ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Διπλωματική Εργασία της

Άννας Ρεπούση (Α.Μ: 2006030029)

«Σύστημα συσκευών για διαχείριση ηλεκτρονικής καρτέλας ασθενούς σε νοσοκομειακές μονάδες με τεχνολογία Android»

Επιτροπή:

Πολυχρόνης Κουτσάκης

Ευάγγελος Σακκόπουλος

Λαγουδάκης Μιχαήλ

Χανιά, Μάιος 2014

Ευχαριστίες

Η υλοποίση της παρούσας διπλωματικής εργασίας έγινε με την υποστήριξη ανθρώπων, οι οποίοι ανήκουν στο οικογενειακό, φιλικό και εκπαιδευτικό περιβάλλον μου.

Θα ήθελα αρχικά να ευχαριστήσω τους γονείς μου για την ηθική συμπαράσταση και ανοχή που έδειξαν απέναντι μου καθόλη τη διάρκεια της εργασίας μου.

Επιπλέον θα ήθελα να ευχαριστήσω για τη συναισθηματική τους υποστήριξη τους ανθώπους του στενού φιλικού μου κύκλου, με τους οποίους μοιράστηκα όλες τις σκέψεις μου (ευχάριστες και δυσάρεστες) όλο αυτό το διάστημα. Σε αυτούς ανήκουν και οι συμφοιτητές μου, τους οποίους ευχαριστώ για την ενθάρρυνσή τους αλλά και για όλες τις στιγμές που περάσαμε μαζί κατά τη διάρκεια των φοιτητικών μου χρόνων.

Τέλος θα ήθελα να ευχαριστήσω τους επιβλέποντες Κο Πολυχρόνη Κουτσάκη και Κο Ευάγγελο Σακκόπουλο για την επιβλεψή τους και για τις συστατικές επιστολές που έστειλαν σε πανεπιστήμια του εξωτερικού, όπως και τον Κο Μιχαήλ Λαγουδάκη, ο οποίος δέχτηκε να είναι μέλος της τριμελούς επιτροπής. Ενα μεγάλο ευχαριστώ, επίσης, στον Κο Αθανάσιο Τσακαλίδη που μου έδωσε την ευκαιρία να ασχοληθώ με τον συγκεκριμένο τομέα, προσφέροντάς μου το θέμα της παρούσας διπλωματικής εργασίας.

Πίνακας Περιεχομένων

ΠΕΡΙΛΗΨΗ 6
1. ΕΙΣΑΓΩΓΗ 8
1.1. Ιατρική και σύγχρονη τεχνολογία 8
1.2. Τεχνολογία Android 12
1.3. Αντικείμενο Διπλωματικής 16
1.4. Οργάνωση Κειμένου 17
2. ΛΕΙΤΟΥΡΓΙΚΕΣ ΑΠΑΙΤΗΣΕΙΣ 18
2.1 Βασική Αρχιτεκτονική Συστήματος 18
2.2 Λειτουργικές Απαιτήσεις – Διαγράμματα 20
2.2.1 Λειτουργικές Απαιτήσεις Γιατρού 20
2.2.2 Λειτουργικές Απαιτήσεις Νοσηλευτή 28
3. ΣΧΕΔΙΑΣΜΟΣ ΚΕΝΤΡΙΚΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ 33
3.1. Εργαλείο XAMPP 33
3.2. PhpMyAdmin 36
3.3. Βάση Δεδομένων 37
3.3.1 Πίνακες – Icd10 37
3.3.2 Σχέσεις Πινάκων 53
3.3.3 E-R 56
3.4. Μεταφορά βάσης 57
4. ΣΧΕΔΙΑΣΜΟΣ RESTFUL WEB SERVICES 59
4.1. Ορισμός 59
4.2. Php 61
4.2.1 Συγγραφή Αρχείων 61
4.2.2. ΚλήσηΑρχείωνμέσωκώδικα Java 74
4.3. Κωδικοποίηση JSON 76
5. ΤΟΠΙΚΗΒΑΣΗ SQLITE 78
5.1. Γενικές Πληροφορίες 78
5.2. Πίνακες 81
5.2.1. Δημιουργία πινάκων 81
5.2.2. Γέμισμα δεδομένων πινάκων 83

5.2.3. Ανάγνωση δεδομένων πινάκων 84
5.2.4. Διαγραφή δεδομένων πινάκων 86
5.3. Χρήση στην εφαρμογή 87
6. ΠΑΡΟΥΣΙΑΣΗ ΕΦΑΡΜΟΓΗΣ 92
6.1. Οθόνες Γιατρού 96
6.2. Οθόνες Νοσηλευτή 112
6.3. Αξιοσημείωτα αποσπάσματα κώδικα Java 126
7. ΕΠΙΛΟΓΟΣ 141
7.1. Συμπεράσματα 141
7.2. Μελλοντικές Βελτιώσεις 142
8. ΒΙΒΛΙΟΓΡΑΦΙΑ - ΑΝΑΦΟΡΕΣ 143

Πίνακας Εικόνων

Εικόνα 1: Medscape 9
Εικόνα 2: Epocrates 10
Εικόνα 3: Skyscape 10
Εικόνα 4: Evernote 11
Εικόνα 5: PubMedMobile 11
Εικόνα 6: Android 12
Εικόνα 7: Εκδόσεις Android 13
Εικόνα 8: Εφαρμογή ηλεκτρονικής καρτέλας ασθενούς σε νοσοκομειακές μονάδες 16
Εικόνα 9: Βασική Αρχιτεκτονική Συστήματος 18
Εικόνα 10: UseCaseDiagram για Γιατρό 22
Εικόνα 11: SequenceDiagram για ιστορικό ασθενή 24
Εικόνα 12: SequenceDiagram για προσθήκη διάγνωσης & φαρμάκου 25
Εικόνα 13: SequenceDiagram για ζωτικά σημεία 26
Εικόνα 14: SequenceDiagram για θερμομετρικό διάγραμμα 26
Εικόνα 15: SequenceDiagram για αποστολή παραπεμπτικών 27
Εικόνα 16: UseCaseDiagram για Νοσηλευτή 29
Εικόνα 17: SequenceDiagram για εισαγωγή ασθενή 30
Εικόνα 18: SequenceDiagram για καταγραφή ζωτικών σημείων 31
Εικόνα 19: SequenceDiagram για ανάγνωση φαρμακευτικής αγωγής και προσθήκης σχολίου. 32
Εικόνα 20: Xampp 33
Εικόνα 21: Εκκίνηση Xampp 34
Εικόνα 22: Xampp στον περιηγητή 35
Εικόνα 23: Κεντρική βάση hospital 36

Εικόνα 24: Πίνακας allergy 37
Εικόνα 25: Πίνακας allergyicd10 37
Εικόνα 26: Πίνακας clinic 38
Εικόνα 27: Πίνακας comment 38
Εικόνα 28: Πίνακας diagnosis 38
Εικόνα 29:Πίνακας diagnosisicd10 39
Εικόνα 30: Πίνακας doctor 39
Εικόνα 31: Πίνακας doctorclinic 39
Εικόνα 32: Πίνακας drug 40
Εικόνα 33: Πίνακας examination 40
Εικόνα 34: Πίνακας icd10 41
Εικόνα 35: Απόσπασμα Πίνακα excelIcd10 42
Εικόνα 36: Πίνακας lab 42
Εικόνα 37: Πίνακας Γενική Αίματος 43
Εικόνα 38: Πίνακας Ουρία 43
Εικόνα 39: Πίνακας Αμυλάση Ούρων 44
Εικόνα 40: Πίνακας Μαστογραφία 44
Εικόνα 41: Πίνακας medication 45
Εικόνα 42: Πίνακας medicationdrug 45
Εικόνα 43: Πίνακας nurse 45
Εικόνα 44: Πίνακας nurseclinic 45
Εικόνα 45: Πίνακας nursecomment 46
Εικόνα 46: Πίνακας patient 46
Εικόνα 47: Πίνακας patientallergy 47
Εικόνα 48: Πίνακας patientcomment 47
Εικόνα 49: Πίνακας patientdiagnosis 47
Εικόνα 50: Πίνακας patientmedication 47
Εικόνα 51: Πίνακας patientsurgery 48
Εικόνα 52: Πίνακας patientsymptom 48
Εικόνα 53: Πίνακας patientvitalsigns 48
Εικόνα 54: Πίνακας patientward 48
Εικόνα 55: Πίνακας previoushospitalization 49
Εικόνα 56: Πίνακας reference 49
Εικόνα 57: Πίνακας referenceexamination 50
Εικόνα 58: Πίνακας referencelab 50
Εικόνα 59: Πίνακας speciality 50
Εικόνα 60: Πίνακας surgery 50
Εικόνα 61: Πίνακας symptom 51
Εικόνα 62: Πίνακας symptomicd10 51
Εικόνα 63: Πίνακας vitalsigns 51
Εικόνα 64: Πίνακας ward 52
Εικόνα 65: Ορισμός Ευρετηρίου 53

Εικόνα 66: Υλοποίηση συσχετίσεων 1-N 54
Εικόνα 67: Υλοποίηση συσχετίσεων M-N 55
Εικόνα 68: E-R 56
Εικόνα 69: Export της βάσης 57
Εικόνα 70: Error στοimport 58
Εικόνα 71: Rest Web Service 60
Εικόνα 72: LoginDoctor.php 63
Εικόνα 73: AimatologikesExetaseis.php 64
Εικόνα 74: AvailiableDiagnosis.php 65
Εικόνα 75: ClinicsRelatedToDoctor.php 66
Εικόνα 76: GetCommentId.php 67
Εικόνα 77: LabResults1.php 68
Εικόνα 78: PatientAllergies.php 69
Εικόνα 79: PatientProf.php 70
Εικόνα 80: PatientReferences.php 71
Εικόνα 81: PatientsOfSpecificClinic.php 72
Εικόνα 82: InsertComment.php 73
Εικόνα 83: Αίτημα και απάντηση στον και από τον server 75
Εικόνα 84: Json parsing 78
Εικόνα 85: Μέθοδος OnCreate 82
Εικόνα 86: Συναρτήσεις add 83
Εικόνα 87: add_diagnosis 83
Εικόνα 88: Συναρτήσεις get 84
Εικόνα 89: getAllDiagnosis 85
Εικόνα 90: Συναρτήσεις delete 86
Εικόνα 91: deleteAllDiagnosis 86
Εικόνα 92: Λειτουργικότητα σε λειτουργία Γιατρού, σε περίπτωση επιτυχούς και αποτυχούς σύνδεσης 88
Εικόνα 93: Λειτουργικότητα σε περίπτωση αποτυχίας σύνδεσης σε λειτουργία Νοσηλευτή 89
Εικόνα 94: Κώδικας για refreshbutton 91
Εικόνα 95: Run 93
Εικόνα 96: WelcomePage 94
Εικόνα 97: Login Page 95
Εικόνα 98: Έγκυρο Login: Μετάβασηστο Clinics Page(α), Μηέγκυρο Login: Παραμονήστο Login Page(β) 96
Εικόνα 99: PatientsPage(α), Spinner(β), Αλλαγή κλινικής μέσω spinner(γ), Toast σε περίπτωση απουσίας ασθενών από κλινική(δ) 98
Εικόνα 100: DoctorTabs 99
Εικόνα 101: Προφίλ Ασθενούς 100
Εικόνα 102: Χειρουργικές Επεμβάσεις(α), Παροντικά Συμπτώματα(β) 100

Εικόνα 103: Διαγνώσεις - Αρχική σελίδα(α), Σελίδα Προσθήκης διάγνωσης(β), Επιστροφή στο tab Διαγνώσεων για επαλήθευση(γ) 102
Εικόνα 104: Φάρμακα - Αρχική σελίδα(α), Σελίδα Προσθήκης φαρμάκου(β), Σελίδα Προσθήκης φαρμάκου συνέχεια(γ), Επιστροφή στο tab Φαρμάκων για επαλήθευση(δ).. 103
Εικόνα 105: Progress Tab 105
Εικόνα 106: Ζωτικά Σημεία(α) - Ζωτικά Σημεία συνέχεια(β) - Γενική Αίματος(γ) - Γενική Αίματος συνέχεια(δ) - Ουρία(ε) - Μαστογραφία(στ) - Σχόλια(ζ) - Θερμ/κό Διάγραμμα με σφάλμα(η) - Θερμ/κό Διάγραμμα για 1 μέρα(θ) - Θερμ/κό Διάγραμμα για 2 μέρες(θ) - Θερμ/κό Διάγραμμα για 3 μέρες(θ) - Θερμ/κό Διάγραμμα για 4 μέρες(θ) - Θερμ/κό Διάγραμμα για 5 μέρες(θ) 109
Εικόνα 107: ReferenceTab(α), Επιλογή Εξέτασης(β), Ενημερωτικό toast στο πάτημα του φακέλου(γ), Επαλήθευση(δ) 111
Εικόνα 108: Μηέγκυρο Login: Παραμονήστο Login Page(α), Έγκυρο Login: Μετάβασηστησελίδαβασικώνεπιλογώννοσηλευτή(β), Basic Nurse Page(γ) 113
Εικόνα 109: Add new patient page(α), Add new patient page συνέχεια(β), Έλεγχοιεγκυρότητας(γ,δ,ε,στ,ζ), Πλήκτρογιαπληροφορίεςκλινικών(η), Προσθήκηνέουασθενή(θ) 116
Εικόνα 110: Clinics Page(α), Patients Page(β), Nurse Tabs(γ) 118
Εικόνα 111: VitalsignsTab(α), spinner για ώρα(β), Μηνύματα εγκυρότητας(γ,δ,ε,στ,ζ,η,θ,ι), Προσθήκη Μέτρησης(κ) 121
Εικόνα 112: MedicineTab 122
Εικόνα 113: CommentTab(α), Προσθήκη Σχολίου(β) 123
Εικόνα 114: LogOut(α), Συγχρονισμός(β), Επαλήθευση(γ,δ,ε) 125
Εικόνα 115: PatientsOfClinic Activity 131
Εικόνα 116: CustomList2 132
Εικόνα 117: activity_patients_of_clinic.xml 133
Εικόνα 118: History Tab Activity 136
Εικόνα 119: activity_history_tab.xml 137
Εικόνα 120: ThermometricalDiagram5 140

ΠΕΡΙΛΗΨΗ

Σε αυτήν τη διπλωματική εργασία σχεδιάσαμε και υλοποιήσαμε μια εφαρμογή για έξυπνα κινητά τηλέφωνα (smartphones) και tablets με τεχνολογία Android, η οποία απευθύνεται στο προσωπικό των νοσοκομειακών μονάδων (γιατρούς και νοσηλευτές), διευκολύνοντας την εργασία τους.

Πιο συγκεκριμένα, οι γιατροί έχουν τη δυνατότητα να δουν το ιατρικό προφίλ των ασθενών (προσωπικά στοιχεία, χειρουργικές επεμβάσεις, αλλεργίες, παροντικά συμπτώματα), να προσθέσουν κάποια διάγνωση από ήδη υπάρχουσες στην κεντρική βάση – server, να κάνουν ηλεκτρονική συνταγογράφηση, να μελετήσουν τα ζωτικά σημεία του εκάστοτε ασθενούς καθώς και τα αποτελέσματα των εξετάσεων που έχει κάνει, να δουν κάποιο πιθανό σχόλιο νοσηλευτών, να εξετάσουν γραφικά το θερμομετρικό του διάγραμμα και να στείλουν παραπεμπτικά για εξετάσεις στα εργαστήρια.

Από την άλλη, οι νοσηλευτές έχουν τη δυνατότητα να μελετήσουν το προφίλ των ασθενών (προσωπικά στοιχεία, χειρουργικές επεμβάσεις, αλλεργίες, τρέχοντα συμπτώματα), να έχουν πρόσβαση στη φαρμακευτική αγωγή και τη διάγνωση που έχει καθοριστεί από το γιατρό, να εισάγουν τιμές για ζωτικά σημεία και να γράψουν κάποιο πιθανό σχόλιο κατά την προσέλευσή τους στο θάλαμο του ασθενή που θα ήταν χρήσιμο προς το γιατρό. Επιπλέον έχουν τη δυνατότητα να εισάγουν τα προσωπικά στοιχεία κάποιου νέου ασθενή.

Μέσω της συγκεκριμένης εφαρμογής μπορεί να μειωθεί σημαντικά ο όγκος γραφικής ύλης (καρτέλες ασθενών, συνταγογραφήσεις) και μειώνονται αρκετά τα ανθρώπινα λάθη κατά την εισαγωγή στοιχείων χάρις στους ελέγχους εγκυρότητας που λαμβάνουν χώρα. Επίσης επιτυγχάνεται εξοικονόμηση χρόνου, καθώς δεν απαιτείται πλέον η διαδικασία εύρεσης φακέλων – εξετάσεων ασθενή. Ακόμα, η διαδικασία αποστολής παραπεμπτικών στα εργαστήρια είναι πλέον αυτοματοποιημένη. Κατ' αυτόν τον τρόπο όταν τα αποτελέσματα των εξετάσεων είναι έτοιμα το σύστημα ενημερώνεται αυτόματα χωρίς να απαιτούνται επιπλέον ενέργειες από το νοσοκομειακό προσωπικό.

Τα εργαλεία που χρησιμοποιήθηκαν για την εφαρμογή είναι το Agilian 10.2-Visualparadigm για την κατασκευή των διαγραμμάτων sequence, usecase και e-r, το Visio 2013 για την κατασκευή του διαγράμματος της βασικής αρχιτεκτονικής, το Xampp 1.8.3&phpMyAdminγια τη μετατροπή του υπολογιστή σε server και τη δημιουργία της κεντρικής βάσης του συστήματος, το AndroidSdk, AdtPlugin&Eclipse για την υλοποίηση κώδικα σε javaκαι την παρουσίαση του emulatorκαι το SqlLite

για τοπική αποθήκευση δεδομένων στη συσκευή του κινητού. Τέλος, για την υλοποίηση των αρχείων php που χρειάστηκαν για την σύνδεση της κινητής συσκευής με τον serverχρησιμοποιήθηκε το εργαλείο Notepad++.

