση Ασθενή

Ο χρήστης επιλέγοντας έναν ασθενή και στη συνέχεια το κουμπί ECG List εμφανίζονται τα καταγεγραμμένα ηλεκτροκαρδιογραφήματα και η διαδρομή που αποθηκεύονται. Ο προκαθορισμένος φάκελος λέγεται ecg data και βρίσκεται στην εσωτερική μνήμη της συσκευής. Με απλό πάτημα πάνω σε ένα ηλεκτροκαρδιογράφημα έχουμε την δυνατότητα να ανακτήσουμε τις μετρήσεις. Τα ecg παρουσιάζονται σε μία List View και με την χρήση του onClickListener μπορούμε να ξέρουμε ποιο σήμα επέλεξε ο χρήστης.

5.1.5 Λίστα Ηλεκτροκαρδιογραφημάτων

Οθόνη με την λίστα ηλεκτροκαρδιογραφημάτων

Όπως έχουμε αναφέρει τα δεδομένα σήματος βρίσκονται σε μορφή αρχείου κειμένου το οποίο έχει μία στήλη για τον χρόνο και μία άλλη για το πλάτος. Με την παρακάτω μέθοδο της κλάσης PatientActivity.java διαβάζεται το αρχείο ανά γραμμή και χωρίζει τις δύο στήλες και τις φορτώνει σε μία ArrayList.

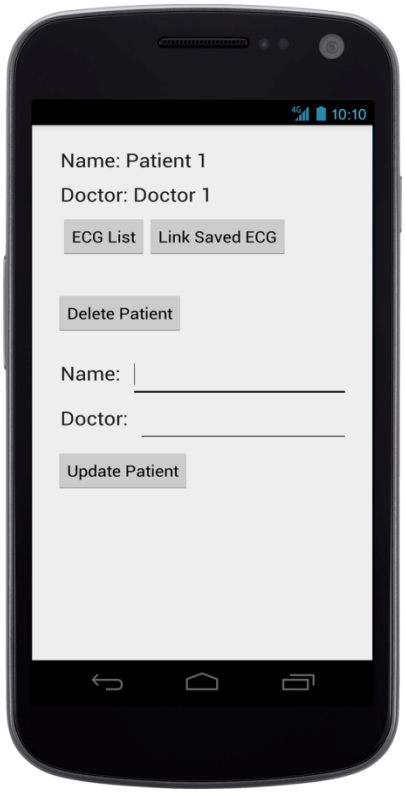
```
lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {

    @Override
    public void onItemClick(AdapterView<?> arg0, View arg1, int arg2,
                            long arg3) {
        Toast.makeText(getApplicationContext(), "ECG Data Loaded",
Toast.LENGTH_SHORT).show();

        String selected_ecg =
(lv.getItemAtPosition(arg2)).toString();
```

```
        //clear data before loading
        xvalues_list.clear();
        yvalues_list.clear();

        StringBuilder text = new StringBuilder();
        try {
            File file = new File(selected_ecg);
            BufferedReader br = new BufferedReader(new
FileReader(file));
            String line;
            while ((line = br.readLine()) != null) {
                text.append(line);
                text.append('\n');
                String[] line_parts = line.split(" ");
                xvalues_list.add(Float.parseFloat(line_parts[0]));
                yvalues_list.add(Float.parseFloat(line_parts[1]));
            }
            br.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
});
```

Στην συνέχεια μπορούμε να απεικονίσουμε το σήμα στην οθόνη του κινητού όπως θα αναφερθεί παρακάτω.

5.1.6 Σύνδεση ηλεκτροκαρδιογραφήματος με ασθενή

Τελευταίο πλήκτρο είναι το «Link Saved ECG» το οποίο αντιστοιχίζει ένα αποθηκευμένο ECG σε έναν ασθενή. Ο χρήστης βρισκόμενος στον ασθενή που θέλει να κάνει την αντιστοίχιση πιέζει το πλήκτρο «Link Saved ECG».



Οθόνη για την σύνδεση του ECG με τον επιλεγμένο ασθενή

Στην συνέχεια επιλέγει το αποθηκευμένο αρχείο κειμένου από την συσκευή του και η εφαρμογή το μεταφέρει στον προκαθορισμένο φάκελο ecg data και το μετονομάζει κατάλληλα χρησιμοποιώντας το πρόθεμα ecg-save-, τον κωδικό του ασθενή σε συνδυασμό με το αγγλικό γράμμα p και τον αύξοντα αριθμό αποθήκευσης διαχωρισμένο με μία παύλα (-) όπως εμφανίζεται στην οθόνη. Π.χ. για τον ασθενή με ID 1 το πρώτο σήμα που αποθηκεύεται ως ecg-save-p1-1.txt ενώ το 3^{ο} ως ecg-save-p1-3.txt

Οθόνη επιλογής αρχείο κειμένου από τον χώρο αποθήκευσης της συσκευής

Ενημερωμένη λίστα μετά το πάτημα του πλήκτρου «ECG Saved Link»

Η μετακίνηση και η μετονομασία του αρχείου καθώς και η σύνδεση του ECG με έναν ασθενή γίνεται από το συγκεκριμένο κομμάτι κώδικα. Η μέθοδος getPath λαμβάνει την τωρινή αποθηκευμένη θέση του αρχείου και καλώντας την μέθοδο renameTo μετονομάζουμε κατάλληλα το αρχείο. Παρατηρούμε την μεταβλητή newaa η οποία λαμβάνει τον επόμενο αύξοντα αριθμό για να τα διατηρηθεί η σωστή σειρά αποθήκευσης του σήματος. Η μέθοδο createECG εισάγει μια καινούρια γραμμή στην στον πίνακα με τα ηλεκτροκαρδιογραφήματα στην βάση δεδομένων.