1. ΕΙΣΑΓΩΓΗ

1.1. Ιατρική και σύγχρονη τεχνολογία

Τις τελευταίες δεκαετίες είμαστε μάρτυρες της ραγδαίας εισχώρησης των σύγχρονων τεχνολογικών επιτευγμάτων Πληροφορικής και Τηλεπικοινωνιών σε πολλούς κλάδους της καθημερινότητας προς απλούστευσή της. Ένας από τους σημαντικότερους αυτούς κλάδους είναι και αυτός της Ιατρικής.

Χαρακτηριστικό παράδειγμα αποτελεί η **τηλεϊατρική**, η οποία αναφέρεται στην εφαρμογή των σύγχρονων τεχνολογιών, των τηλεπικοινωνιών και της πληροφορικής, για να προσφέρει σε ασθενείς κλινική βοήθεια από απόσταση. Η τηλεϊατρική βοηθάει περισσότερο εκείνους που βρίσκονται σε απομακρυσμένες περιοχές, όπως οι αγρότες ή όταν ο θεράπων ιατρός βρίσκεται σε άλλη περιοχή. Η χρήση των νέων τεχνολογιών επιτρέπει την εύκολη επικοινωνία του ιατρού με τον ασθενή μέσω της μετάδοσης ήχου και εικόνας.

Ένα ακόμη σημαντικό επίτευγμα είναι ο **ηλεκτρονικός φάκελος ασθενή**, που εν γένει είναι ένας ψηφιακά αποθηκευμένος φάκελος με σκοπό να υποστηριχτεί η φροντίδα υγείας του ατόμου εφ' όρου ζωής, προωθεί την έρευνα και την εκπαίδευση των επαγγελματιών υγείας και βοηθά στην πρόσβαση και στο διαμοιρασμό πληροφοριών στους επαγγελματίες υγείας με φιλικό τρόπο καθώς ελέγχεται και η ασφάλεια των δεδομένων.

Θα μπορούσαμε επίσης να αναφερθούμε στην **τηλεπαρακολούθηση** των ασθενών μέσω αισθητήρων. Με την πάροδο των χρόνων έχει παρατηρηθεί μια μεταστροφή από την εξυγίανση των αρρώστων στο να παραμένουν οι πολίτες υγιείς και να παρακολουθούνται μη επεμβατικά με όσο το δυνατόν μικρότερη ενόχληση σε χώρους που θεωρούν οικείους. Παρατηρείται με άλλα λόγια μια στροφή στην άσκηση προληπτικής ιατρικής και αποασυλοποιήσης των ασθενών και την παρακολούθηση από το σπίτι με αποτέλεσμα τη μείωση των κατ'οίκων επισκέψεων και των επισκέψεων στο νοσοκομείο. Αυτό μπορεί να επτευχθεί με τη χρήση ενός ασύρματου δικτύου και ενός απλού αισθητήρα διευκολύνοντας δραστικά τον ασθενή , ιδιαιτέρως σε περιπτώσεις που είναι ηλικιωμένος ή ανήκει σε ευπαθείς ομάδες ή διανύει κάποια μετεγχειρητική περίοδο.

Κάποιες από τις ήδη υπάρχουσες σύγχρονες εφαρμογές για smartphoneskαι tablets που απευθύνονται σε γιατρούς και άλλους επαγγελματίες υγείας είναι το **Medscape**, το **Epocrates**, το **Skyscape**, το**Evernote**και το**PubMedMobile**.(gkatzios, 2013)

Εικόνα 1: Medscape

Ο ιστότοπος **Medscape Education** είναι η απόλυτη διαδικτυακή πηγή πληροφοριών και εκπαίδευσης σε ιατρικά θέματα.Περιέχει μεταξύ άλλων ειδήσεις, άρθρα, επιστημονικές εργασίες, βάσεις δεδομένων για ασθένειες, ανατομία, φάρμακα, θεραπευτικές μεθόδους, οπτικοακουστικό υλικό, εκπαιδευτικό υλικό, κυρίως για επιστήμονες του χώρου (με δυνατότητα κατεβάσματος από το χρήστη). Πολύ χρήσιμη είναι η δυνατότητα αναζήτησης παρενεργειών από συνδυασμό φαρμάκων. Ο χρήστης επιλέγει όποιο συνδυασμό φαρμάκων τον ενδιαφέρει και λαμβάνει ως απάντηση τις παρενέργειες (εφόσον αυτές υπάρχουν).ΤοMedscape είναι μια μεγάλη App για το φάρμακο, αλλά πολλοί δεν συνειδητοποιούν ότι είναι επίσης ένα μίνι βιβλίο γεμάτο με τα πρωτόκολλα για τις παθολογίες της νόσου,ιδανικό για μια γρήγορη αναφορά και φρεσκάρισμα των ιατρικών τους γνώσεων.

Εικόνα 2: Epocrates

Το **Epocrates**είναι αναμφισβήτητα το πιο ευρέως χρησιμοποιούμενο ιατρικό app. Στην ουσία είναι μια κινητή πηγή αναφοράς των φαρμάκων που παρέχει κλινικά στοιχεία σχετικά με χιλιάδες συνταγές και OTC φαρμακευτικών προιόντων. Διαθέτει σε βάθος πληροφορίες, συνταγολόγιο, πληροφορίες αλληλεπίδρασης και δεκάδες υπολογισμούς (π.χ. ΔΜΣ, GFR). Η εφαρμογή έχει λάβει πάνω από 43.000 βαθμολογίες για όλες τις εκδόσεις και έχει κατά μέσο όρο 3 αστέρια.

Εικόνα 3: Skyscape

Η εφαρμογή **Skyscape** είναι μια από τις πιο διαδεδομένες εφαρμογές ανάμεσα στους επαγγελματίες υγείας που διαθέτουν έξυπνα τηλέφωνα με λειτουργικό Android. Κάποιες από τις παροχές της είναι ευρετήριο φαρμακευτικών ουσιών και σκευασμάτων, ευρετήριο φυσιολογικών και παθολογικών οντοτήτων της ιατρικής με σύντομη επισήμανση των κυριότερων σημείων, υπολογιστής πολλών χρήσιμων ιατρικών δεικτών και σημαντικά νέα από την παγκόσμια ιατρική κοινότητα.

Εικόνα 4: Evernote

Το**Evernote**είναι ουσιαστικά ένα εξελιγμένο σημειωματάριο που μπορεί να συγχρονιστεί σε οποιαδήποτε συσκευή για markupεικόνων, εγγράφων και σημειώσεων. Προσφέρει επίσης τη δυνατότητα για δημιουργία to-doListκαθώς και προβολή RelatedNotesγια ειδικές κατηγορίες χρηστών.

Εικόνα 5: PubMedMobile

Η εφαρμογή **PubMedMobile**είναι μια γρήγορη και εύκολη πρόσβαση στο PubMed από μια φορητή συσκευή. Οι χρήστες μπορούν να αναζητήσουν και να δουν αποσπάσματα δημοσιεύσεων. Επίσης μετά την τοποθέτηση ετικετών στα άρθρα που παρουσιάζουν ενδιαφέρον, μπορούν να στείλουν το άρθρο μέσω e-mail ή να το μοιραστούν μέσω των κοινωνικώνδικτύων.

1.2. Τεχνολογία Android

Εικόνα 6: Android

Το **Android**(Meier, 2010) (developer.android)(wikipedia)είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την OpenHandsetAlliance.Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από τηνGoogle.

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού OpenHandsetAlliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της ApacheLicense, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια IrinaBlok.

Παρόλο που το Android είναι ένα προϊόν ελεύθερου λογισμικού, ένα κομμάτι της ανάπτυξης του λογισμικού συνεχίζεται σε ιδιωτικό παρακλάδι. Για να έρθει αυτό το λογισμικό σε κοινή θέαση δημιουργήθηκε ένα παρακλάδι του μόνο ανάγνωσης, εν ονόματι "Cupcake". ToCupcake συνήθως συγχέεται με τον τίτλο μιας ενημέρωσης, σε αντίθεση με όσα δηλώνει η ίδια η Google στην ιστοσελίδα ανάπτυξης του Android: "το Cupcake αποτελεί ακόμη ένα έργο σε εξέλιξη, όχι μια επίσημη έκδοση."Αξιοσημείωτες αλλαγές στο λειτουργικό Android θα παρουσιαστούν στο cupcake και περιλαμβάνουν αλλαγές στο σύστημα διαχείρισης των μεταφορτώσεων (Downloadmanager), το Framework, Bluetooth, το λογισμικό

συστήματος, το ραδιόφωνο και το σύστημα τηλεφωνίας, εργαλεία προγραμματισμού, το κυρίως σύστημα και διάφορες εφαρμογές, καθώς και πληθώρες διορθώσεις σφαλμάτων.Στις 30 Απριλίου 2009, κυκλοφόρησε η επίσημη ενημέρωση έκδοσης 1.5 για το Android.Αποτελείται από πολλά νέα χαρακτηριστικά και βελτιώσεις στο γραφικό περιβάλλον:

- Δυνατότητα καταγραφής κινούμενης εικόνας με την χρήση της αντίστοιχης λειτουργίας του τηλεφώνου.
- Μεταφόρτωση αρχείων βίντεο στο YouTube και εικόνων στο Picasa κατευθείαν από το τηλέφωνο.
- Επανασχεδιασμένο λογισμικό πληκτρολογίου με λειτουργία αυτόματης συμπλήρωσης κειμένου.
- Δυνατότητα αυτόματης σύνδεσης ασύρματης συσκευής ακουστικού Bluetooth εφόσον εντοπιστεί σε μια συγκεκριμένη απόσταση.
- Νέα widgets και φάκελοι που μπορούν να τοποθετηθούν στην επιφάνεια εργασίας.
- Εφέ αλλαγής οθονών και μενού.
- Διευρυμένη λειτουργία αντιγραφής/επικόλλησης για να περιλαμβάνει δικτυακές διευθύνσεις.

Οι διαθέσιμες **εκδόσεις**Androidπαρουσιάζονται στον παρακάτω πίνακα:

Έκδοση	Κωδική ονομασία	Ημερομηνία	API level	Διανομή
4.4	*Kitkat*	31 Οκτωβρίου 2013	19	0%
4.3	*Jelly Bean*	24 Ιουλίου 2013	18	2,3%
4.2.x	*Jelly Bean*	13 Νοεμβρίου 2012	17	12,5%
4.1.x	*Jelly Bean*	9 Ιουλίου 2012	16	37,3%
4.0.x	*Ice Cream Sandwich*	16 Δεκεμβρίου 2011	15	19,8%
3.2	*Honeycomb*	15 Ιουλίου 2011	13	0.1%
2.3-2.3.7	*Gingerbread*	9 Φεβρουαρίου 2011	10	26,3%
2.2	*Froyo*	20 Μαΐου 2010	8	1,2%
2.0-2.1	*Eclair*	26 Οκτωβρίου 2009	7	?
1.6	*Donut*	15 Σεπτεμβρίου 2009	4	?

Εικόνα 7: Εκδόσεις Android

Ορισμένα από τα **ΠΛΕΟΝΕΚΤΗΜΑΤΑ**του Androidλειτουργικού έναντι άλλων ανταγωνιστικών είναι τα εξής:

- **Εναλλακτικά Πληκτρολόγια**

  Το να πληκτρολογεί κάποιος σε ένα μικροσκοπικό πληκτρολόγιο κινητού δεν αποτελεί διασκέδαση, οπότε είναι σπουδαίο που το Android προσφέρει πολλές επιλογές για να το κάνει όσο πιο ανώδυνο γίνεται και πολύ εύκολο στην εγκατάσταση. Άλλες τεχνολογίες παρέχουν άλλα πληκτρολόγια, αλλά είναι συνήθως ξεχωριστές εφαρμογές που απαιτούν την εισαγωγή κειμένου σε ένα άλλο πρόγραμμα.

- **Αυτοματοποίηση**

  Μια από τις πιο ισχυρές και χρήσιμες εφαρμογές στο Android είναι το Tasker, το πρόγραμμα αυτοματοποίησης που επιτρέπει στο χρήστη να μετατρέψει το κινητό του σε ένα σούπερ κινητό. Μπορεί να ενεργοποιήσει και απενεργοποιήσει ρυθμίσεις για συγκεκριμένες εφαρμογές, ανάλογα την τοποθεσία, την ώρα και σχεδόν οποιαδήποτε άλλη κατάσταση μπορεί να σκεφτεί. Με τις σωστές εντολές, το Tasker μπορεί να έχει πρόσβαση ακόμα και στις πιο βαθιές και σκοτεινές ρυθμίσεις του κινητού, κάτι το οποίο δεν μπορούν να το κάνουν τα άλλα λειτουργικά.

- **Widgets**

  Σίγουρα, καταλαμβάνουν κάποιον χώρο, αλλά δεν υπάρχει υποκατάστατο για την ευκολία του να έχειοχρήστης ένα μεγάλο widget με τον καιρό στην αρχική οθόνη ή ένα widget μουσικής που να του δείχνει το κομμάτι που παίζει εκείνη την στιγμή. Ακόμα πιο χρήσιμο είναι τα to-do-Listwidgets, που βοηθάνε περισσότερο στην παραγωγικότητα.

- **Αφαιρούμενα Μέσα Αποθήκευσης και Μπαταρία**

  Δεν είναι μέρος του Android λογισμικού απαραίτητα, αλλά η ανοιχτού λογισμικού φύση του δίνει αρκετά πλεονεκτήματα όπως την εξαγωγή και αναβάθμισητης μπαταρίας και της κάρτας μνήμης. Αν ο χρήστης διαπιστώσει ότι έχει φτάσει στο όριο του χώρου αποθήκευσης σε άλλα λειτουργικά δεν μπορεί να κάνει και πολλά εκτός από το να αρχίσει να σβήνει πράγματα, ενώ στο Android μπορεί απλά να βάλει μια μεγαλύτερη κάρτα μνήμης. Παρόμοια μπορεί να αλλάξει την μπαταρία με μια

μεγαλύτερης διάρκειας που θα κάνει το κινητό να κρατήσει ακόμα περισσότερο χωρίς να χρειαστεί φόρτιση.

- **Ασύρματη Εγκατάσταση Εφαρμογών**

  Η αναζήτηση και ανακάλυψη νέων εφαρμογών πρέπει να είναι διασκεδαστική διαδικασία για τον εκάστοτε χρήστη, όχι να αποτελεί πρόκληση να τα καταφέρει μέσω ενός μικροσκοπικού AppStore. Το AppStore και το CydiaAppStore δεν είναι και τόσο διασκεδαστικά να περιηγείται σε αυτά μέσω του iPhone, αλλά είτε θα πρέπει να κατεβάσει τις εφαρμογές στο κινητό του είτε να το συνδέσει με το iTunes για να τις συγχρονίσει. Με το AndroidMarket αντιθέτως μπορεί να βρει μια εφαρμογή, να πατήσει το κουμπί της εγκατάστασης και θα έχει εγκατασταθεί στο κινητό του την επόμενη φορά που θα το χρησιμοποιήσει.

- **Προσαρμοσμένες ROM**

  Αν και υπάρχουν πολλές εφαρμογές τρίτων στο Android που δίνουν προηγμένα χαρακτηριστικά, ένα από τα πιο ενδιαφέροντα πράγματα σχετικά με το λειτουργικό αυτό και το ότι είναι ανοιχτού κώδικα, είναι το γεγονός ότι οι άνθρωποι μπορούν να το πάρουν, να το «πειράξουν» και να εγκαταστήσουν την δικιά τους έκδοση αντί για αυτή που είχε το κινητό εξαρχής.

- **Έλεγχος του τηλεφώνου από τον Υπολογιστή**

  Υπάρχουν αρκετές εφαρμογές που επιτρέπουν στον χρήστη να ελέγξει το Android κινητό του από τον υπολογιστή του, είτε απλά θέλει να στείλει κάποιο μήνυμα κειμένου από τον Chrome, είτε να έχει πρόσβαση σε άλλες λειτουργίες απευθείας από έναν webbrowser.

- **FlashPlayer**

  Το flashplayer βρίσκεται οπουδήποτε και όταν ο χρήστης αναγκάζεται να σερφάρει στο internet χωρίς αυτό, τότε καταλαβαίνει πόσο πολύ βασίζεται σε αυτό μέρα με την μέρα. Είτε πρόκειται για μια σελίδα εξ' ολοκλήρου φτιαγμένη σε Flash, είτε για βίντεο, είτε για παιχνίδια, το να έχει το Flash εγκατεστημένο στο κινητό του ή το Tablet του δίνει πρόσβαση σε πολλά πράγματα που διαφορετικά δεν θα μπορούσε.

1.3. Αντικείμενο Διπλωματικής

Εικόνα 8: Εφαρμογή ηλεκτρονικής καρτέλας ασθενούς σε νοσοκομειακές μονάδες

Το αντικείμενο αυτής της διπλωματικής εργασίας είναι όπως προαναφέρθηκε η δημιουργία μιας εφαρμογής για έξυπνα κινητά τηλέφωνα (smartphones)και tabletsμε τεχνολογία Android, η οποία απευθύνεται στο προσωπικό των νοσοκομειακών μονάδων (γιατρούς και νοσηλευτές), διευκολύνοντας την εργασία τους και αφορά σε γενικές γραμμές τη διαχείρηση ηλεκτρονικής καρτέλας ασθενή.