Από την κλάση PatientActivity.java

```
if (requestCode == REQUESTCODE_PICK_FILE
        && resultCode == Activity.RESULT_OK) {
    uri = data.getData();
    Log.d("", "URI= " + uri);
    if (uri != null) {
        String str = uri.getPath();
        File from = new File(str);
        int newaa=listECG.size()+1;
        File renamed =new
File(Environment.getExternalStorageDirectory().getAbsolutePath() +
"/ecg data/" +  "ecg-save-p" +id+ "-"+newaa+ ".txt");
        from.renameTo(renamed);
        db.createECG(id, newaa);
    }
}
```

5.1.7 Εξυπηρετητής (Server)

Η λειτουργία “Start Dummy Server” μετατρέπει το κινητό σε έναν εξυπηρετητή ο οποίος αποστέλλει διαρκώς τα δεδομένα του ηλεκτροκαρδιογραφήματος μέσω socket όπως αναφέραμε. Πατώντας το αντίστοιχο πλήκτρο από το κεντρικό μενού καλείται η παρακάτω κλάση.

```
public class ServerActivity extends Activity {

    private TextView serverStatus;

    // DEFAULT IP
    public static String SERVERIP = "localhost";
    //public static String SERVERIP = "10.0.2.15";

    // DESIGNATE A PORT
    public static final int SERVERPORT = 8080;
    private Handler handler = new Handler();
    private ServerSocket serverSocket;
    private static final String TAG = "BroadcastService";
    Float xvalue=0f;
    Float yvalue=0f;



    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.server);
        serverStatus = (TextView) findViewById(R.id.server_status);

        SERVERIP = getLocalIpAddress();

        Thread fst = new Thread(new ServerThread());
        fst.start();
    }


    public class ServerThread implements Runnable {
        public void run() {
            try {
                if (SERVERIP != null) {
                    handler.post(new Runnable() {
                        @Override
                        public void run() {
                            serverStatus.setText("Listening on IP: "
+ SERVERIP);
                        }
                    });
                    serverSocket = new ServerSocket(SERVERPORT);
                    while (true) {
                        // LISTEN FOR INCOMING CLIENTS
                        Socket client = serverSocket.accept();
                        handler.post(new Runnable() {
                            @Override
                            public void run() {
                                serverStatus.setText("Connected.");
                            }
                        });

                        try {

                            String line = null;
                            OutputStream out_2 =
client.getOutputStream();
```

```
                            DataOutputStream dos = new
DataOutputStream(out_2);
                            InputStream is;
                            BufferedReader br;
                            is =
getResources().openRawResource(R.raw.ecgsyn2);
                            br= new BufferedReader(new
InputStreamReader(is));
                            StringBuilder text = new StringBuilder();

                            try {
                                int i=0;
                                if ((line = br.readLine()) != null) {

                                    while (i<255) {
                                        line = br.readLine();
                                        text.append(line);
                                        text.append('\n');
                                        String[]
line_parts=line.split(" ");
                                        xvalue=
Float.parseFloat(line_parts[0]);
                                        yvalue=
Float.parseFloat(line_parts[1]);
                                        dos.writeFloat(xvalue);
                                        dos.writeFloat(yvalue);
                                        i++;
                                    }
                                }
                                else {br.close();}

                            }
                            catch (IOException e) {
                                e.printStackTrace();}



                            dos.writeFloat(xvalue);
                            dos.flush();
                            Log.d("serveractivity",
Float.toString(xvalue));
                            dos.close();



                            break;
                        } catch (Exception e) {
                            handler.post(new Runnable() {
                                @Override
                                public void run() {
                                    serverStatus.setText("Oops.
Connection interrupted. Please reconnect your phones.");
                                }
                            });
                            e.printStackTrace();
                        }
                    }
                } else {
```

```
                    handler.post(new Runnable() {
                        @Override
                        public void run() {
                            serverStatus.setText("Couldn't detect
internet connection.");
                        }
                    });
                }