Μετά την εισαγωγή και ταυτοποίηση εγκυρότητας των στοιχείων του εκάστοτε χρήστη στην εφαρμογή, θα του δίνεται η δυνατότητα να εκτελέσει μια πληθώρα λειτουργιών. Το ποιες θα είναι αυτές εξαρτάται από την ιδιότητά του γιατρός ή νοσηλευτής και έχει γίνει συνοπτική αναφορά τους στην περίληψη της διπλωματικής.

Θα πρέπει να σημειωθεί επίσης πως το σύστημα αποτελείται ουσιαστικά από 2 βάσεις δεδομένων, μία τοπική και μία κεντρική(server). Η ανάγκη δημιουργίας της τοπικής βάσης προέκυψε από το γεγονός πως δεν είναι πάντοτε ευφικτή η σύνδεση με τον server, επομένως θα πρέπει να αποθηκεύονται κάπου τα δεδομένα που δίνει ο χρήστης, ώστε όποτε η σύνδεση είναι επιτυχής να μεταφέρονται στον server, δίχως απώλειες. Ο συγχρονισμός των 2 αυτών βάσεων γίνεται με το πάτημα ενός κουμπιού «refresh»από το χρήστη.

1.4. Οργάνωση Κειμένου

Η δομή της εργασίας παρουσιάζεται παρακάτω:

ΚΕΦΑΛΑΙΟ 1: Εισαγωγικά Στοιχεία, Περιληπτική παρουσίαση κάποιων εφαρμογών που σχετίζονται με την υγεία, Περιγραφή και πλεονεκτήματα του λειτουργικού συστήματος Android, Αντικείμενο διπλωματικής εργασίας, Κεφάλαια και miniπεριγραφή τους

ΚΕΦΑΛΑΙΟ 2:Λειτουργικές απαιτήσεις,Παρουσίαση βασικής αρχιτεκτονικής συστήματος, ΔιαγράμματαUMLusecases, sequence

ΚΕΦΑΛΑΙΟ 3:Εργαλείο Xampp, Σχεδίαση κεντρικής βάσης δεδομένων(phpMyAdmin), Τρόπος δημιουργίας πινάκων και σχέσεων μεταξύ τους, e-rschema, Τρόπος μεταφοράς υπάρχουσας βάσης σε άλλο pc

ΚΕΦΑΛΑΙΟ 4:RESTfulWebServices, Αρχεία php, Σύνδεση με server, Χρήση εργαλείου Json για αποθήκευση δεδομένων και μελλοντική τους χρήση στον κώδικα

ΚΕΦΑΛΑΙΟ 5:SQLite τοπική βάση, Δημιουργία, Γέμισμα, Ανάγνωση, Διαγραφή, Χρήση πινάκων

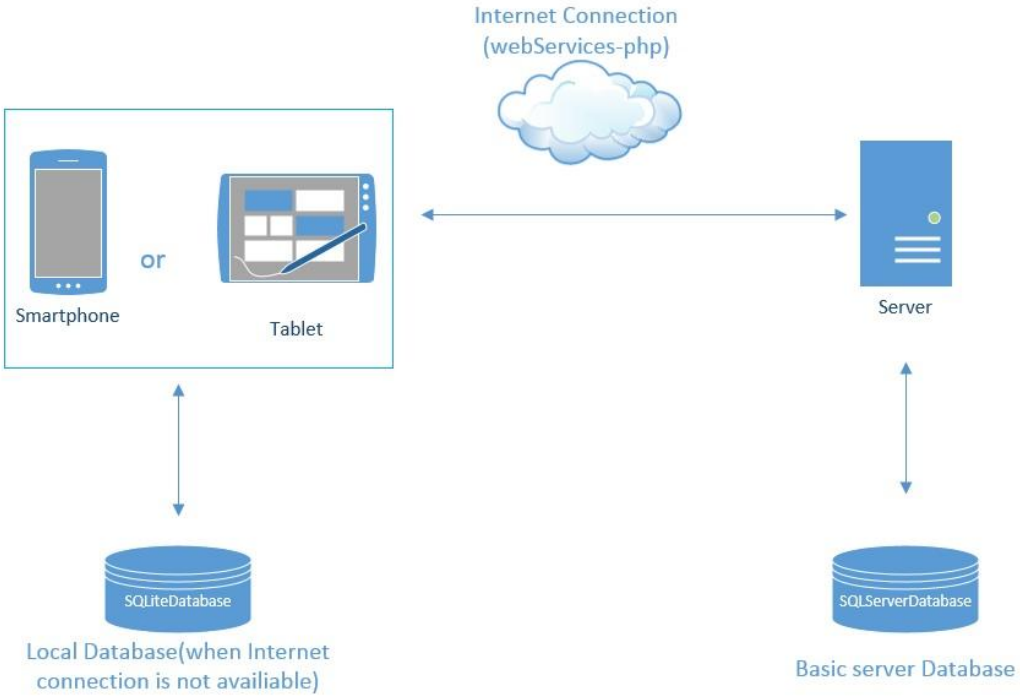
ΚΕΦΑΛΑΙΟ 6:Λεπτομερειακή παρουσίαση βήμα προς βήμα της εφαρμογής, Ενδεικτικά σημεία κώδικα java που χρήζουν παρατήρησης – επεξήγησης

ΚΕΦΑΛΑΙΟ 7:Συμπεράσματα, Βελτιώσεις, Μελλοντικές Επεκτάσεις

ΚΕΦΑΛΑΙΟ 8:Βιβλιογραφία και linksπου χρησιμοποιήθηκαν κατά την υλοποίηση

2. ΛΕΙΤΟΥΡΓΙΚΕΣ ΑΠΑΙΤΗΣΕΙΣ

2.1 Βασική Αρχιτεκτονική Συστήματος



Εικόνα 9: Βασική Αρχιτεκτονική Συστήματος

Η βασική αρχιτεκτονική του συστήματος παρουσιάζεται γραφικά στην παραπάνω εικόνα. Στο σχήμα παρατηρούμε την ύπαρξη 2 βάσεων δεδομένων, μία που χρησιμοποιείται σε περίπτωση αποτυχίας σύνδεσης στο serverκαι άλλη μία, η οποία είναι ουσιαστικά η κεντρική βάση και περιλαμβάνει όλα τα απαραίτητα δεδομένα για την ορθή λειτουργία της εφαρμογής. Η ιδέα κατ' ουσίαν είναι πως αρχικά επιχειρείται σύνδεση στον serverμέσω webservices, όπως παρουσιάζεται και στο παραπάνω σχήμα. Πιο συγκεκριμένα μέσω αρχείων .phpεπικοινωνούμε με την κεντρική βάση από την οποία είτε παίρνουμε, είτε δίνουμε στοιχεία . Σε περίπτωση που δεν είναι διαθέσιμη σύνδεση στο Internet, η εφαρμογή εξακολουθεί να λειτουργεί και χειρίζεται δεδομένα μέσω της τοπικής βάσης. Για παράδειγμα, ένας νοσηλευτής αν θελήσει να εισάγει έναν νέο ασθενή, πληκτρολογεί τα στοιχεία του και αυτά αποθηκεύονται τοπικά, ώστε να μην χαθούν οι εγγραφές. Όταν ο χρήστης επιθυμεί, υλοποιεί συγχρονισμό των 2 βάσεων πατώντας ένα συγκεκριμένο κουμπί

«refresh»και ό,τι δεδομένα έχουν αποθηκευτεί στην τοπική βάση, με προϋπόθεση ύπαρξης επιτυχημένης σύνδεσης στο Διαδίκτυο, μεταφέρονται στην κεντρική. Κατόπιν, μετά την επιτυχημένη μεταφορά των δεδομένων, υλοποιείται διαγραφή τους από την τοπική βάση, με σκοπό να «ελαφρύνει» η μνήμη της συσκευής. Περισσότερες λεπτομέρειες για τη συγκεκριμένη λειτουργικότητά παρατίθενται στο κεφάλαιο 5.

Αξίζει να σημειωθεί πως οι λόγοι για τους οποίους επιλέξαμε τη συγκεκριμένη αρχιτεκτονική, η οποία περιλαμβάνει και την τοπική βάση SQLite, είναι οι εξής:

- Τα στοιχεία που αποθηκεύονται τοπικά είναι διαθέσιμα αποκλειστικά και μόνο στην εφαρμογή μας, πράγμα που διασφαλίζει το απόρρητο των δεδομένων.
- Αποφυγή του να «σκάει» η εφαρμογή σε περίπτωση αποτυχίας σύνδεσης στο διαδίκτυο, γεγονός που εξασφαλίζει χρηστικότητα σε ρεαλιστικές συνθήκες. Ιδιαίτερα στον τομέα της υγείας, θα πρέπει να λαμβάνονται όλα τα απαραίτητα μέτρα για την ορθή και άμεση εξυπηρέτηση του χρήστη.
- Η SQLite έχει μικρό μέγεθος και αρκετά μεγάλη ευελιξία

2.2 Λειτουργικές Απαιτήσεις – Διαγράμματα

2.2.1 Λειτουργικές Απαιτήσεις Γιατρού

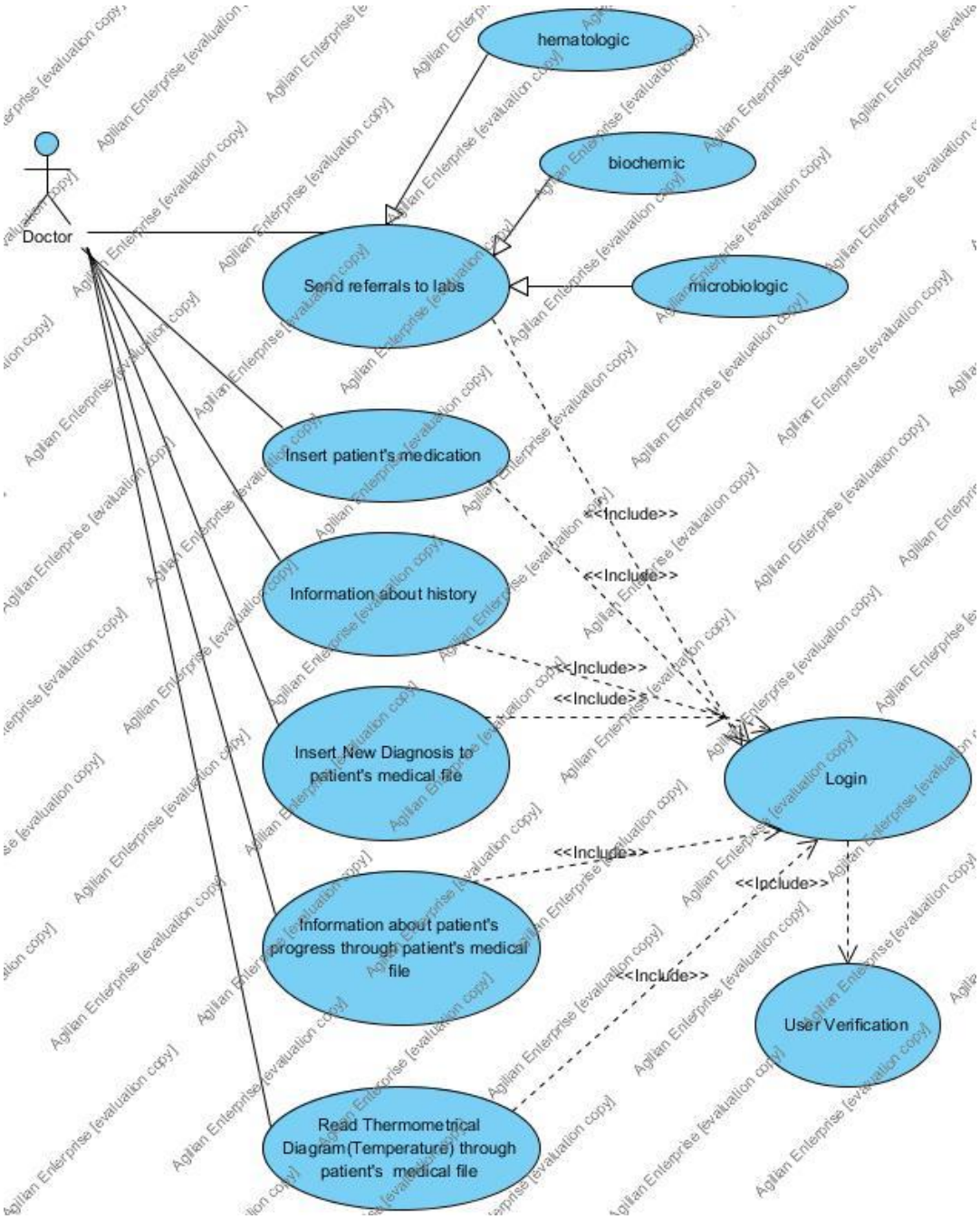
Όπως έχει προαναφερθεί η συγκεκριμένη εφαρμογή δεν απευθύνεται σε ασθενείς αλλά αποκλειστικά σε νοσηλευτές και γιατρούς των νοσοκομειακών μονάδων.

Πιο συγκεκριμένα οι λειτουργικές απαιτήσεις των γιατρών είναι οι εξής:

1. Πληροφόρηση τους για το προσωπικό και ιατρικό ιστορικό των ασθενών. Ο γιατρός έχει τη δυνατότητα να δει τα προσωπικά στοιχεία του επιλεγμένου ασθενή (όνομα, επίθετο, ημ/νία γέννησης, αμκα, αφμ, ημ/νία εισαγωγής-εξαγωγής από το νοσοκομείο κτλ.), τυχόν αλλεργίες που παρουσιάζει, τυχόν χειρουργικές επεμβάσεις στις οποίες έχει υποβληθεί καθώς και τα παροντικά του συμπτώματα, ώστε να έχει μια πλήρη προσωπική και κλινική του εικόνα.
2. Πληροφόρηση για τυχόν διαγνώσεις που έχουν γίνει ήδη στον ασθενή για όσο διάστημα βρίσκεται στο νοσοκομείο και δυνατότητα εισαγωγής νέας διαγνωσης. Προς διευκόλυνση του χρήστη φορτώνονται στην εφαρμογή κάποιες ασθένειες και εκείνος επιλέγει με «τικ» όποια-ες επιθυμεί ως διάγνωση-εις για τον επιλεγμένο ασθενή.
3. Ενημέρωση για τη φαρμακευτική αγωγή που λαμβάνει ο χρήστης για όσο διάστημα νοσηλεύεται(ονομασία φαρμάκου, δοσολογία, ημ/νία εκκίνησης και διακοπής της αγωγής). Του δίνεται, συν τοις άλλοις, η δυνατότητα να επιλέξει κάποιο-α προς προσθήκη φάρμακα δηλώνοντας την ημ/νία εκκίνησης και διακοπής αγωγής καθώς και τη δοσολογία, λειτουργία που προσομοιώνει κατά κάποιο τρόπο την ηλεκτρονική συνταγογράφηση.
4. Ενημέρωση για την κλινική εικόνα του ασθενή μέσω των μετρήσεων των ζωτικών σημείων(πίεση, αναπνοές, σφυγμόι, SpO2, ΚΦΠ, κεννώσεις, σάκχαρο), οι οποίες υποθέσαμε οτι λαμβάνονται 4 φορές ανά 24 ωρο. Υποστηρίζεται, επίσης, η ανάγνωση όλων των αποτελεσμάτων των εξετασεων στις οποίες έχει υποβληθεί ο χρήστης σε όλο το διάστημα νοσηλείας του. Ο γιατρός έχει τη δυνατότητα να αναγνώσει τυχόν σχόλια νοσηλευτών συνοδευόμενα από την περιγραφή τους, ημ/νία και ώρα που έγιναν. Τέλος, του παρέχεται η δυνατότητα ανάγνωσης του θερμομετρικού διαγράμματος του ασθενή για maximum5 μέρες πριν την τωρινή ημερομηνία, ώστε να έχει μια όσο το δυνατόν πληρέστερη εικόνα για την πρόοδο της θερμοκρασίας του.

5. Αποστολή παραπεμπτικών σε εργαστήρια για να λάβουν χώρα κάποιες εξετάσεις. Η εφαρμογή υποστηρίζει ενδεικτικά 4 εργαστήρια(αιματολογικό, μικροβιολογικό, βιοχημικό και ακτινολογικό). Κάθε εργαστήριο έχει τη δυνατότητα να υλοποιήσει μια πληθώρα εξετάσεων. Ο γιατρός επιλέγει την εξέταση που θα ήθελε να γίνει και το σύστημα ενημερώνεται για την συγκεκριμένη αποστολή παραπεμπτικού. Για πρακτικούς λόγους, η εφαρμογή υποστηρίζει έναν τύπο εξέτασης για κάθε εργαστήριο και πιο συγκεκριμένα Γενική Αίματος(Αιματολογικό), Αμυλάση Ούρων(Μικροβιολογικό), Ουρία (Βιοχημικό) και Μαστογραφία(Ακτινολογικό).