            } catch (Exception e) {
                handler.post(new Runnable() {
                    @Override
                    public void run() {
                        serverStatus.setText("Error");
                    }
                });
                e.printStackTrace();
            }
        }
    }

    // GETS THE IP ADDRESS OF YOUR PHONE'S NETWORK
    private String getLocalIpAddress() {
        try {
            for (Enumeration<NetworkInterface> en =
NetworkInterface.getNetworkInterfaces(); en.hasMoreElements();) {
                NetworkInterface intf = en.nextElement();
                for (Enumeration<InetAddress> enumIpAddr =
intf.getInetAddresses(); enumIpAddr.hasMoreElements();) {
                    InetAddress inetAddress =
enumIpAddr.nextElement();
                    if (!inetAddress.isLoopbackAddress() &&
inetAddress instanceof Inet4Address) { return
inetAddress.getHostAddress().toString(); }
                }
            }
        } catch (SocketException ex) {
            Log.e("ServerActivity", ex.toString());
        }
        return null;
    }

    @Override
    protected void onStop() {
        super.onStop();
        try {
            // MAKE SURE YOU CLOSE THE SOCKET UPON EXITING
            serverSocket.close();
            finish();// try to go backtomainactivity
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

Τα παρακάτω βήματα γίνονται όταν δημιουργηθεί μία σύνδεση μέσω TCP Socket.

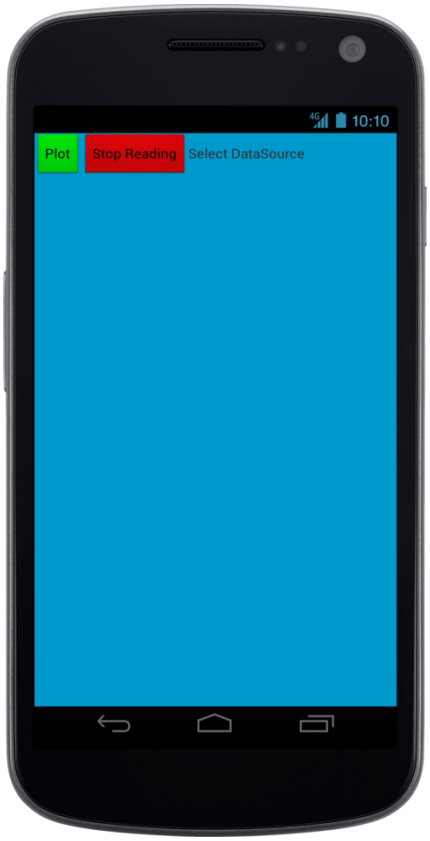
- Ο server αρχικοποιεί ένα στιγμιότυπο του αντικειμένου ServerSocket, καθορίζοντας την θύρα μέσω της οποίας θα γίνει η σύνδεση.
- Ο server καλεί την μέθοδο accept() της κλάσης ServerSocket. Αυτή η μέθοδος αναμένει την σύνδεση του client στην συγκεκριμένη θύρα.
- Στην συνέχεια ο client δημιουργεί ένα στιγμιότυπο Socket το οποίο δηλώνει το όνομα του server και την θύρα.
- Στην γίνεται προσπάθεια σύνδεσης στο συγκεκριμένο server και θύρα. Αν είναι επιτυχής ο client έχει ένα αντικείμενο socket το οποίο μπορεί να επικοινωνήσει με τον server.
- Στην μεριά του server η μέθοδος accept() επιστρέφει μία αναφορά (reference) στο νέο socket το οποίο συνδέεται με τον client.
- Αφότου γίνουν οι συνδέσεις, η επικοινωνία γίνεται μέσω I/O streams. Κάθε socket έχει OutputStream και InputStream. Το InputStream του client συνδέεται στο OutputStream του server και αντίστοιχα το OutputStream του client συνδέεται στο InputStream του server.

Η οθόνη που εμφανίζεται στην οθόνη είναι σχετικά απλή και το μόνο που εμφανίζει είναι η διεύθυνση ip στην οποία πρέπει να συνδεθεί ο client. Όταν ο client συνδεθεί εμφανίζεται μήνυμα Connected στην θέση της διεύθυνσης ip. Σε περιπτώσεις διακοπής της σύνδεσης ή απώλειας σύνδεσης δικτύου εμφανίζονται κατάλληλα μηνύματα.

Οθόνη Server

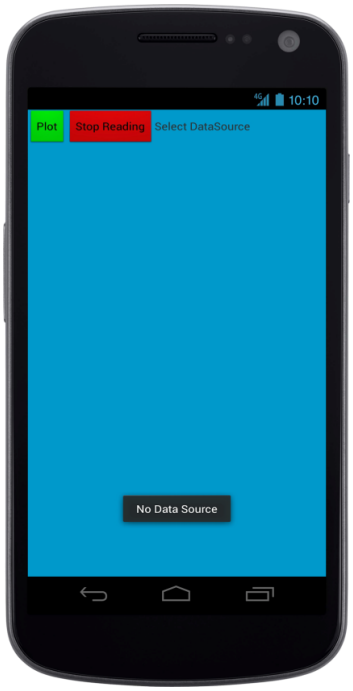
5.1.8 Απεικόνιση Δεδομένων Ηλεκτροκαρδιογραφήματος

Η κυριότερη λειτουργία της εφαρμογής είναι η απεικόνιση του σήματος. Από το κεντρικό μενού της εφαρμογής πατώντας το πλήκτρο “ECG” ο χρήστης μεταβαίνει στην παρακάτω οθόνη.



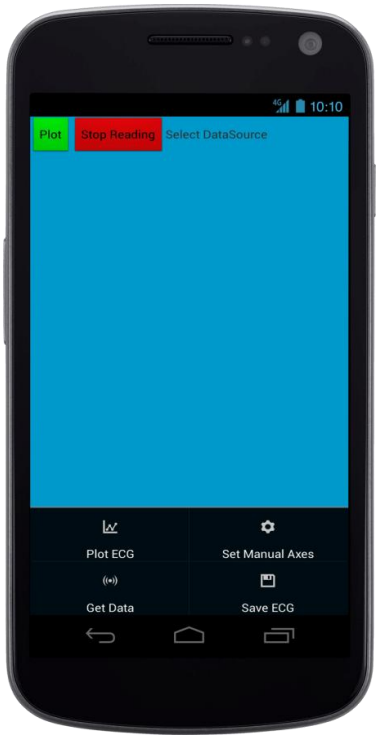
Βασική Οθόνη Απεικόνισης

Ένα μήνυμα εμφανίζεται πάνω δεξιά και δηλώνει στον χρήστη να διαλέξει την πηγή των δεδομένων. Επιλέγοντας το πράσινο πλήκτρο Plot χωρίς να έχει γίνει επιλογή από πού θα αντληθούν τα δεδομένα εμφανίζεται ειδοποιητικό μήνυμα «No Data Source” όπως φαίνεται παρακάτω.



Οθόνη – Μήνυμα σφάλματος (No Data Source)

Πατώντας το πλήκτρο «menu» της συσκευής εμφανίζονται 4 νέα πλήκτρα. Το πλήκτρο «Get Data» καθορίζει από πού θα ληφθούν τα δεδομένα. Πιέζοντας το, εμφανίζεται η παρακάτω οθόνη που έχει ακόμα 4 επιλογές.



Οθόνη Απεικόνισης με λειτουργίες πλήκτρων

Οθόνη επιλογής προέλευσης δεδομένων

Στην συνέχεια αναλύουμε τις διαφορετικές περιπτώσεις λήψης του ηλεκτροκαρδιογραφήματος. Οι επιλογές να αντλήσουμε δεδομένα είναι οι εξής:

- From Wifi: Τα δεδομένα έρχονται από έναν εξυπηρετητή με την χρήση socket όπως έχει αναφερθεί.
- From Patient List: Τα δεδομένα που θα χρησιμοποιηθούν είναι αυτά που επιλέχθηκαν από την βάση δεδομένων των ασθενών όπως αναφέρθηκε προηγουμένως
- Demo Data: Γίνεται προσομοίωση ενός καταγεγραμμένου ηλεκτροκαρδιογραφήματος σε πραγματικό χρόνο
- From .txt: Τα δεδομένα που θα χρησιμοποιηθούν για την απεικόνιση, επιλέγονται από ένα αρχείο .txt αρκεί να πληροί συγκεκριμένη μορφοποίηση. Τα δεδομένα πρέπει να είναι σε δύο στήλες, η πρώτη του χρόνου και η δεύτερη του πλάτους. Κάθε ζευγάρι τιμών θα είναι σε μία νέα γραμμή στο αρχείο κειμένου και θα χωρίζονται με ένα κενό διάστημα.

Αναλυτικά

From Wifi

Με την επιλογή From WiFi ανοίγει μία νέα οθόνη. Αποτελείται από ένα πεδίο κειμένου στο οποίο θα πρέπει να πληκτρολογηθεί η διεύθυνση του server και ένα πλήκτρο με το οποίο ξεκινά ένα service που τρέχει στο παρασκήνιο και λαμβάνει συνεχώς δεδομένα από τον server. Κάθε φορά που διαβάζονται δεδομένα προστίθενται στις συνδεδεμένες λίστες οι οποίες χρησιμοποιούνται στην συνέχεια για την απεικόνιση των δεδομενων μέσω της κλάσης Plot2d_array.java. Παρακάτω εμφανίζεται η κλάση που καλείται πατώντας το «Connect» και η οθόνη για την πληκτρολόγηση της διεύθυνσης IP.

Κλάση ClientService.java

```
public class ClientService extends Service {
```

```
    private String serverIpAddress = "";
    private boolean connected = false;
    private Handler handler = new Handler();
    private static final String TAG = "BroadcastService";
    public static final String BROADCAST_ACTION =
"com.ecg.pantelis.ecg_test";
    Intent intent;
    public float f1; //inputstream xvalue
    public float f2; //inputstram yvalue


    @Override

    public int onStartCommand(Intent intent, int flags, int startId)
{
        // For time consuming an long tasks you can launch a new
thread here...
        serverIpAddress = intent.getStringExtra("serverIP");

        handler.removeCallbacks(sendUpdatesToUI);

        if (!serverIpAddress.equals("")) {
            Log.d("ClientService","IP is " +serverIpAddress);
            Thread cThread = new Thread(new ClientThread());
            cThread.start();

        }
        else Log.d("ClientService","Error with IP");
        return 1;
    }


    public IBinder onBind(Intent intent) {

        throw new UnsupportedOperationException("Not yet
implemented");

    }

    public class ClientThread implements Runnable {

        public void run() {
            try {
                InetAddress serverAddr =
InetAddress.getByName(serverIpAddress);
                Log.d("ClientActivity", "C: Connecting...");
                Socket socket = new Socket(serverAddr,
ServerActivity.SERVERPORT);
                Context context = getApplicationContext();

                connected = true;
                while (connected) {
                    try {
                        Log.d("ClientActivity", "C: Sending
command.");

                        DataInputStream dis = new
```

```
DataInputStream(socket.getInputStream());

                        while (f1!=100f){
                        f1 = dis.readFloat();
                        f2 = dis.readFloat();
                            System.out.println("PI=" + f1);
                            DisplayLoggingInfo();
                        }
                        dis.close();
                        Log.d("ClientActivity", "C: Sent.");
                    } catch (Exception e) {
                        Log.e("ClientActivity", "S: Error", e);
                    }
                }
                socket.close();
                Log.d("ClientActivity", "C: Closed.");
            } catch (Exception e) {
                Log.e("ClientActivity", "C: Error", e);
                connected = false;
            }
        }
    }

    @Override
    public void onCreate() {
        super.onCreate();
        intent = new Intent(BROADCAST_ACTION);
    }

    private Runnable sendUpdatesToUI = new Runnable() {
        public void run() {
            DisplayLoggingInfo();
        }
    };

    private void DisplayLoggingInfo() {
        Log.d(TAG, "entered DisplayLoggingInfo");
        intent.putExtra("service_data1", f1);
        intent.putExtra("service_data2", f2);
        sendBroadcast(intent);
    }
}
```

Απεικόνιση

Στην συνέχεια επιλέγοντας το πράσινο πλήκτρο Plot ή το Plot ECG από το αναδυόμενο μενού απεικονίζουμε το ECG σε πραγματικό χρόνο στην οθόνη του κινητού αν έχουμε επιλέξει τις επιλογές From WiFi ή Demo Data οι οποίες προσωμοιώνουν ένα ασύρματο σύστημα μετάδοσης δεδομένων ηλεκτροκαρδιογραφήματος. Αν επιλεχτεί να διαβαστεί αποθηκευμένο αρχείο τότε εμφανίζεται όλο το σήμα και ο χρήστης έχει δυνατότητα να μετακινήσει το γράφημα με το δάχτυλο του.

Οθόνη Απεικόνισης δεδομένων ECG

Η κλάση που είναι υπεύθυνη για την απεικόνιση του σήματος είναι η Plot2d_array.java και παίρνει σαν παραμέτρους δύο λίστες με αριθμούς float. Στην συνέχεια με την χρήση ενός thread από την MainActivity.java ανανεώνεται η οθόνη ανά δευτερόλεπτο. Επίσης οι μέθοδοι xvaluesInPixels και yvaluesInPixels μετατρέπουν τα δεδομένα σε pixels κατάλληλα ώστε να χωράνε στην οθόνη του κινητού και να είναι ευδιάκριτο το σήμα. Οι μέθοδοι locxAxisInPixels και locyAxisInPixels καθορίζουν τα σημεία των αξόνων ανάλογα με τις τιμές που έχουν εισαχθεί.

```
public class plot2d_array extends View {

    private Paint paint;
    private List<Float> xvalues = new ArrayList<Float>();
    private List<Float> yvalues= new ArrayList<Float>();
    private float maxx,maxy,minx,miny,locxAxis,locyAxis;
    private int axes = 1;
    public int distance=0;
    public int auto_manual;
    public int scrollspeed=0;
    boolean land_flag=false;
    boolean portrait_flag=false;
    boolean or_change=false;

   public plot2d_array(Context context, List<Float> xvalues,
List<Float> yvalues, int axes, int auto_manual) {
        super(context);
        this.xvalues=xvalues;
        this.yvalues=yvalues;
        this.axes=axes;
        paint = new Paint();
       Log.d("onDraw", "plot2dcalling");
        getAxes(xvalues, yvalues);
        this.auto_manual=auto_manual;

    }

    @Override
    protected void onDraw(Canvas canvas) {

        Log.d("onDraw", "redrawing");
        float canvasHeight = getHeight();
        float canvasWidth = getWidth();