Οι παραπάνω λειτουργίες που διατίθενται στο γιατρό προϋποθέτουν την εισαγωγή του στο σύστημα και την ταυτοποίηση της εγκυρότητας των στοιχείων του. Το παρακάτω usecasediagram, το οποίο υλοποιήθηκε με το εργαλείο Agilian 10.2-Visualparadigm (visual-paradigm), τιςπαρουσιάζει επακριβώς.(uml-diagrams)



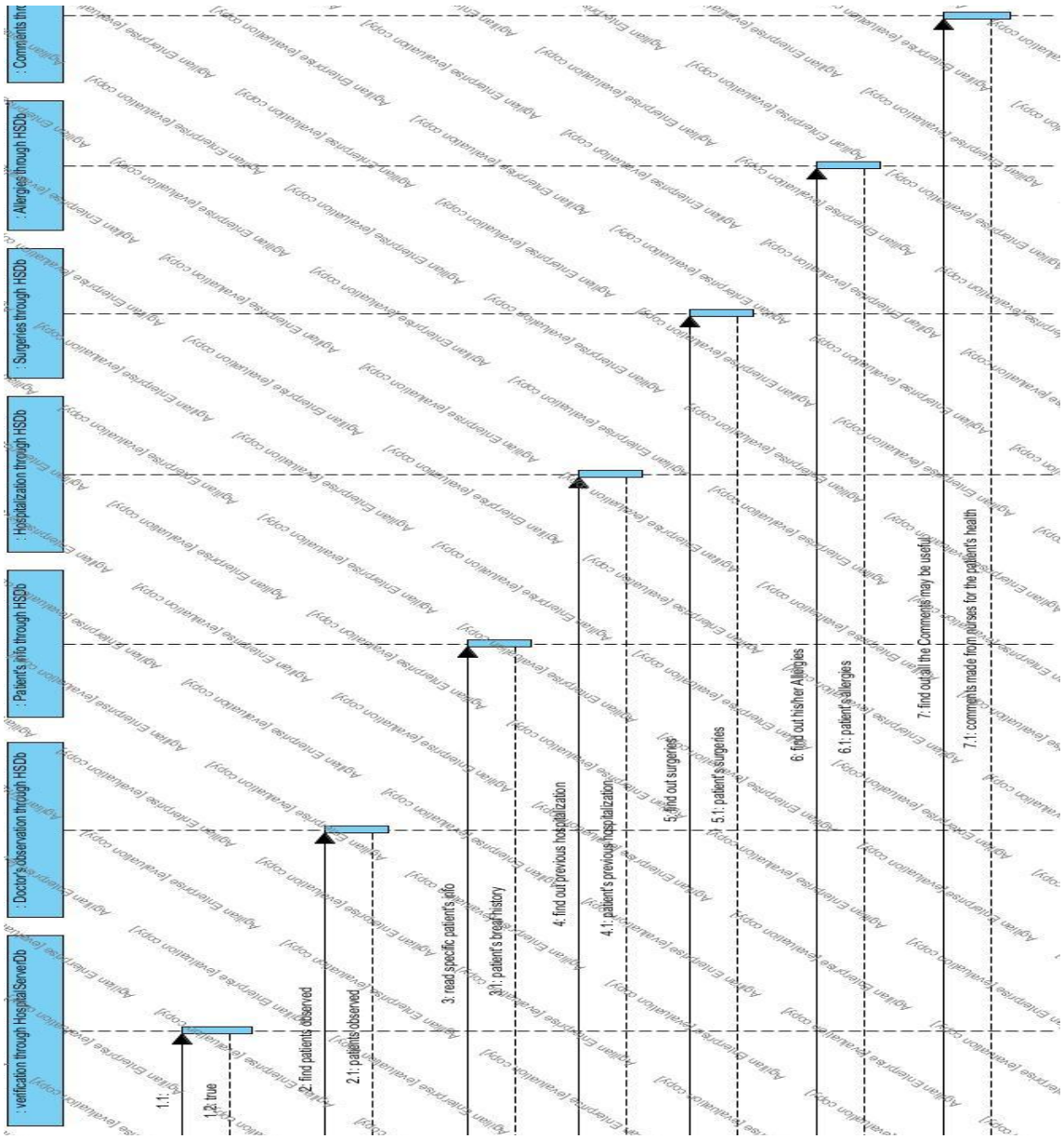
Εικόνα 10: UseCaseDiagram για Γιατρό

Κατόπιν, με τη βοήθεια sequencediagrams(διαγράμματα ακολουθίας), τα οποία υλοποιήθηκαν επίσης με το εργαλείο Agilian 10.2- Visualparadigm, παρουσιάζεται η απεικόνιση μιας συνεργασίας αντικειμένων. Σε αυτού του είδους

τα διαγράμματα απεικονίζονται οι ανταλλαγές μηνυμάτων μεταξύ των αντικειμένων του συστήματος για την επίτευξη ενός σεναρίου μιας περίπτωσης χρήσης. Για έναν προγραμματιστή είναι εύκολο κοιτάζοντας ένα διάγραμμα ακολουθίας να καταλάβει πώς ακριβώς πρέπει να γράψει τον κώδικα και κυρίως να αντιληφθεί εύκολα τη σειριακή ακολουθία των διαφόρων γεγονότων για την τελική επίτευξη του στόχου.Ένα διάγραμμα ακολουθίας δείχνει, ως παράλληλες κάθετες γραμμές (ρέλια), διάφορες διαδικασίες ή αντικείμενα που ζουν ταυτόχρονα, και, ως οριζόντια βέλη, τα μηνύματα που ανταλλάσσονται μεταξύ τους, με τη σειρά με την οποία εμφανίζονται. Αυτό επιτρέπει τον προσδιορισμό των απλών σεναρίων χρόνου εκτέλεσης σε ένα γραφικό τρόπο.(uml-diagrams)

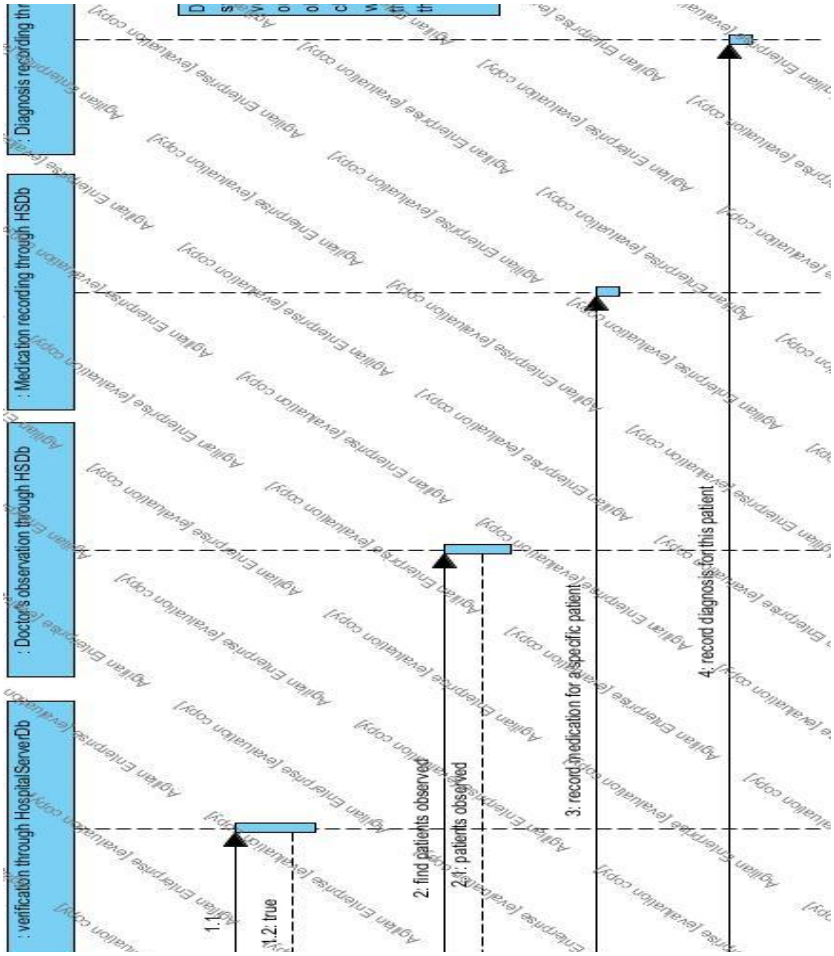
Παρακάτω παρατίθενται τα sequencediagramsγια την εκάστοτε περίπτωση χρήσης του γιατρού. Εφόσον κάποια από τα διαγράμματα που προέκυψαν είναι μεγάλου μεγέθους, επισυνάφθηκαν με rotate, προς διευκόλυνση της ανάγνωσής τους.

Όσο αφορά το **πρώτο** σενάριο χρήσης του γιατρού, **ανάγνωση ιστορικού ασθενή**, αφότου ο χρήστης δώσει τα στοιχεία του και ελεγχθεί η εγκυρότητά τους, το σύστημα κάνει κλήση προς την κεντρική βάση δεδομένων(server)ώστε να βρεθούν όλοι οι ασθενείς που σχετίζονται με το συγκεκριμένο γιατρό. Μετά την επιλογή ενός από αυτούς, γίνονται κλήσεις προς την κεντρική βάση για εύρεση των προσωπικών δεδομένων, χειρουργκών επεμβάσεων, αλλεργιών και σχολίων που ενδεχομένως έχουν γίνει από νοσηλευτές, δίνοντας ως κλειδί το idτου ασθενή που επιλέχθηκε προς παρακολούθηση. Στη συνέχεια, επιστρέφονται ως απάντηση, όσα στοιχεία ζήτησε να μάθει ο γιατρός.



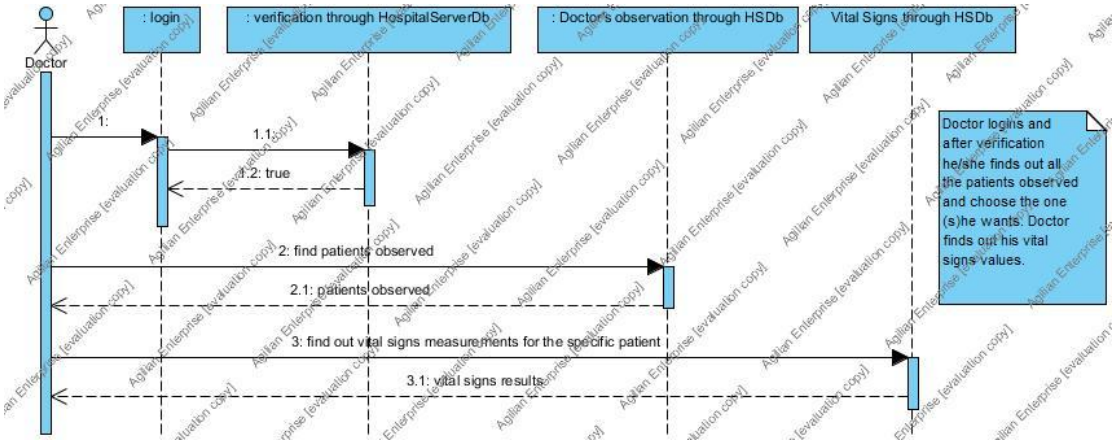
Εικόνα 11: SequenceDiagram για ιστορικό ασθενή

Όσο αφορά το **δεύτερο** και **τρίτο** σενάριο χρήσης, **προσθήκη διάγνωσης**και **φαρμακευτικής αγωγής**, κατά τον ίδιο τρόπο, εφόσον ο γιατρός εισέλθει στο σύστημα και ταυτοποιηθούν τα στοιχεία του, επιλέγει έναν ασθενή που επιθυμεί και εισάγει για αυτόν κάποια-ες διάγνωση και φαρμακευτική αγωγή. Σε αυτή την περίπτωση δεν υπάρχει κάποια απάντηση από το σύστημα, απλώς ενημερώνεται η κεντρική βάση δεδομένων με τα νέα στοιχεία.

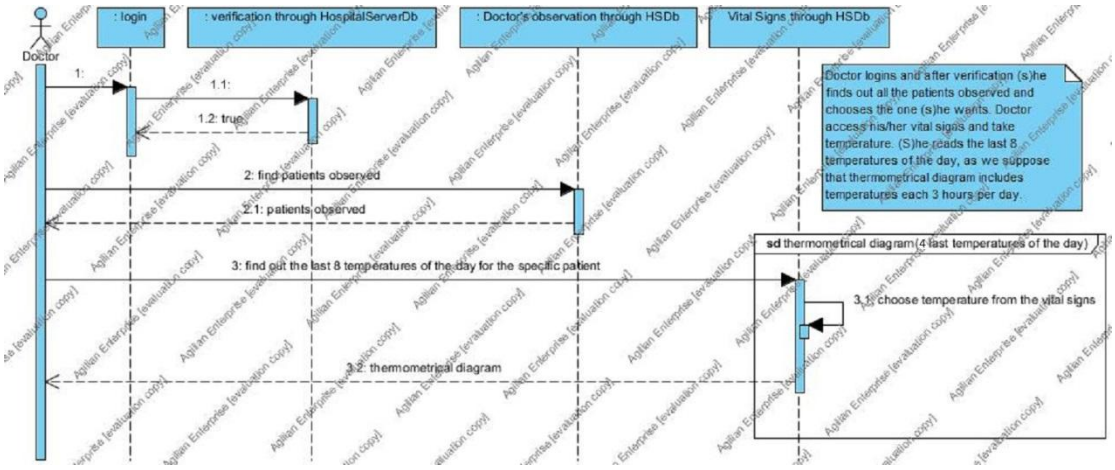


Εικόνα 12: SequenceDiagram για προσθήκη διάγνωσης & φαρμάκου

Όσο αφορά το **τέταρτο** σενάριο χρήσης, **παρατήρηση προόδου ασθενή** και **ανάγνωση θερμομετρικού διαγράμματος**, αφότου ο χρήστης εισαχθεί με έγκυρα στοιχεία στο σύστημα και επιλέξει τον προς παρατήρηση ασθενή, κάνει κλήση στην κεντρική βάση από την οποία λαμβάνει τις μετρήσεις για τα ζωτικά σημεία συμπεριλαμβανομένου και της θερμοκρασίας. Για τη δημιουργία του θερμομετρικού διαγράμματος, από υπάρχει ένα miniυποσύστημα, στο οποίο ο χρήστης λαμβάνει ημερισίως τις 4 τιμές θερμοκρασίας του χρήστη και κατόπιν επεξεργασίας, δημιουργείται το τελικό θερμομετρικό διάγραμμα για maximum5 ημέρες πριν της σημερινής(σημερινή θεωρείται η ημερομηνία που αναγράφεται στην κινητή συσκευή κατά τη διάρκεια που τρέχει η εφαρμογή).

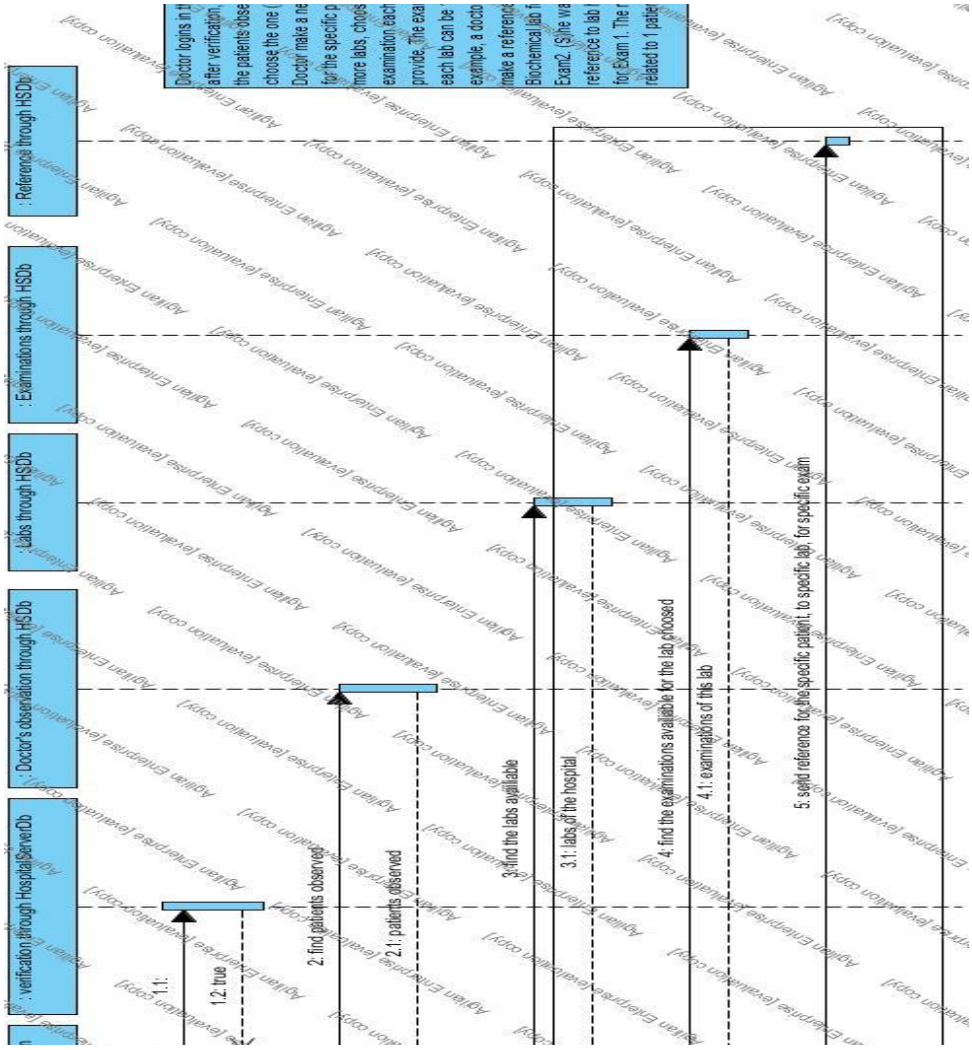


Εικόνα 13: SequenceDiagram για ζωτικά σημεία



Εικόνα 14: SequenceDiagram για θερμομετρικό διάγραμμα

Τέλος, για το **πέμπτο** σενάριο χρήσης, **αποστολή παραπεμπτικών σε εργαστήρια**, ο γιατρός μετά την έγκυρη εισαγωγή του στο σύστημα και αφότου επιλέξει προς παρατήρηση ασθενή, στέλνει referencesε ένα ή περισσότερα εργαστήρια για μία η περισσότερες εξετάσεις. Για το λόγο αυτό, στο αντίστοιχο sequencediagram, που παρατίθεται, απεικονίζεται με μορφή loopη διαδικασία εύρεσης εργαστηρίου, εν συνεχεία εξέτασης που σχετίζεται με το συγκεκριμένο εργαστήριο και τέλος η δημιουργία νέου referenceγια συγκεκριμένο ασθένη, εργαστήριο και εξέταση.