        //orientation check
        if (or_change) {
            scrollspeed = scrollspeed -
(int)(canvasWidth*0.8/MainActivity.maxx);
            or_change=false;
        }else {scrollspeed = scrollspeed +
(int)(canvasWidth*0.8/MainActivity.maxx);}
        if (MainActivity.ScrollFlag==false){scrollspeed=0;}
        Log.d("onDraw",Boolean.toString(MainActivity.ScrollFlag));
        if (canvasWidth<canvasHeight) {
            portrait_flag=true;
        }
        if (canvasWidth>canvasHeight) {
            land_flag=true;
        }
        if(portrait_flag && land_flag){or_change=true;
            portrait_flag=false;
            land_flag=false;
        }

        //case 0: Axes from 1000 points
        //Case 1: Axes from all points
        //case 2: Axes from user
        //case 3: Axes from user v2
```

```
        switch(auto_manual) {
            case 0:
                minx=getMin_case0(xvalues);
                miny=getMin_case0(yvalues);
                maxx=getMax_case0(xvalues);
                maxy=getMax_case0(yvalues);
                break;
            case 1:
                minx=getMin(xvalues);
                miny=getMin(yvalues);
                maxx=getMax(xvalues);
                maxy=getMax(yvalues);
                break;
            case 2:
                minx=MainActivity.minx;
                miny=getMin(yvalues);
                maxx=MainActivity.maxx;
                maxy=getMax(yvalues);
                break;
            case 3:
                minx=MainActivity.minx;
                miny=MainActivity.miny;
                maxx=MainActivity.maxx;
                maxy=MainActivity.maxy;
                        }
```

```
        int[] xvaluesInPixels = toPixel(canvasWidth, minx, maxx,
xvalues);  //data to pixels
        int[] yvaluesInPixels = toPixel(canvasHeight, miny, maxy,
yvalues);
        int locxAxisInPixels = toPixelInt(canvasHeight, miny, maxy,
locxAxis); //Coordinates for axes
        int locyAxisInPixels = toPixelInt(canvasWidth, minx, maxx,
locyAxis);
        String xAxis = "x-axis";
        String yAxis = "y-axis";
        distance=0;
        distance=(int) GestureListener.distance; //scrolling
        checkBounds(minx - distance,canvasWidth);
        paint.setStrokeWidth(2);
        canvas.drawARGB(200, 200, 200, 200);

        //draw ECG
        for (int i = 0; i < xvalues.size()-1; i++) {
            paint.setColor(Color.RED);
            canvas.drawLine(xvaluesInPixels[i] - distance -
scrollspeed, canvasHeight - yvaluesInPixels[i], xvaluesInPixels[i +
1] - distance - scrollspeed, canvasHeight - yvaluesInPixels[i + 1],
paint);

        }

        //draw axes
        paint.setColor(Color.BLACK);
```

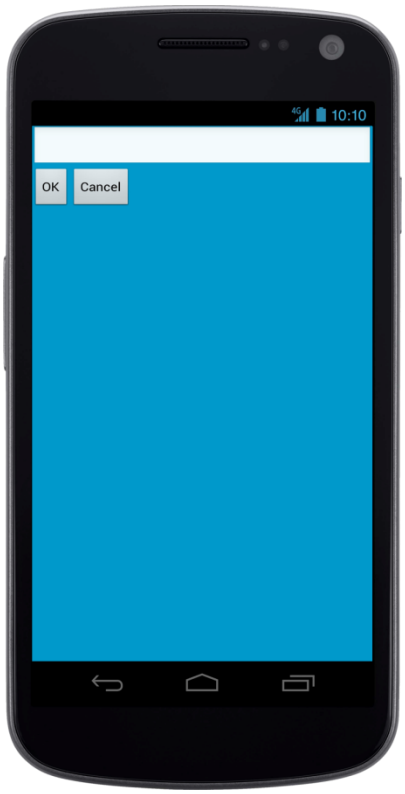
```
        canvas.drawLine(0, canvasHeight - locxAxisInPixels,
canvasWidth, canvasHeight - locxAxisInPixels, paint);

canvas.drawLine(locyAxisInPixels,0,locyAxisInPixels,canvasHeight,pain
t);

        //Automatic axes markings, modify n to control the number of
axes labels
        if (axes!=0){
            float temp = 0.0f;
            int n=3;
            paint.setTextAlign(Paint.Align.CENTER);
            paint.setTextSize(20.0f);
            for (int i=1;i<=n;i++){
                temp = Math.round(10*(minx+(i-1)*(maxx-minx)/n))/10;
                canvas.drawText(""+temp,
(float)toPixelInt(canvasWidth, minx, maxx, temp),canvasHeight-
locxAxisInPixels+20, paint);
                temp = Math.round(10*(miny+(i-1)*(maxy-miny)/n))/10;
                canvas.drawText(""+temp,
locyAxisInPixels+20,canvasHeight-(float)toPixelInt(canvasHeight,
miny, maxy, temp), paint);
            }
            canvas.drawText(""+maxx, (float)toPixelInt(canvasWidth,
minx, maxx, maxx),canvasHeight-locxAxisInPixels+20, paint);
            canvas.drawText(""+maxy,
locyAxisInPixels+20,canvasHeight-(float)toPixelInt(canvasHeight,
miny, maxy, maxy), paint);
        }
    }


    private int[] toPixel(float pixels, float min, float max,
List<Float> value) {

        double[] p = new double[value.size()];
        int[] pint = new int[value.size()];

        for (int i = 0; i < value.size(); i++) {
            p[i] = .1*pixels+((value.get(i)-min)/(max-
min))*.8*pixels;
            pint[i] = (int)p[i];
        }

        return (pint);
    }

    private void getAxes(List<Float> xvalues, List<Float> yvalues) {

        minx=getMin(xvalues);
        miny=getMin(yvalues);
        maxx=getMax(xvalues);
        maxy=getMax(yvalues);

        if (minx>=0)
            locyAxis=minx;
        else if (minx<0 && maxx>=0)
            locyAxis=0;
```

```
        else
            locyAxis=maxx;

        if (miny>=0)
            locxAxis=miny;
        else if (miny<0 && maxy>=0)
            locxAxis=0;
        else
            locxAxis=maxy;