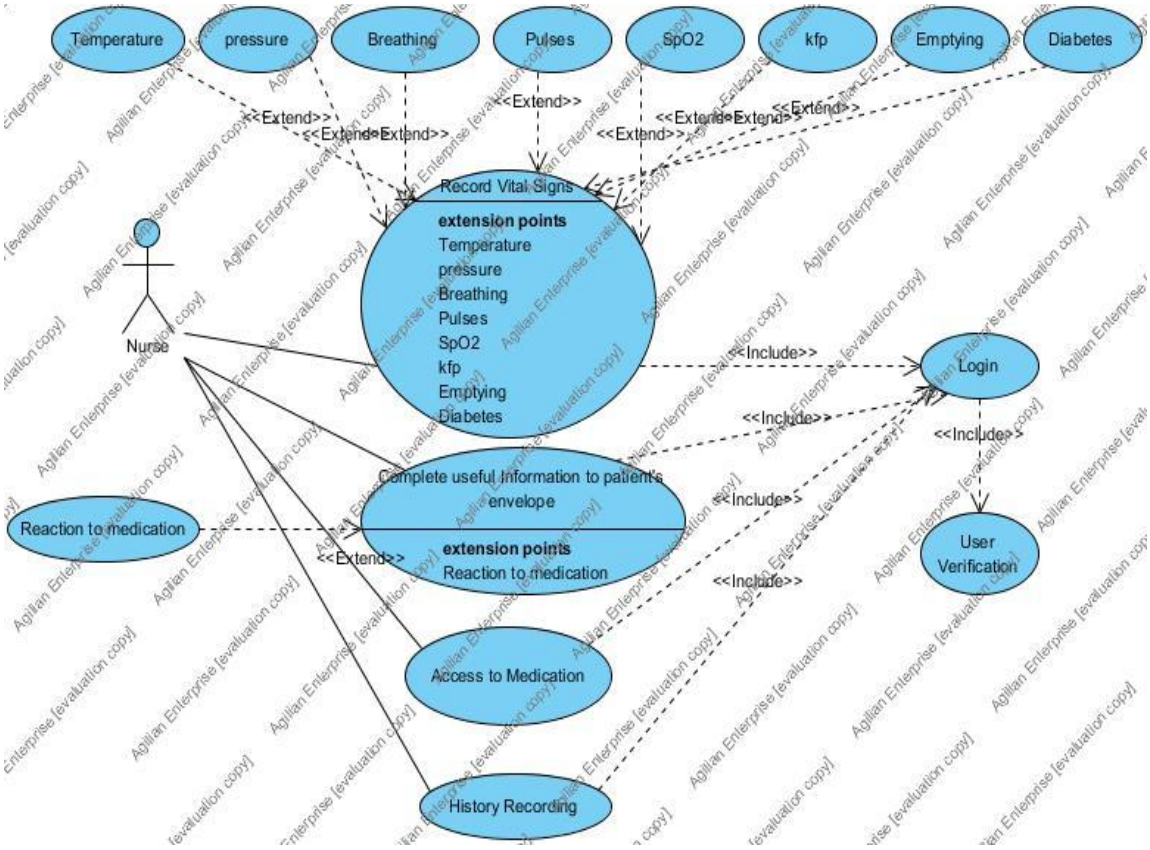
Εικόνα 15: SequenceDiagram για αποστολή παραπεμπτικών

2.2.2 Λειτουργικές Απαιτήσεις Νοσηλευτή

Οι λειτουργικές απαιτήσεις των νοσηλευτών είναι οι εξής:

1. Εισαγωγή νέου ασθενή. Ο νοσηλευτής έχει τη δυνατότητα να εισάγει νέο ασθενή γράφοντας τα στοιχεία του προφίλ του. Κατά τη διαδικασία εισαγωγής των νέων στοιχείων γίνονται έλεγχοι εγκυρότητας όσο αφορά πεδία όπως το email, url κτλ. Θα δείξουμε και παρακάτω πιο λεπτομερώς τους ελέγχους που λαμβάνουν χώρα.
2. Πληροφόρηση τους για το προσωπικό και ιατρικό ιστορικό των ασθενών. Ο νοσηλευτής έχει τη δυνατότητα να δει τα προσωπικά στοιχεία του επιλεγμένου ασθενή (όνομα, επίθετο, ημ/νία γέννησης, αμκα, αφμ, ημ/νία εισαγωγής- εξαγωγής από το νοσοκομείο κτλ.), τυχόν αλλεργίες που παρουσιάζει, τυχόν χειρουργικές επεμβάσεις στις οποίες έχει υποβληθεί καθώς και τα παροντικά του συμπτώματα, ώστε να έχει μια πλήρη προσωπική και κλινική του εικόνα.
3. Καταγραφή ζωτικών σημείων. Ο νοσηλευτής καταγράφει τις μετρήσεις των ζωτικών σημείων του επιλεγμένου ασθενή(πίεση, αναπνοές, σφυγμόι, SpO2, ΚΦΠ, κεννώσεις, σάκχαρο, θερμοκρασία) για μία συγκεκριμένη ώρα που επιλέγει εκ των 05:30, 11:30, 17:30, 23:30. Η εφαρμογή «υποχρεώνει» το χρήστη να επιλέξει μία από τις παραπάνω ώρες κυρίως για λόγους ευκολίας κατά τη δημιουργία του θερμομετρικού διαγράμματος, αλλά και για να εξασφαλιστεί πως οι μετρήσεις γίνονται 4 φορές το 24 ωρο όπως συμβαίνει στην πραγματικότητα. Αξίζει να σημειωθεί πως οι τιμές που δίνει ο χρήστης ελέγχονται για την εγκυρότητά τους, πράγμα που θα αναλύσουμε ενδελεχέστερα παρακάτω.
4. Ενημέρωση για τη φαρμακευτική αγωγή που λαμβάνει ο χρήστης για όσο διάστημα νοσηλεύεται(ονομασία φαρμάκου, δοσολογία, ημ/νία εκκίνησης και διακοπής της αγωγής).
5. Εγγραφή κάποιου σχολίου σχετικού με τον ασθενή που έχει επιλεχθεί. Το σχόλιο θα μπορούσε να είναι οτιδήποτε θεωρείται διαφωτιστικό για τον επιβλέποντα γιατρό. Για παράδειγμα, κάποια αντιδραση στη φαμακευτική αγωγή ή κάποιο απρόσμενο συμπτωμα κτλ.

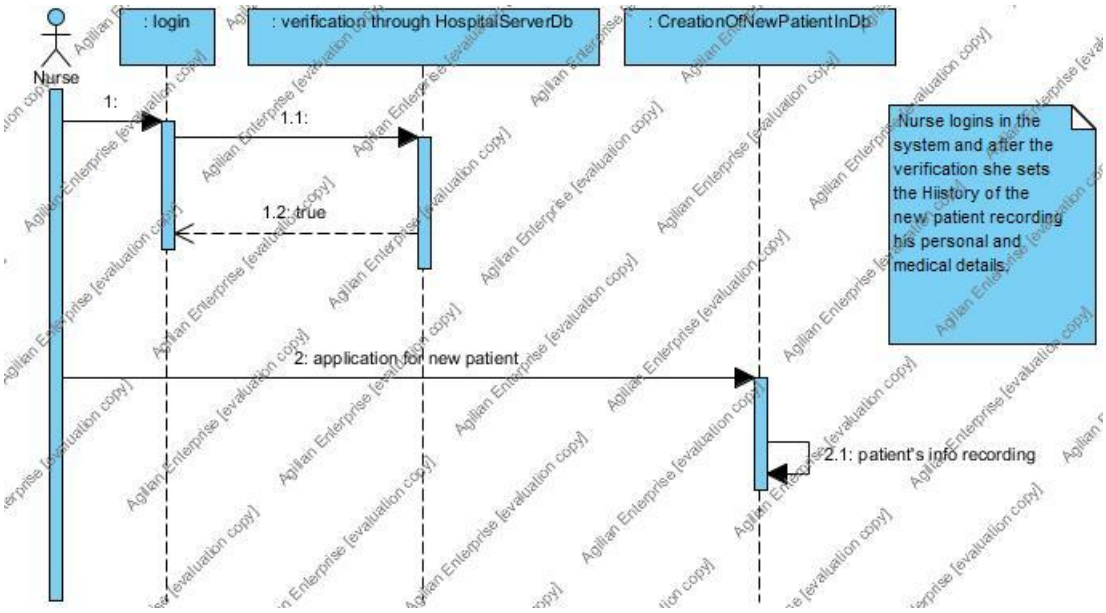
Οι παραπάνω λειτουργίες που διατίθενται στον νοσηλευτή προϋποθέτουν την εισαγωγή του στο σύστημα και την ταυτοποίηση της εγκυρότητας των στοιχείων του. Το παρακάτω usecasediagram, το οποίο υλοποιήθηκε με το εργαλείο Agilian 10.2- Visualparadigm, τιςπαρουσιάζει επακριβώς.



Εικόνα 16: UseCaseDiagramγια Νοσηλευτή

Παρακάτω παρατίθενται τα sequencediagramsγια την εκάστοτε περίπτωση χρήσης του νοσηλευτή. Εφόσον κάποια από τα διαγράμματα που προέκυψαν είναι μεγάλου μεγέθους, επισυνάφθηκαν με rotate, προς διευκόλυνση της ανάγνωσής τους.

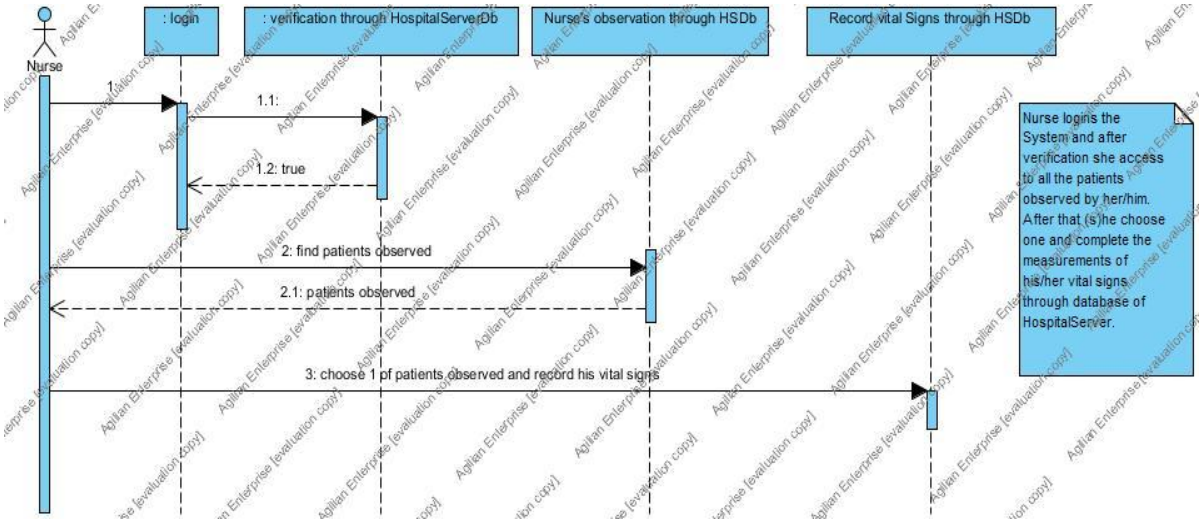
Όσο αφορά το **πρώτο** σενάριο χρήσης του νοσηλευτή, **εισαγωγή νέου ασθενή**, αφού ο νοσηλευτής δώσει τα στοιχεία του και μελετηθεί η εγκυρότητά τους, ακολουθεί η καταγραφή των στοιχείων του νέου ασθενή και η φόρτωσή τους στην βάση.



Εικόνα 17: SequenceDiagram για εισαγωγή ασθενή

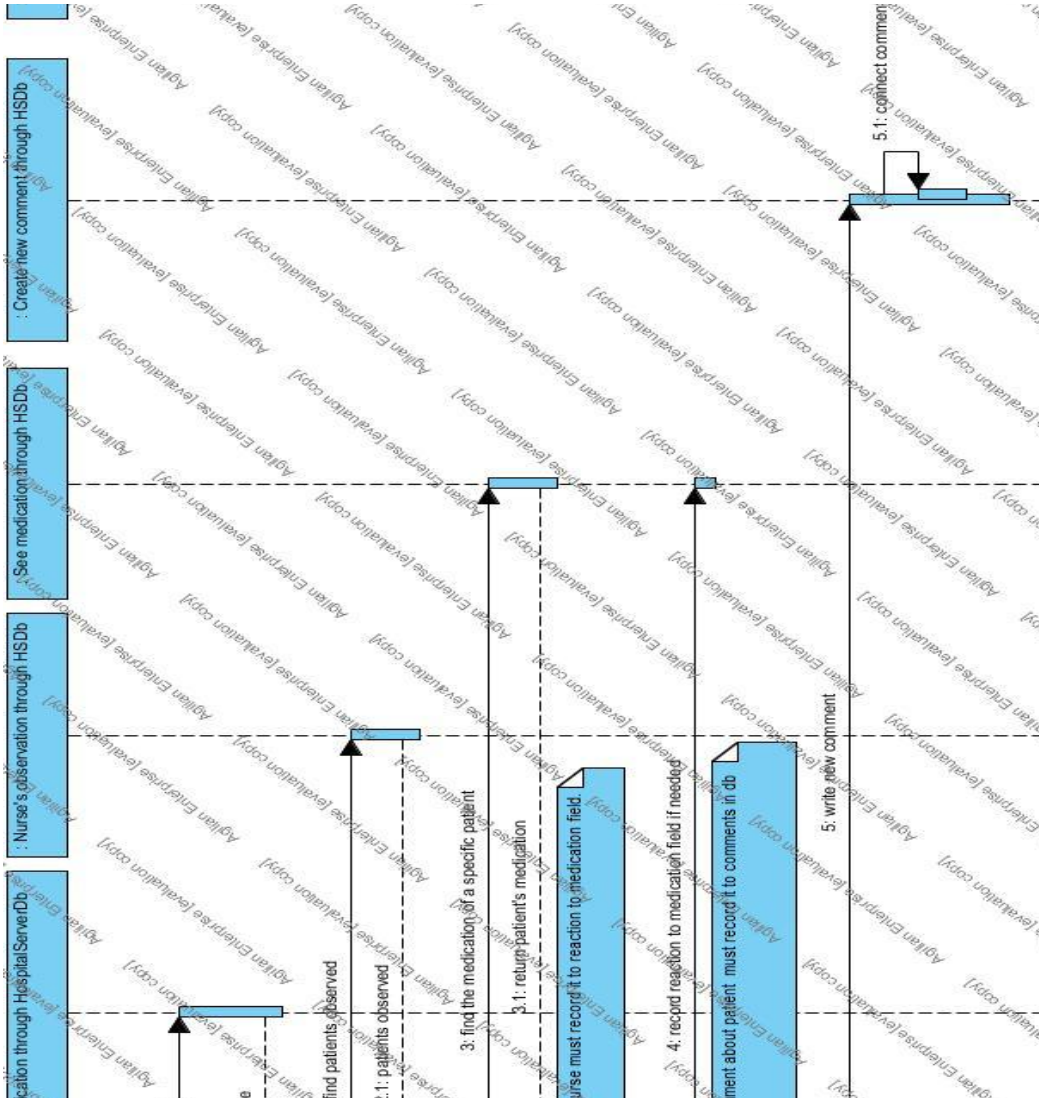
Όσο αφορά το **δεύτερο** σενάριο χρήσης του νοσηλευτή, **ανάγνωση ιστορικού ασθενή,** είναι ακριβώς ίδιο με το πρώτο σενάριο χρήσης των γιατρών και έχει περιγραφεί με λεπτομέρεια λίγο πιο πάνω. Το διάγραμμά ακολουθίας έχει επίσης παρατεθεί πιο πάνω στα sequence των γιατρών(Εικόνα 11).

Το **τρίτο** σενάριο χρήσης των νοσηλευτών, **καταγραφή ζωτικών σημείων**, και πάλι προϋποθέτει την εγκυρότητα των στοιχείων εισαγωγής του νοσηλευτή στο σύστημα και την επιλογή ενός εκ των ασθενών. Κατόπιν, όπως και στα περισσότερα διαγράμματα ακολουθίας που έχουμε ήδη δει, γίνεται η καταγραφεί από τον νοσηλευτή των μετρήσεων και τα νέα δεδομένα αποθηκεύονται στην κεντρική βάση.



Εικόνα 18: SequenceDiagram για καταγραφή ζωτικών σημείων

Το **τέταρτο** και **πέμπτο** σενάριο χρήσης, **ανάγνωση φαρμακευτικής αγωγής** και **καταγραφή σχολίου**, προϋποθέτουν επίσης την εγκυρότητα των στοιχείων εισαγωγής του νοσηλευτή στο σύστημα και την επιλογή ενός εκ των ασθενών. Αφότου ο νοσηλευτής βρει τη φαρμακευτική αγωγή του προς παρακολούθηση ασθενή, σε περίπτωση που έχει να σχολίασει κάτι σχετικό με την αντίδραση στο φάρμακο, μπορεί να εισάγει σαν σχόλιο αυτή του την παρατήρηση. Επίσης, γενικότερα του δίνεται η δυνατότητα καταγραφής κάποιου σχολίου που θα βοηθούσε τη δουλειά του θεράποντα γιατρού και θα έδινε περισσότερες πληροφορίες για την κλινική εικόνα του ασθενή



Εικόνα 19: SequenceDiagram για ανάγνωση φαρμακευτικής αγωγής και προσθήκης σχολίου.

Θα πρέπει να σημειωθεί πως τα συγκεκριμένα διαγράμματα υλοποιήθηκαν στην πριν την υλοποίηση του κώδικα της εφαρμογής και γι’αυτό το λόγο δεν αναφέρεται κάπου η σύνδεση με την τοπική βάση, παρα μόνο με την κεντρική του server.