    }

    private int toPixelInt(float pixels, float min, float max, float
value) {

        double p;
        int pint;
        p = .1*pixels+((value-min)/(max-min))*.8*pixels;
        pint = (int)p;
        return (pint);
    }

    private float getMax(List<Float> v) {
        float largest = v.get(0);
        for (int i = 0; i < v.size(); i++)
            if (v.get(i) > largest)
                largest = v.get(i);
        return largest;
    }

    private float getMin(List<Float> v) {
        float smallest = v.get(0);
        for (int i = 0; i < v.size(); i++)
            if (v.get(i) < smallest)
                smallest = v.get(i);
        return smallest;
    }

    private int checkBounds(float pixels,float w){

        minx=getMin(xvalues);

        if (pixels>w)
        distance=distance+200;
        return distance;
    }

    private float getMin_case0(List<Float> v) {
        float smallest = v.get(0);
        for (int i = 0; i < 1000; i++)
            if (v.get(i) < smallest)
                smallest = v.get(i);
        return smallest;
    }

    private float getMax_case0(List<Float> v) {
        float largest = v.get(0);
        for (int i = 0; i < 1000; i++)
```

```
            if (v.get(i) > largest)
                largest = v.get(i);
        return largest;
            }


}
```

Καθορισμός κλίμακας

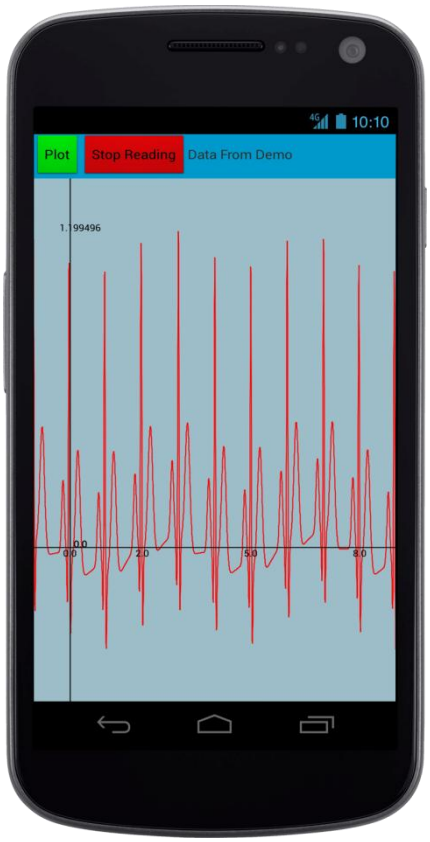
Ο χρήστης έχει τη δυνατότητα να αλλάξει την κλίμακα του σήματος, αλλάζοντας τον μέγιστο χρόνο που απεικονίζει η οθόνη του κινητού. Ο προκαθορισμένος χρόνος που εμφανίζεται στην οθόνη του κινητού είναι 2sec αλλά πατώντας το πλήκτρο menu της συσκευής και επιλέγοντας «Set Manual Axes» εμφανίζεται η παρακάτω οθόνη στην οποία μπορεί να αλλάξει το μέγεθος του άξονα του χρόνου.



Οθόνη επιλογής τιμής άξονα χρόνου

Επιλέγοντας τυχαία μία τιμή, π.χ. το 8 και πατώντας το Ok, το ECG ανανεώνεται αμέσως και εμφανίζεται η παρακάτω οθόνη. Με το πλήκτρο Cancel επιστρέφουμε στην οθόνη απεικόνισης χωρίς να γίνει καμία αλλαγή. Ο άξονας y'y του πλάτους προσαρμόζεται κατάλληλα κάθε φορά ανάλογα τα δεδομένα ακόμα σε περίπτωση αρνητικών τιμών.

Παρατηρούμε ότι στην οθόνη απεικονίζονται 8 δευτερόλεπτα σήματος αλλά το σήμα εξακολουθεί να εμφανίζεται ανά δευτερόλεπτο



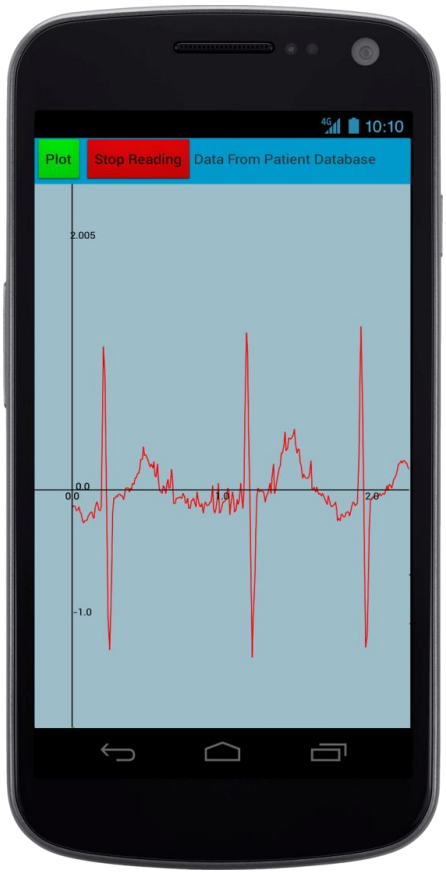
Οθόνη ECG μετά την αλλαγή κλίμακας

From Patient List

Όπως αναφέραμε ο γιατρός έχει την δυνατότητα να ανακτήσει ένα σήμα από την τοπική βάση δεδομένων. Επιλέγουμε το σήμα p1-2 του ασθενή 1 όπως φαίνεται παρακάτω.

Οθόνη με την λίστα ECG για έναν ασθενή

Πηγαίνοντας στην λειτουργία απεικόνισης και ρυθμίζοντας να ληφθούν τα δεδομένα από το φορτωμένο ηλεκτροκαρδιογράφημα εμφανίζεται το παρακάτω σήμα. Όπως αναφέραμε ο χρήστης έχει την δυνατότητα να μετακινήσει το σήμα ώστε να δει όλη την διάρκεια του με καλή ευκρίνεια ή ακόμα να αλλάξει την κλίμακα ώστε να κάνει μια πιο ειδική παρατήρηση, επιλέγοντας μια μικρή κλίμακα. Αν ο χρήστης επιλέξει μια μεγάλη κλίμακα θα μπορεί να δει μεγαλύτερο μέρος του σήματος αλλά δεν θα μπορεί να παρατηρήσει μικρές αλλαγές στο σήμα λόγω των περιορισμών της οθόνης των κινητών.



Οθόνη Απεικόνισης Δεδομένων ECG

Όπως είπαμε η φόρτωση των δεδομένων γίνεται σε δύο λίστες μέσω της PatientActivitity.java. Γι' αυτό τον λόγο ορίστηκαν δυο static ArrayLists (xvalues_list, yvalues_list) οι οποίες προσπελάζονται από την MainActivity όπως φαίνεται παρακάτω για να απεικονιστούν τα δεδομένα.

```
plot2d_array graph_patient = new plot2d_array(this,
PatientActivity.xvalues_list, PatientActivity.yvalues_list, 1, 2);
    buttonGraph.removeAllViews();
    buttonGraph.addView(buttons, lp);
    buttonGraph.addView(graph_patient, lp);
```

Βλέπουμε ότι το αντικείμενο plot2d_array που κατασκευάζεται από την MainActivity παίρνει τα εξής ορίσματα:

Τις δύο λίστες που περιέχουν τις τιμές του σήματος και δύο ακέραιους. Ο πρώτος καθορίζει αν η plot2d_array σχεδιάσει τους άξονες και ο δεύτερος καθορίζει την

περίπτωση που η plot2d_array καλείται με συγκεκριμένη κλίμακα (2sec στην περίπτωση μας)

Demo Data

Η επιλογή Demo Data επιτρέπει στην εφαρμογή να προσομοιώσει την καταγραφή του βιοσήματος από έναν ηλεκτροκαρδιογράφο. Αυτή η λειτουργία υλοποιήθηκε ως εξής:

Ένα αρχείο κειμένου περιέχει δύο στήλες δεδομένων χωρισμένες μεταξύ τους με ένα κενό χαρακτήρα. Η εφαρμογή όπως έχει αναφερθεί και στις προηγούμενες περιπτώσεις διαβάζει γραμμή – γραμμή και απεικονίζει τα δεδομένα. Για να προσομοιωθεί ο πραγματικός χρόνος χρησιμοποιείται ένα thread για να διαβάζει κάθε γραμμή σε συγκεκριμένο χρόνο.

Παρατηρώντας το αρχείο text βλέπουμε ότι για ένα δευτερόλεπτο έχουμε καταγραφή 256 σημείων. Για να μειωθεί η υπολογιστική ισχύς που καταναλώνει το νήμα, επιλέχτηκε να επαναλαμβάνεται κάθε 1 sec άρα θα πρέπει να διαβάζονται 256 σημεία. Αυτό φαίνεται και παρακάτω στην μέθοδο startRepeatingTask().

```
void startRepeatingTask() {
    Log.d("thread2","reading text data");
    dthreadflag =1;
    try {
        int i=0;
        if ((line = br.readLine()) != null) {

            while (i<255) {
                line = br.readLine();
                text.append(line);
                text.append('\n');
                String[] line_parts=line.split(" ");
                xvalues_list.add(Float.parseFloat(line_parts[0]));
                yvalues_list.add(Float.parseFloat(line_parts[1]));
                i++;
            }
        }
        else {br.close();}
        if (!interrupt_thread) {setContentView(buttonGraph);}
    }
    catch (IOException e) {
        e.printStackTrace();}
        }
```

Για να διαβαστεί ένα αρχείο χρησιμοποιούμε δύο αντικείμενα όπως φαίνεται παρακάτω