3. ΣΧΕΔΙΑΣΜΟΣ ΚΕΝΤΡΙΚΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

3.1. Εργαλείο XAMPP

Εικόνα 20: Xampp

Το **XAMPP** είναι ένα πακέτο προγραμμάτων ελεύθερουλογισμικού, λογισμικού ανοικτού κώδικα και ανεξαρτήτου πλατφόρμας το οποίο περιέχει τον εξυπηρετητή ιστοσελίδων httpApache, την βάση δεδομένων MySQL και ένα διερμηνέα για κώδικα γραμμένο σε γλώσσες προγραμματισμού PHP και Perl. Τρέχει σε MicrosoftWindows, Linux, Solaris και MacOSX και χρησιμοποιείται ως πλατφόρμα για την σχεδίαση και ανάπτυξη ιστοσελίδων με την τεχνολογίες όπως PHP, JSP και Servlets.Το XAMPP είναι ακρωνύμιο και αναφέρεται στα παρακάτω αρχικά:

- **X** (αναφέρεται στο «cross-Platform» που σημαίνει λογισμικό ανεξάρτητο πλατφόρμας)
- **A**pache HTTP εξυπηρετητής
- **M**ySQL
- **P**HP
- **P**erl

Το XAMPP προϋποθέτει μόνο τα λογισμικά συμπίεσης αρχείων zip, tar, 7z ή exe κατά την διάρκεια της εγκατάστασης. Επίσης έχει δυνατότητα αναβάθμισης σε νέες εκδόσεις του εξυπηρετητή ιστοσελίδων httpApache, της βάσης δεδομένων MySQL, της γλώσσας PHP και Perl και συμπεριλαμβάνει τα πακέτα OpenSSL και το phpMyAdmin.

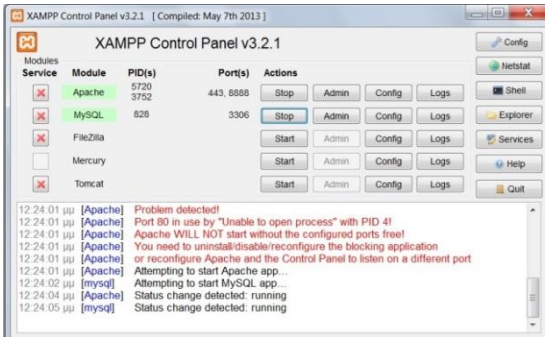
Επίσημα οι σχεδιαστές του XAMPP προόριζαν το λογισμικό ως εργαλείο ανάπτυξης και δοκιμής ιστοσελίδων τοπικά στον υπολογιστή χωρίς να είναι απαραίτητη η σύνδεση στο διαδίκτυο. Για να είναι δυνατή η χρήση του, πολλές σημαντικές λειτουργίες ασφάλειας έχουν απενεργοποιηθεί . Στην πράξη το XAMPP ορισμένες φορές χρησιμοποιείται και για την φιλοξενία ιστοσελίδων. Υπάρχει ειδικό εργαλείο το οποίο περιέχεται για την προστασία με κωδικό των σημαντικών μερών.

Υποστηρίζει επίσης την δημιουργία και διαχείριση βάσεων δεδομένων τύπου MySQL και SqlLite.(wikipedia)

Όταν το XAMPP εγκατασταθεί στον τοπικό υπολογιστή διαχειρίζεται τον localhost ως ένα απομακρυσμένο κόμβο, ο οποίος συνδέεται με το πρωτόκολλο μεταφοράς αρχείων FTP. Η σύνδεση στον localhost μέσω του FTP μπορεί να γίνει με το όνομα χρήστη «newuser» και το κωδικό «wampp». Για την βάση δεδομένων MySQL υπάρχει ο χρήστης «root» χωρίς κωδικό πρόσβασης.

Το συγκεκριμένο εργαλείο ουσιαστικά μετατρέπει τον τοπικό υπολογιστή σε server. Η διαδικασία που ακολούθησα βήμα προς βήμα είναι η εξής:

- Εγκατάσταση του Xampp στον υπολογιστή
- Εκκίνηση Apache & MySQL
- Κατά την εκκίνηση του Apacheέχω errorγι'αυτό χρειάζεται να αλλάξω το portαπό 80 που ήταν αρχικά σε 8888. Να σημειωθεί πως εναλλακτική θα ήταν να αλλάξω το portπου δεσμεύει το Skypeαπό τις ρυθμίσεις του καθώς το 80 συνήθως δεσμεύεται από το συγκεκριμένο πρόγραμμα.(how-to-solve-xampp-problem)
- Μετά την παραπάνω αλλαγή ο Apacheλειτουργεί κανονικά στο url : http://localhost:8888.Εμφανίζεται η παρακάτω εικόνα και τώρα πλέον σίγουρα λειτουργεί το εργαλείο.



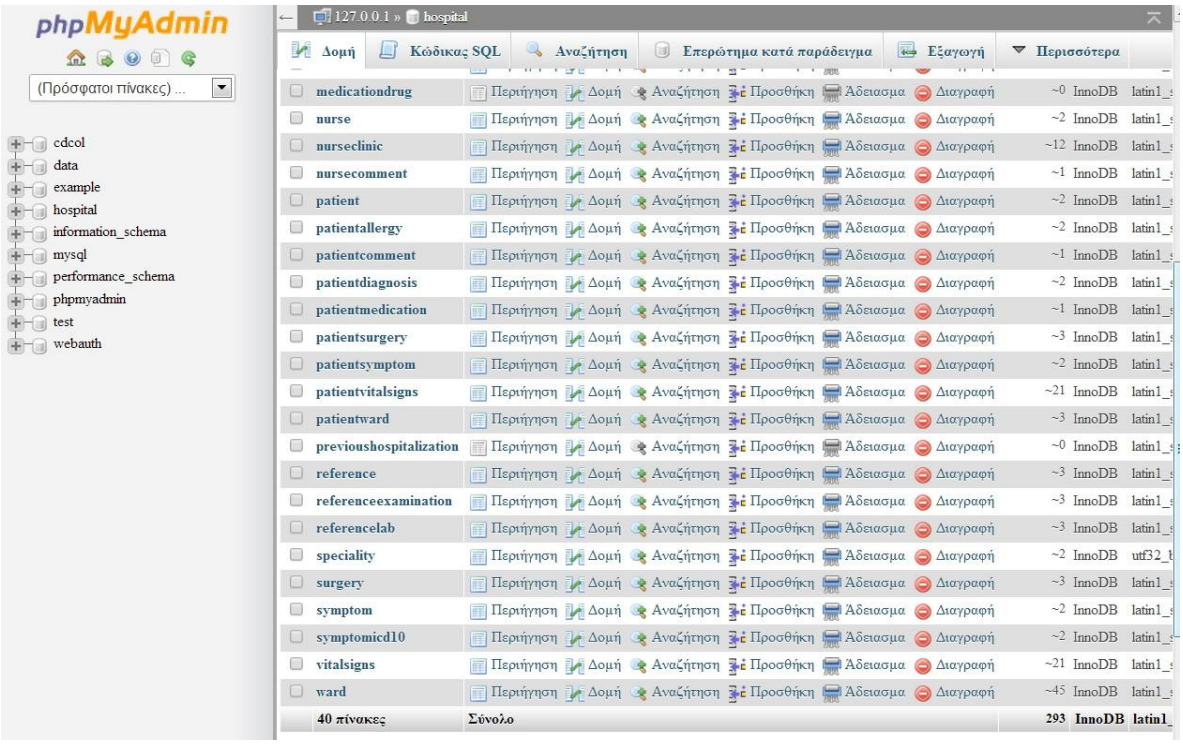
Εικόνα 21: Εκκίνηση Xampp

Εικόνα 22: Xamppστον περιηγητή

3.2. PhpMyAdmin

Το εργαλείο PhpMyAdminπροσφέρει τη δυνατότητα δημιουργίας βάσης δεδομένων στο server. Προσφέρει αρκετές δυνατότητες, κάποιες από τις οποίες είναι η δημιουργία, μετατροπή και διαγραφή βάσεων, πινάκων,πεδίων και γραμμών καθώς επίσης και η εκτέλεση ερωτημάτων sql.(youtube)

Αρχικά λοιπόν ξεκινάμε με τη δημιουργία της βάσης για την εφαρμογή μας. Η βάση ονομάζεται **hospital**είναι τύπου **InnoDB**και σύνθεσης **latin1_swedish_ci**. Αποτελείται συνολικά από 40 πίνακες τους οποίους αναλύουμε συνοπτικά στην επόμενη υποενότητα. Στον browser φαίνεται η εξής εικόνα:



Εικόνα 23: Κεντρική βάση hospital

3.3. Βάση Δεδομένων

3.3.1 Πίνακες – Icd10

Οι πίνακες που υλοποιήθηκαν είναι όπως προαναφέρθηκε στο σύνολό τους 40 και παρουσιάζονται με αλφαβητική σειρά παρακάτω:

127.0.0.1 » hospital » allergy

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγωγ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	AllergyId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	NameAllergy	varchar(50)	utf32_bin		Όχι	Καμία	
3	DescriptionAllergy	text	utf32_bin		Όχι	Καμία	
4	EncodeAllergy	varchar(50)	utf32_bin		Όχι	Καμία	

Εικόνα 24: Πίνακαςallergy

Ο πίνακας **allergyicd10**, είναι ουσιαστικά ένας ενδάμεσος πίνακας που συνδέει τις αλλεργίες με την κωδικοποίηση icd10.Θα ειπωθούν παρακάτω λιγο πιο αναλυτικά πληροφορίες σχετικές με τη δυγκεκριμένη κωδικοποίηση.

127.0.0.1 » hospital » allergyicd10

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	AllergyIcd10Id	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	AllergyId	int(11)			Όχι	Καμία	
3	Icd10Id	int(11)			Όχι	Καμία	

Εικόνα 25: Πίνακας allergyicd10

127.0.0.1 » hospital » clinic

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**ClinicId**	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	**NameClinic**	varchar(50)	utf32_bin		Όχι	*Καμία*	

Εικόνα 26: Πίνακας clinic

127.0.0.1 » hospital » comment

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**CommentId**	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	**DateComment**	datetime			Όχι	*Καμία*	
3	**About**	text	utf32_bin		Όχι	*Καμία*	

Εικόνα 27: Πίνακας comment

127.0.0.1 » hospital » diagnosis

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**DiagnosisId**	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	**NameDiagnosis**	varchar(50)	utf32_bin		Όχι	*Καμία*	

Εικόνα 28: Πίνακας diagnosis

127.0.0.1 » hospital » diagnosisicd10

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	DiagnosisIcd10Id	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	DiagnosisId	int(11)			Όχι	Καμία	
3	Icd10Id	int(11)			Όχι	Καμία	

Εικόνα 29:Πίνακας diagnosisicd10

127.0.0.1 » hospital » doctor

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγω

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	DoctorId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	LnameDoctor	varchar(50)	utf32_bin		Όχι	Καμία	
3	FnameDoctor	varchar(50)	utf32_bin		Όχι	Καμία	
4	UsernameDoctor	varchar(50)	utf32_bin		Όχι	Καμία	
5	PasswordDoctor	varchar(50)	utf32_bin		Όχι	Καμία	
6	SpecialityDoctor	int(11)			Όχι	Καμία	

Εικόνα 30: Πίνακας doctor

127.0.0.1 » hospital » doctorclinic

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	DoctorClinicId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	DoctorId	int(11)			Όχι	Καμία	
3	ClinicId	int(11)			Όχι	Καμία	

Εικόνα 31: Πίνακας doctorclinic

Ο πίνακας **drug**σχετίζεται με τις δραστικές ουσίες που περιέχουν τα φάρμακα. Για λόγους ευκολίας εντέλει δεν τον χρησιμοποιήσαμε. Παρόλα αυτά θα μπορούσε να συμπληρωθεί ως επέκταση στη συγκεκριμενη ή παραπλήσιες εργασίες, καθώς τα

συστατικά των φαρμάκων αποτελούν χρήσιμη πληροφορία για γιατρούς και νοσηλευτές.

127.0.0.1 » hospital » drug

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγω

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**DrugId**	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	**NameDrug**	varchar(50)	utf32_bin		Όχι	*Καμία*	
3	**EncodeDrug**	varchar(50)	utf32_bin		Όχι	*Καμία*	
4	**FormDrug**	varchar(50)	utf32_bin		Όχι	*Καμία*	
5	**AtcCategory**	varchar(50)	utf32_bin		Όχι	*Καμία*	
6	**ActiveSubstance**	varchar(50)	utf32_bin		Όχι	*Καμία*	
7	**CompositionDrug**	varchar(50)	utf32_bin		Όχι	*Καμία*	

Εικόνα 32: Πίνακας drug

127.0.0.1 » hospital » examination

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγω

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**ExamId**	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	**NameExam**	varchar(50)	utf32_bin		Όχι	*Καμία*	
3	**LabId**	int(11)			Όχι	*Καμία*	
4	**DescriptionExam**	text	utf32_bin		Όχι	*Καμία*	

Εικόνα 33: Πίνακας examination

Ο πίνακας **icd10**αποθηκεύει την ονομασία και την αντίστοιχη κωδικοποίηση icd10 ορισμένων ασθενειών, συμπτωμάτων, αλλεργιών.

← 127.0.0.1 » hospital » icd10

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγ

	#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
☐	1	Icd10Id	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
☐	2	NameDisease	varchar(300)	utf32_bin		Όχι	*Καμία*	
☐	3	EncodeDisease	varchar(300)	utf32_bin		Όχι	*Καμία*	

Εικόνα 34: Πίνακας icd10

Η **Διεθνής Στατιστική Ταξινόμηση Νοσημάτων και Συναφών Προβλημάτων Υγείας** (**ICD**) αποτελεί μία κωδικοποίηση των νοσημάτων από τον Παγκόσμιο Οργανισμό Υγείας που αρχικά έχει σαν σκοπό την όσο το δυνατόν καλύτερη ανάλυση και επεξεργασία των ιατρικών διαγνώσεων παγκοσμίως. Ο Παγκόσμιος Οργανισμός Υγείας διατηρεί στο διαδίκτυο μία έκδοση (στα αγγλικά) της τελευταίας αναθεώρησης η οποία είναι αναζητήσιμη.

Δεν μπορεί να χρησιμοποιηθεί για πολύ εξειδικευμένους σκοπούς καθώς δεν είναι δυνατή η καταχώρηση πολύ αναλυτικών καταστάσεων όπως "για ποιον λόγο ήρθε στο νοσοκομείο", είναι όμως χρήσιμη στην ομαδοποίηση και ανάλυση διαγνώσεων για επιδημιολογικούς λόγους.

Η δέκατη αναθεώρηση της ταξινόμησης(**ICD10**) ξεκίνησε το 1983 και ολοκληρώθηκε το 1992. Το Ελληνικό Υπουργείο Υγείας έχει μεταφράσει αυτήν την αναθεώρηση. Η βασική ταξινόμηση γίνεται με τριψήφιους κωδικούς που είναι και υποχρεωτικό επίπεδο κωδικοποίησης για τη διεθνή αναφορά στη βάση δεδομένων της θνησιμότητας της ICD και για γενικές διεθνείς συγκρίσεις. Ακολουθεί μία δεύτερη κατηγοριοποίηση με τετραψήφιο κωδικό η οποία δεν είναι υποχρεωτική αλλά γενικά χρησιμοποιείται και σε μερικές κατηγορίες αποτελεί ένα αναπόσπαστο τμήμα της ICD.

Θα πρέπει να σημειωθεί πως η συμπλήρωση του πίνακα υλοποιήθηκε βάσει ρεαλιστικού πίνακα icd10 που βρήκαμε στον παγκόσμιο ιστό και λόγω μεγάλου μεγέθους του παρατίθεται αποσπασματικά. Περιλαμβάνει στο σύνολο 11008 εγγραφές εκ των οποίων συμπεριλάβαμε, για εξοικονόμηση χώρου και χρόνου, στον αντίστοιχο πίνακα της βάσης 19. Να σημειωθεί επίσης πως η επιλογη των συγκεκριμένων 19 εγγραφών έγινε με παντελώς τυχαίο τρόπο.(wikipedia)

	A	
1	A00	Χολέρα
2	A00.0	Χολέρα από δονάκιο της χολέρας [*Vibrio cholerae*] 01, βιότυπο cholerae
3	A00.1	Χολέρα από στο δονάκιο της χολέρας [*Vibrio cholerae*] 01, βιότυπο eltor
4	A00.9	Χολέρα, μη καθορισμένη
5	A01	Τυφοειδείς και παρατυφοειδείς πυρετοί
6	A01.0	Τυφοειδής πυρετός
7	A01.1	Παρατυφοειδής πυρετός A
8	A01.2	Παρατυφοειδής πυρετός B
9	A01.3	Παρατυφοειδής πυρετός C
10	A01.4	Παρατυφοειδής πυρετός, μη καθορισμένος
11	A02	Άλλες λοιμώξεις από σαλμονέλλα
12	A02.0	Εντερίτιδα από σαλμονέλλα
13	A02.1	Σηψαιμία από σαλμονέλλα
14	A02.2	Εντοπισμένες λοιμώξεις από σαλμονέλλα
15	A02.8	Άλλες καθορισμένες λοιμώξεις από σαλμονέλλα
16	A02.9	Λοίμωξη από σαλμονέλλα, μη καθορισμένη

Εικόνα 35: Απόσπασμα Πίνακαexcelicd10

Προαποφασίσαμε πως ο πίνακας labθα περιέχει 4 εγγραφές, 4 δηλαδή τύπους εργαστηρίων Αιματολογικό, Μικροβιολογικό, Βιοχημικό και Ακτινολογικό (homeomedi).