```
InputStream is;
BufferedReader br;
is = getResources().openRawResource(R.raw.ecgsyn2);
br= new BufferedReader(new InputStreamReader(is));
```

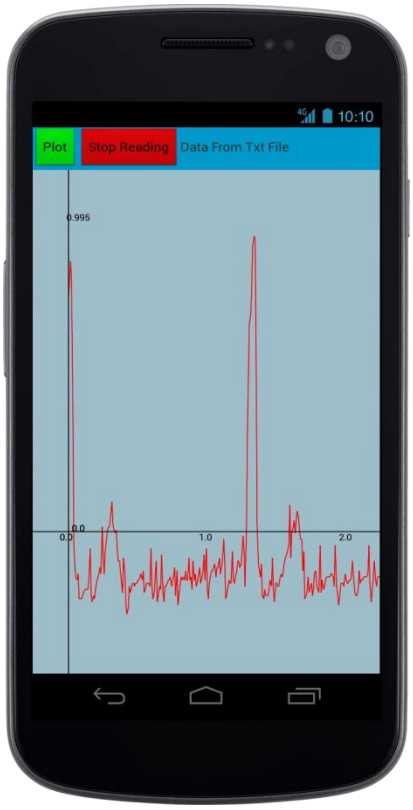
Το ecgsyn2 είναι ένα αρχείο αποθηκευμένο στον φάκελο resources της εφαμοργής, ο οποίος είναι κρυφός από τον χρήστη.

From .txt

Αυτός ο τρόπος λήψης δεδομένων είναι όμοιος με την επιλογή να αντλήσουμε τα δεδομένα από την βάση δεδομένων. Η διαφορά βρίσκεται στον τρόπο επιλογής του αρχείου. Το αρχείο επιλέγεται κατευθείαν από την sd κάρτα όπως φαίνεται στην παρακάτω οθόνη. Πατώντας το κουμπί "From txt" η εφαρμογή προτρέπει στον χρήστη να χρησιμοποιήσει περιηγητή αρχείων (file browser) της συσκευής του και να επιλέξει το αρχείο .txt που θέλει να απεικονίσει στην οθόνη.

Οθόνη λίστας αποθηκευμένων δεδομένων στον εσωτερικό χώρο του κινητού

Για παράδειγμα επιλέγοντας το αρχείο ecg-save-p2-2.txt το οποίο, όπως φαίνεται από την ονομασία είναι η δεύτερη καταγραφή του ασθενή 2 το σήμα απεικονίζεται παρακάτω.



Οθόνη απεικόνισης δεδομένων ECG

6 Επίλογος

6.1 Περίληψη

Στην παρούσα διπλωματική εργασία αναπτύχθηκε μια εφαρμογή για Smartphones που χρησιμοποιούν το λειτουργικό σύστημα android που απευθύνεται σε ιατρικό προσωπικό. Αυτή η εφαρμογή είχε σκοπό την απομακρυσμένη παρακολούθηση ενός ασθενή και συγκεκριμένα του ηλεκτροκαρδιογραφήματος του σε πραγματικό χρόνο.

Η ανάπτυξη της τεχνολογίας της επικοινωνίας επιτρέπει την μετάδοση κρίσιμων ζωτικών βιοσημάτων, όπως είναι το ηλεκτροκαρδιογράφημα, σε πραγματικό χρόνο. Η διάγνωση του γιατρού γίνεται πιο άμεσα και έχει σε διαρκή παρακολούθηση τον ασθενή του, ακόμα και αν αυτός βρίσκεται στο σπίτι. Επίσης διευκολύνεται η διαχείριση των επειγόντων περιστατικών και μειώνεται το κρίσιμο διάστημα μέχρι να φτάσει ο ασθενής στο νοσοκομείο, διότι έχουμε έγκαιρη ενημέρωση του γιατρού και του νοσοκομείου.

Τέλος, αξίζει να αναφερθεί ότι η αποθήκευση ιατρικών δεδομένων παρουσιάζει σημαντικό ενδιαφέρον. Δημιουργείται ένα προφίλ του ασθενή το οποίο αντανακλά επακριβώς την πορεία της ασθένειας και να δίνει πληροφορίες σχετικά αίτια της ασθένειας. Αυτό ονομάζεται ιατρικός φάκελος ασθενή και προσφέρει ευκολία στην πρόσβαση των δεδομένων σε αντίθεση με τον συμβατικό ιατρικό φάκελο.

6.2 Μελλοντικές επεκτάσεις

Ο σχεδιασμός της εφαρμογής αναπτύχθηκε στα πλαίσιο της μιας διπλωματικής εργασίας και δεν είχε εμπορικό σκοπό. Ο τομέας της βιοϊατρικής αναπτύσσεται ραγδαία και γι’ αυτό μπορούν να προστεθούν πολλές δυνατότητες. Με μικρές αλλαγές οι οποίες εξαρτώνται από το πρωτόκολλο επικοινωνίας των ιατρικών μηχανημάτων, η εφαρμογή θα μπορούσε να παρουσιάζει όλα τα βιοσήματα που λαμβάνονται από τον ασθενή.

1) Η επεξεργασία του σήματος και ο υπολογισμός του καρδιακού ρυθμού είναι από τις πιο βασικές λειτουργίες που μπορούν να προστεθούν. Αυτό παρέχει ευκολία στην εξαγωγή μιας γρήγορης διάγνωσης από τον γιατρό, ο οποίος σε περίπτωση κάποιου θα οδηγηθεί άμεσα στην αποστολή οδηγιών στον ασθενή ή στο κοντινό ιατρικό προσωπικό.
2) Επόμενο στάδιο της ανάλυσης του σήματος είναι η ανίχνευση διάφορων ανωμαλιών από την εφαρμογή ενημερώνοντας τον γιατρό αλλά και τον ασθενή καθώς και η κατηγοριοποίηση του σήματος σε συγκεκριμένες ομάδες μη φυσιολογικών ηλεκτροκαρδιογραφημάτων.
3) Ανάπτυξη αντίστοιχης εφαρμογής για τον ασθενή, η οποία δεν θα είναι απαραίτητα συνδεδεμένη με κάποιον server, αλλά θα ενημερώνει τον ασθενή για τα φυσιολογικά όρια και θα στέλνει αυτόματα ειδοποίηση στον γιατρό και στο νοσοκομείο αν υπάρχει κίνδυνος της υγείας του ασθενή.
4) Βελτίωση στην αποθήκευση της πληροφορίας. Λόγω του όγκου των δεδομένων αλλά και της ευκολίας πρόσβασης τους κρίνεται σημαντικό να αναπτυχθεί ένα κεντρικό σύστημα αποθήκευσης δεδομένων. Η πληροφορία θα πρέπει να αποθηκεύεται σε μία Διαδικτυακή βάση δεδομένων στην οποία θα έχουν πρόσβαση όλοι οι ιατροί.
5) Προσθήκη επιπλέον βιοσημάτων και άλλων πληροφοριών στην βάση δεδομένων για κάθε ασθενή με σκοπό να δημιουργηθεί ένας ιατρικός φάκελος.

7 Βιβλιογραφία

[1] Κουτσούρης, Δ., Παυλόπουλος, Σ., & Πρέντζα, Α. (2003). *Εισαγωγή Στη Βιοϊατρική Τεχνολογία και Ανάλυση Ιατρικών Σημάτων.* Αθήνα: Εκδόσεις Τζιόλα.

[2] Coeira, E. *Guide to Health Informatics*, 3rd edition

[3] http://en.wikipedia.org/wiki/Telemedicine

[4] https://en.wikipedia.org/wiki/EHealth

[5]https://ecoanemos.files.wordpress.com/2011/08/staggelidou_eisigisiimeridanaxos.pdf

[6] https://en.wikipedia.org/wiki/Android_(operating_system)

[7] http://www.tutorialspoint.com/android/android_architecture.htm

[8] http://www.c4learn.com/android/android-os-architecture/

[9]http://developer.android.com/guide/index.html

[10] https://www.sqlite.org/

[11]http://technet.microsoft.com/en-us/library/cc505839.aspx

[12] Gehani, N. (2006). The Database Book: Principles and practice using MySQL. 1st ed., Summit, NJ.: Silicon Press

[13] https://en.wikipedia.org/wiki/Network_socket

[14] http://www.tutorialspoint.com/java/java_networking.htm

[15] www-01.ibm.com/support/knowledgecenter/ssw_ibm_i_71/rzab6/howdosockets.htm

[16] http://pdplab.it.uom.gr/teaching/ince_2e_gr/Text/C2/NetworkprogramminginJava_4.htm

[17] https://el.wikiversity.org/wiki/Βάσεις Δεδομένων/Μοντέλο Οντοτήτων-Συσχετίσεων