127.0.0.1 » hospital » lab

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

	#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
☐	1	LabId	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
☐	2	NameLab	varchar(50)	utf32_bin		Όχι	*Καμία*	

Εικόνα 36: Πίνακας lab

Η συγκεκριμένη εφαρμογή υποστηρίζει 4 τύπους εξετάσεων έναν για κάθε εργαστήριο. Ο πίνακας **labresultex1**, αφορά τη Γενική Αίματος και περιλαμβάνει τα πεδία που σχετίζονται με τη συγκεκριμένη εξέταση (imop).

127.0.0.1 » hospital » labresultex1

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγωγή

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	GenikhAimatosResultId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	RGB	varchar(50)	utf32_bin		Όχι	Καμία	
3	HGB	varchar(50)	utf32_bin		Όχι	Καμία	
4	HT	varchar(50)	utf32_bin		Όχι	Καμία	
5	WBC	varchar(50)	utf32_bin		Όχι	Καμία	
6	NEUT	varchar(50)	utf32_bin		Όχι	Καμία	
7	NEU	varchar(50)	utf32_bin		Όχι	Καμία	
8	LYMPH	varchar(50)	utf32_bin		Όχι	Καμία	
9	FysiologikhTimhRGB	varchar(50)	utf32_bin		Όχι	Καμία	
10	FysiologikhTimhHGB	varchar(50)	utf32_bin		Όχι	Καμία	
11	FysiologikhTimhHT	varchar(50)	utf32_bin		Όχι	Καμία	
12	FysiologikhTimhWBC	varchar(50)	utf32_bin		Όχι	Καμία	
13	FysiologikhTimhNEUT	varchar(50)	utf32_bin		Όχι	Καμία	
14	FysiologikhTimhNEU	varchar(50)	utf32_bin		Όχι	Καμία	
15	FysiologikhTimhLYMPH	varchar(50)	utf32_bin		Όχι	Καμία	
16	ExaminationId	int(11)			Όχι	Καμία	
17	Status	int(1)			Όχι	Καμία	
18	ReferenceId	int(11)			Όχι	Καμία	
19	DateReady	datetime			Όχι	Καμία	

Εικόνα 37: Πίνακας Γενική Αίματος

Ο πίνακας **labresultex2**, αφορά την Ουρία και περιλαμβάνει τα πεδία που σχετίζονται με τη συγκεκριμένη εξέταση (imop).

127.0.0.1 » hospital » labresultex2

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγωγή

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	OuriaId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	OuriaAimatos	varchar(50)	utf32_bin		Όχι	Καμία	
3	FysiologikhTimhOurias	varchar(50)	utf32_bin		Όχι	Καμία	
4	ExamId	int(11)			Όχι	Καμία	
5	Status	int(1)			Όχι	Καμία	
6	ReferenceId	int(11)			Όχι	Καμία	
7	DateReady	datetime			Όχι	Καμία	

Εικόνα 38: Πίνακας Ουρία

Ο πίνακας **labresultex3**, αφορά την Αμυλάση Ούρων και περιλαμβάνει τα πεδία που σχετίζονται με τη συγκεκριμένη εξέταση (wikipedia).

127.0.0.1 » hospital » labresultex3

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγωγή

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	AmylashOurwnId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	AmylashOurwn	varchar(50)	utf32_bin		Όχι	Καμία	
3	FysiologikhTimhAmylashs	varchar(50)	utf32_bin		Όχι	Καμία	
4	ExamId	int(11)			Όχι	Καμία	
5	Status	int(1)			Όχι	Καμία	
6	ReferenceId	int(11)			Όχι	Καμία	
7	DateReady	datetime			Όχι	Καμία	

Εικόνα 39: Πίνακας Αμυλάση Ούρων

Ο πίνακας **labresultex4**, αφορά τη Μαστογραφία και περιλαμβάνει τα πεδία που σχετίζονται με τη συγκεκριμένη εξέταση.

127.0.0.1 » hospital » labresultex4

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξα

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	mastografiaId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	comment	varchar(500)	utf32_bin		Όχι	Καμία	
3	ExamId	int(11)			Όχι	Καμία	
4	Status	int(11)			Όχι	Καμία	
5	ReferenceId	int(11)			Όχι	Καμία	
6	DateReady	datetime			Όχι	Καμία	

Εικόνα 40: Πίνακας Μαστογραφία

127.0.0.1 » hospital » medication

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγωγ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	MedicationId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	NameMedication	varchar(200)	utf32_bin		Όχι	Καμία	

Εικόνα 41: Πίνακας medication

127.0.0.1 » hospital » medicationdrug

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Ε

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	MedicationDrug	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	MedicationId	int(11)			Όχι	Καμία	
3	DrugId	int(11)			Όχι	Καμία	

Εικόνα 42: Πίνακας medicationdrug

127.0.0.1 » hospital » nurse

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγω

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	NurseId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	FnameNurse	varchar(50)	utf32_bin		Όχι	Καμία	
3	LnameNurse	varchar(50)	utf32_bin		Όχι	Καμία	
4	UsernameNurse	varchar(50)	utf32_bin		Όχι	Καμία	
5	PasswordNurse	varchar(50)	utf32_bin		Όχι	Καμία	
6	SpecialityIdNurse	int(11)			Όχι	Καμία	

Εικόνα 43: Πίνακας nurse

127.0.0.1 » hospital » nurseclinic

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	NurseClinicId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	NurseId	int(11)			Όχι	Καμία	
3	ClinicId	int(11)			Όχι	Καμία	

Εικόνα 44: Πίνακας nurseclinic

127.0.0.1 » hospital » nursecomment

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	NurseComment	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	NurseId	int(11)			Όχι	*Καμία*	
3	CommentId	int(11)			Όχι	*Καμία*	

Εικόνα 45: Πίνακας nursecomment

127.0.0.1 » hospital » patient

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγωγή

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientId	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	Fname	varchar(50)	utf32_bin		Ναι	*NULL*	
3	Lname	varchar(50)	utf32_bin		Ναι	*NULL*	
4	FatherFname	varchar(50)	utf32_bin		Ναι	*NULL*	
5	MotherFname	varchar(50)	utf32_bin		Ναι	*NULL*	
6	DateBirth	date			Ναι	*NULL*	
7	Address	varchar(50)	utf32_bin		Ναι	*NULL*	
8	City	varchar(50)	utf32_bin		Ναι	*NULL*	
9	Country	varchar(50)	utf32_bin		Ναι	*NULL*	
10	PostCode	varchar(50)	utf32_bin		Ναι	*NULL*	
11	Job	varchar(50)	utf32_bin		Ναι	*NULL*	
12	Sex	varchar(50)	utf32_bin		Ναι	*NULL*	
13	PhoneNumber	varchar(50)	utf32_bin		Ναι	*NULL*	
14	Afm	varchar(50)	utf32_bin		Ναι	*NULL*	
15	AMKA	varchar(50)	utf32_bin		Ναι	*NULL*	
16	MaritalStatus	varchar(50)	utf32_bin		Ναι	*NULL*	
17	DateHospitalization	datetime			Ναι	*NULL*	
18	DateOffHospitalization	datetime			Όχι	*Καμία*	
19	Url	text	utf32_bin		Ναι	*NULL*	
20	Email	varchar(50)	utf32_bin		Ναι	*NULL*	
21	Fund	varchar(50)	utf32_bin		Ναι	*NULL*	
22	Smoking	varchar(50)	utf32_bin		Ναι	*NULL*	
23	Alcohol	varchar(50)	utf32_bin		Ναι	*NULL*	
24	Diet	text	utf32_bin		Ναι	*NULL*	

Εικόνα 46: Πίνακας patient

127.0.0.1 » hospital » patientallergy

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientAllergyId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	AllergyId	int(11)			Όχι	Καμία	

Εικόνα 47: Πίνακας patientallergy

127.0.0.1 » hospital » patientcomment

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientComment	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	CommentId	int(11)			Όχι	Καμία	

Εικόνα 48: Πίνακας patientcomment

127.0.0.1 » hospital » patientdiagnosis

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientDiagnosis	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	DiagnosisId	int(11)			Όχι	Καμία	
4	DateDiagnosis	datetime			Όχι	Καμία	

Εικόνα 49: Πίνακας patientdiagnosis

127.0.0.1 » hospital » patientmedication

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγω

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientMedication	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	MedicationId	int(11)			Όχι	Καμία	
4	Dosage	varchar(50)	utf32_bin		Όχι	Καμία	
5	StartMedication	datetime			Όχι	Καμία	
6	StopMedication	datetime			Όχι	Καμία	

Εικόνα 50: Πίνακας patientmedication

127.0.0.1 » hospital » patientsurgery

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientSurgeryId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	SurgeryId	int(11)			Όχι	Καμία	
4	DateSurgery	datetime			Όχι	Καμία	
5	SurgeryDescription	text	utf32_bin		Όχι	Καμία	

Εικόνα 51: Πίνακας patientsurgery

127.0.0.1 » hospital » patientsymptom

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientSymptomId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	SymptomId	int(11)			Όχι	Καμία	

Εικόνα 52: Πίνακας patientsymptom

127.0.0.1 » hospital » patientvitalsigns

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξα

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientVitalSignsId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	VitalSignsId	int(11)			Όχι	Καμία	

Εικόνα 53: Πίνακας patientvitalsigns

127.0.0.1 » hospital » patientward

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	PatientWardId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	PatientId	int(11)			Όχι	Καμία	
3	WardId	int(11)			Όχι	Καμία	

Εικόνα 54: Πίνακας patientward

Ο πίνακας **previoushospitalization,**σχετίζεται με τις προηγούμενες εισαγωγές του ασθενή σε νοσοκομεία. Προς το παρόν στην παρούσα εφαρμογή δεν χρησιμοποιείται, παρόλα αυτά θα μπορούσε να χρησιμοποιηθεί για μελλοντική επέκταση της εφαρμογής.

127.0.0.1 » hospital » previoushospitalization

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγωγή

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**HospitalizationId**	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	**HospitalizationDate**	datetime			Όχι	*Καμία*	
3	**NameHospital**	varchar(50)	utf32_bin		Όχι	*Καμία*	
4	**DateOffHospitalization**	varchar(50)	utf32_bin		Όχι	*Καμία*	
5	**DiagnosisId**	int(11)			Όχι	*Καμία*	
6	**PatientId**	int(11)			Όχι	*Καμία*	

Εικόνα 55: Πίνακας previoushospitalization

Ο πίνακας **reference** σχετίζεται με τα παραπεμπτικά που αποστέλουν οι γιατροί στα εργαστήρια για να λάβουν χώρα κάποιες εξετάσεις. Το πεδίο Statusουσιαστικά αποτελεί ένα flagπου ενημερώνει σε τι κατάσταση βρίσκεται το παραπεμτικό. Σε περίπτωση που τα αποτελέσματα της εξέτασης που ζήτησε ο γιατρός είναι έτοιμα, παίρνει την τιμή ‘1’ διαφορετικά παίρνει την τιμή ‘0’. Το πεδίο DateReference, είναι ουσιαστικά η ημερομηνία που στάλθηκε το παραπεμπτικό από το γιατρό.

127.0.0.1 » hospital » reference

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | E

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	**ReferenceId**	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	**DateReference**	datetime			Όχι	*Καμία*	
3	**Status**	int(1)			Όχι	*Καμία*	
4	**PatientId**	int(11)			Όχι	*Καμία*	
5	**DoctorId**	int(11)			Όχι	*Καμία*	

Εικόνα 56: Πίνακας reference

Ο πίνακας **referenceexamination**είναι ένας ενδιάμεσος πίνακας που συνδέει το κάθε παραπεμπτικό με μία συγκεκριμένη εξέταση από τις διαθέσιμες.

127.0.0.1 » hospital » referenceexamination

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	ReferenceExId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	ReferenceId	int(11)			Όχι	Καμία	
3	ExaminationId	int(11)			Όχι	Καμία	

Εικόνα 57: Πίνακας referenceexamination

Ο πίνακας **referencelab**είναι ένας ενδιάμεσος πίνακας που συνδέει το κάθε παραπεμπτικό με ένα συγκεκριμένοεργαστήριο από τα διαθέσιμα.

127.0.0.1 » hospital » referencelab

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	ReferenceLabId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	ReferenceId	int(11)			Όχι	Καμία	
3	LabId	int(11)			Όχι	Καμία	

Εικόνα 58: Πίνακας referencelab

127.0.0.1 » hospital » speciality

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	SpecialityId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	NameSpeciality	varchar(50)	utf32_bin		Όχι	Καμία	

Εικόνα 59: Πίνακας speciality

127.0.0.1 » hospital » surgery

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	SurgeryId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	SurgeryName	varchar(50)	utf32_bin		Όχι	Καμία	

Εικόνα 60: Πίνακας surgery

127.0.0.1 » hospital » symptom

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	SymptomId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	NameSymptom	varchar(50)	utf32_bin		Όχι	Καμία	
3	AppearSymptom	varchar(50)	utf32_bin		Όχι	Καμία	

Εικόνα 61: Πίνακας symptom

127.0.0.1 » hospital » symptomicd10

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Ε

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	SymptomIcd10Id	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	SymptomId	int(11)			Όχι	Καμία	
3	Icd10Id	int(11)			Όχι	Καμία	

Εικόνα 62: Πίνακας symptomicd10

127.0.0.1 » hospital » vitalsigns

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη | Εξαγ

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	MeasurementId	int(11)			Όχι	Καμία	AUTO_INCREMENT
2	DateMeasurement	datetime			Όχι	Καμία	
3	Temperature	float			Όχι	Καμία	
4	Pressure	int(11)			Όχι	Καμία	
5	Breathing	int(11)			Όχι	Καμία	
6	Pulses	int(11)			Όχι	Καμία	
7	SpO2	float			Όχι	Καμία	
8	Kfp	float			Όχι	Καμία	
9	Emptying	int(11)			Όχι	Καμία	
10	Diabetes	float			Όχι	Καμία	

Εικόνα 63: Πίνακας vitalsigns

127.0.0.1 » hospital » ward

Περιήγηση | Δομή | Κώδικας SQL | Αναζήτηση | Προσθήκη

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Πρόσθετα
1	WardId	int(11)			Όχι	*Καμία*	AUTO_INCREMENT
2	NumberBed	int(11)			Όχι	*Καμία*	
3	ClinicId	int(11)			Όχι	*Καμία*	

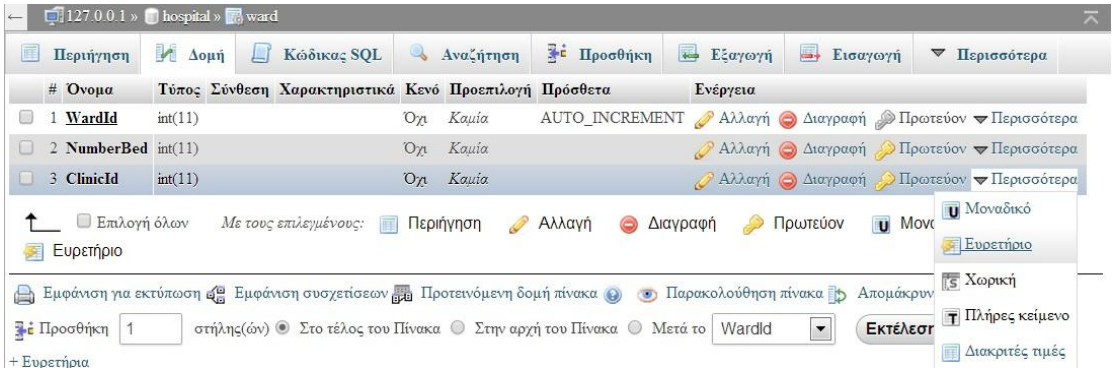
Εικόνα 64: Πίνακας ward

3.3.2 Σχέσεις Πινάκων

Θα πρέπει να σημειωθεί πως ο λόγος ύπαρξης τόσων πινάκων είναι η υλοποίηση σχέσεων πολλά προς πολλά(M-N), η οποία απαιτεί τη δημιουργία νέου ενδιάμεσου πίνακα.

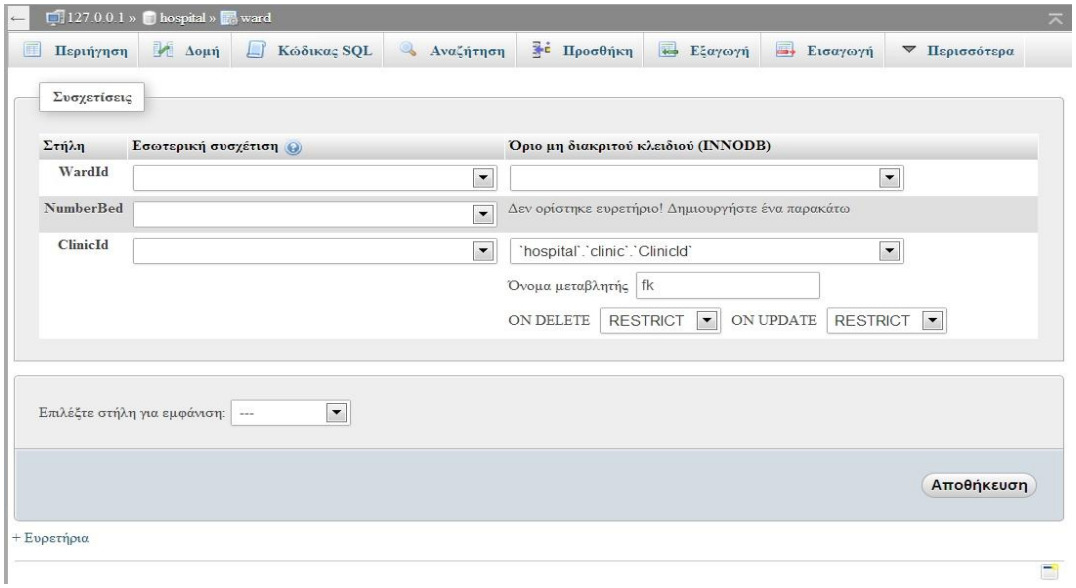
Για τις **σχέσειςένα προς ένα(1-1) και ένα προς πολλά(1-N**)ακολουθήθηκε η εξής διαδικασία:

Στους πίνακες που βρίσκονταν τα ξένα κλειδιά, θέτουμεευρετήριο(index) το εκάστοτε ξένο κλειδί και κατόπιν πάμε στις συσχετίσεις και υλοποιούμε την αντιστοιχία των ξένων κλειδιών με τα primaryτων πινάκων που θέλουμε. Για παράδειγμα στους πίνακες clinickαι wardη σχέση είναι 1-N, καθώς μία κλινική μπορεί να περιλαμβάνει πολλούς θαλάμους. Αρχικά λοιπόν στον πίνακα **ward**θέτουμε το πεδίο **ClinicId**ως ευρετήριο.



Εικόνα 65: Ορισμός Ευρετηρίου

Κατόπιν πατώντας στην επιλογή εμφάνιση συσχετίσεων στη στήλη όριο μη διακριτού κλειδιού του πεδίου ClinicIdθέτουμε το primarykey**ClinicId** του πίνακα **clinic** της βάσης **hospital**ως ξένο κλειδί με ονομασία**fk**και πατάμε Αποθήκευση.

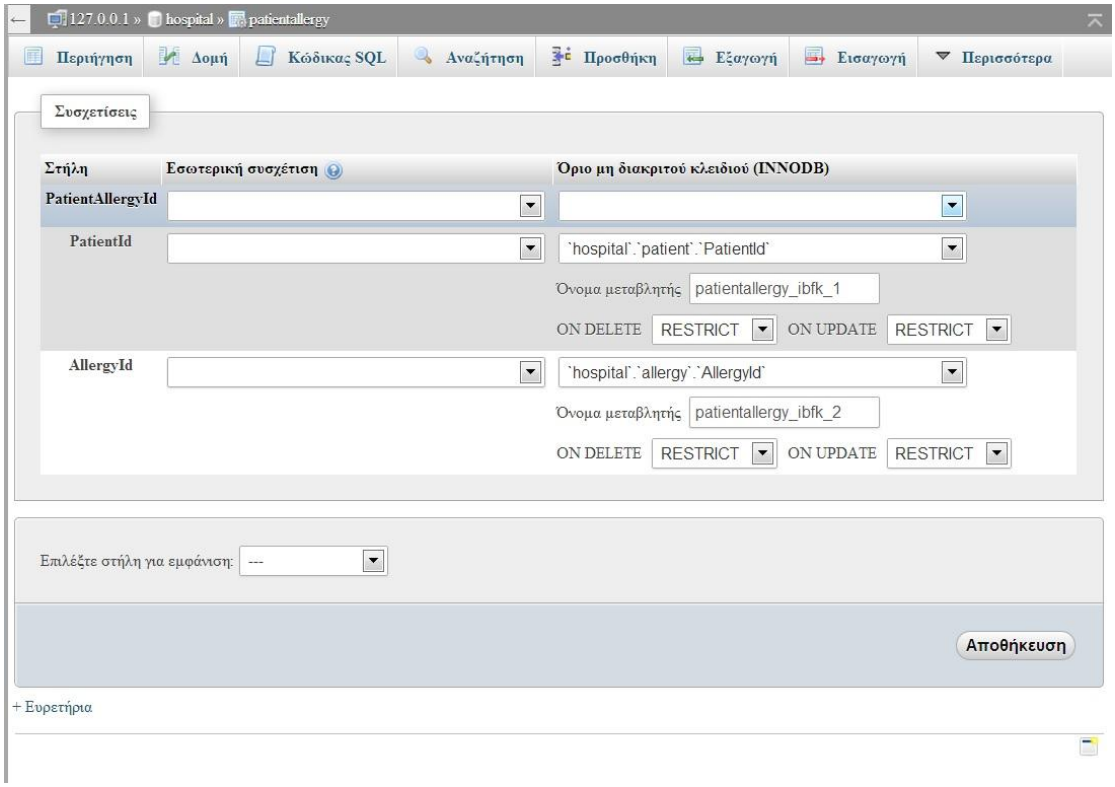


Εικόνα 66: Υλοποίηση συσχετίσεων 1-N

Παρόμοια διαδικασία ακουλουθούμε σε όλες τις σχέσεις πινάκων 1-N και 1-1.

Για τις **σχεσεις πολλά προς πολλά(N-M)**ακολουθήσαμε την παρακάτω διαδικασία:

Ορίσαμε νέο πίνακα 3 πεδίων, το πρώτο είναι primarykeyκαι τα άλλα 2 είναι τα ξένα κλειδιά των πινάκων που θελουμε να συσχετίσουμε. Τα 2 αυτά τελευταία πεδία τα ορίζουμεευρετήριο(index) και ακολουθούμε παρόμοια διαδικασία με παραπάνω. Για παράδειγμα ο πίνακας**patientallergy** είναι ενδιάμεσος πίνακας που συνδέει τους πίνακες patientκαι allergy. Μεταξύ αυτών των πινάκων υπάρχει σχέση πολλά προς πολλά(M-N), καθώς ένας ασθενής μπορεί να πάσχει από πολλές αλλεργίες και μία συγκεκριμένη αλλεργία μπορεί να παρουσιάζεται σε παραπάνω από έναν ασθενείς. Στον πίνακα λοιπόν **patientallergy**έχουμε ως primarykeyτο πεδίο PatientAllergyIdκαι θέτουμε ως ευρετήρια τα πεδία **PatientId**και **AllergyId**. Στη συνέχεια, στην επιλογή εμφάνιση συσχετίσεων αντιστοιχίζουμε τα 2 ευρετήρια που γημιουργήσαμε(**PatientId**και **AllergyId**) με τα primarykeyστων πινάκων που επιθυμούμε(**PatientId**του πίνακα**patient**και **AllergyId**του πίνακα**allergy**αντίστοιχα).



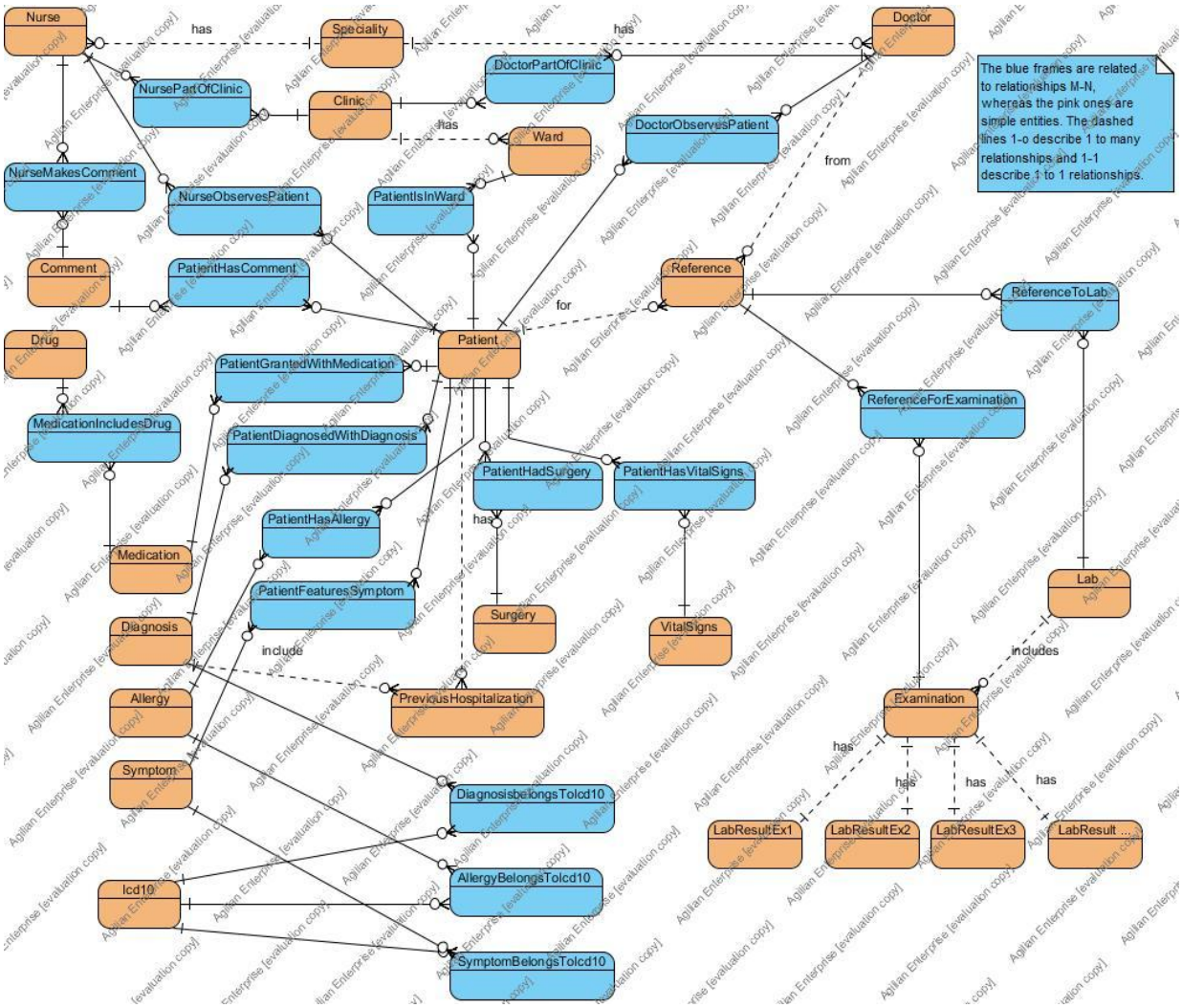
Εικόνα 67: Υλοποίηση συσχετίσεων M-N

Παρόμοια διαδικασία ακολουθήσαμε σε όλες τις σχέσεις πινάκων M-N.

Θα πρέπει ακόμη να αναφέρουμε οτι τα primarykeys όλων των πινάκων είναι **AUTO_INCREMENT**, πράγμα που σημαίνει πως με την εισαγωγή νέου στοιχείου αυξάνεται και η τιμή του αυτόματα. Τέλος, όσο αφορά τα constraints(fk) ισχύει στο **delete** και **update** η λειτουργια **restrict**, δηλαδή αν τυχόν ο χρήστης επιχειρήσει να διαγράψει πίνακα που περιλαμβάνει ξένα κλειδιά, το σύστημα δεν θα του το επιτρέψει.

3.3.3 E-R

Το διάγραμμα οντοτήτων-συσχετίσεων (entity-relationshipdiagram) είναι ένας τρόπος αφηρημένης και εννοιολογικής αναπαράστασης των δεδομένων. Ουσιαστικά επιτρέπει στο χρήστη να δει γραφικά τις σχέσεις που συνδέουν τα αντικείμενα της βάσης ώστε να έχει μια πλήρη εικόνα για τη λειτουργικότητα του συστήματος. Το e-r για την κεντρική βάση της εφαρμογής που υλοποιούμε παρατίθεται παρακάτωκαι υλοποιήθηκε με το εργαλείο Agilian 10.2-Visualparadigm. Στο συγκεκριμένο γράφημα οι οντότητες που παρουσιάζονται με μπλε χρώμα είναι οι πίνακες που υλοποιούν τις σχέσεις M-N.



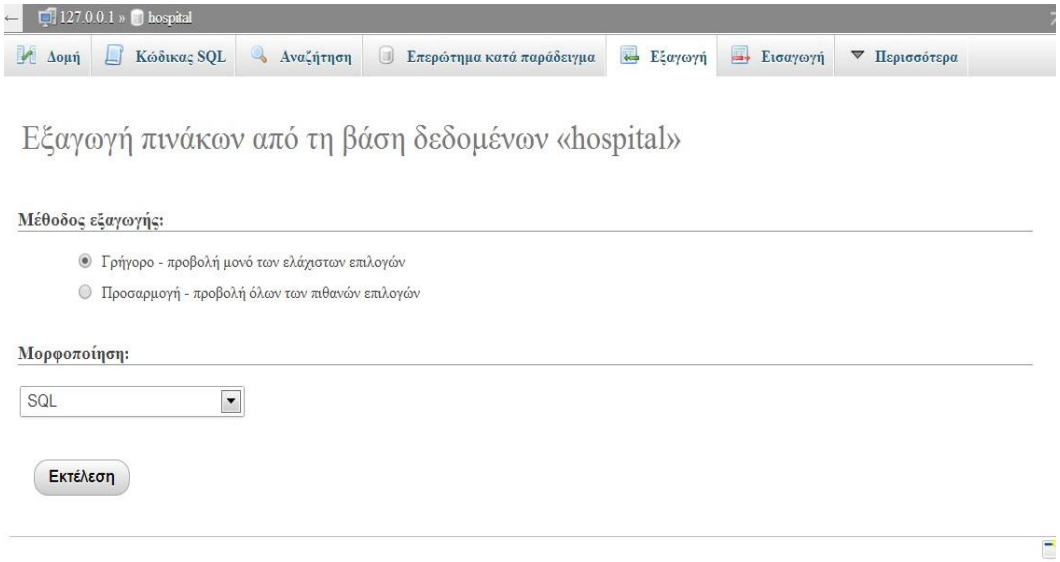
Εικόνα 68: E-R

3.4. Μεταφορά βάσης

Αφότου υλοποιήσαμε την κεντρική βάση στο phpMyAdminμε ονομασία hospital, ενδεχομένως υπάρχει ανάγκη μεταφοράς της σε κάποιο άλλο υπολογιστή. Για να επιτευχθεί η μεταφορά και εγκατάστασή της χωρίς να χαθούν τα δεδομένα της πρέπει να γίνει αρχικά εξαγωγή της (backup)και κατόπιν ειαγωγή της. Επίσης θεωρούμε ως προϋπόθεση οτι στον άλλο υπολογιστή είναι σωστά εγκατεστημένο το εργαλείο phpMyAdmin.

Τα βήματα για την εξαγωγή της βάσης είναι τα εξής (techrepublic):

- Επιτυχής είσοδος στο phpMyAdmin.
- Επιλέγουμε τη βάση που θέλουμε να κάνουμε export.
- Πατάμε την επιλογή εξαγωγή.
- Επιλέγουμε μέθοδο εξαγωγής(γρήγορη ή προσαρμοσμένη).
- Πατάμε το κουμπί εκτέλεση και στην επιφάνεια εργασίας δημιουργείται ένα αρχείο με επέκταση .sqlκαι το όνομα της βάσης μας. Στη συγκεκριμένη περίπτωση εφόσον η βάση ονομάζεται hospital, το backupθα έχει την ονομασία hospital.sql.



Εικόνα 69: Exportτης βάσης

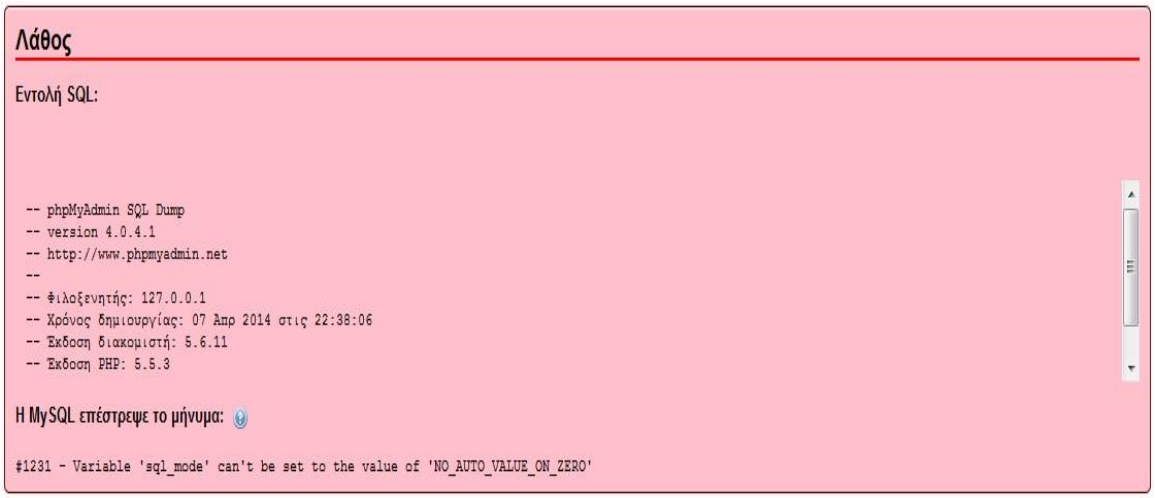
Τα βήματα για την εισαγωγή της βάσης είναι τα εξής:

- Επιτυχής είσοδος στο phpMyAdmin.
- Δημιουργία μιας νέας κενής βάσης.
- Πατάμε το κουμπί εισαγωγή.
- Επιλέγουμε το .sqlαρχείο προς εισαγωγή. Πρέπει να σημειωθεί οτι η σύνθεση της νέας κενής βάσης θα πρέπει να είναι ίδια με τη σύνθεση της βάσης που σκοπεύουμε να εισάγουμε, διαφορετικά η εισαγωγή δεν θα επιτευχθεί.
- Πατάμε το κουμπί εκτέλεση και η καινούρια βάση πλέον είναι ακριβώς ίδια σε ονόματα στοιχείων και περιεχόμενο με αυτή που κάναμε export. Να σημειωθεί πως σε περίπτωση που πεταχτει κάποιο error της μορφής που φαίνεται στην παρακάτω εικόνα, θα πρέπει να ανοίξουμε το αρχείο της βάσης .sqlκαι να σβήσουμε τις εξής γραμμές.

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";

SETtime_zone = "+00:00";
```

Κατόπιν επαναλαμβάνουμε τη διαδικασία και η εισαγωγή είναι πλέον επιτυχής.



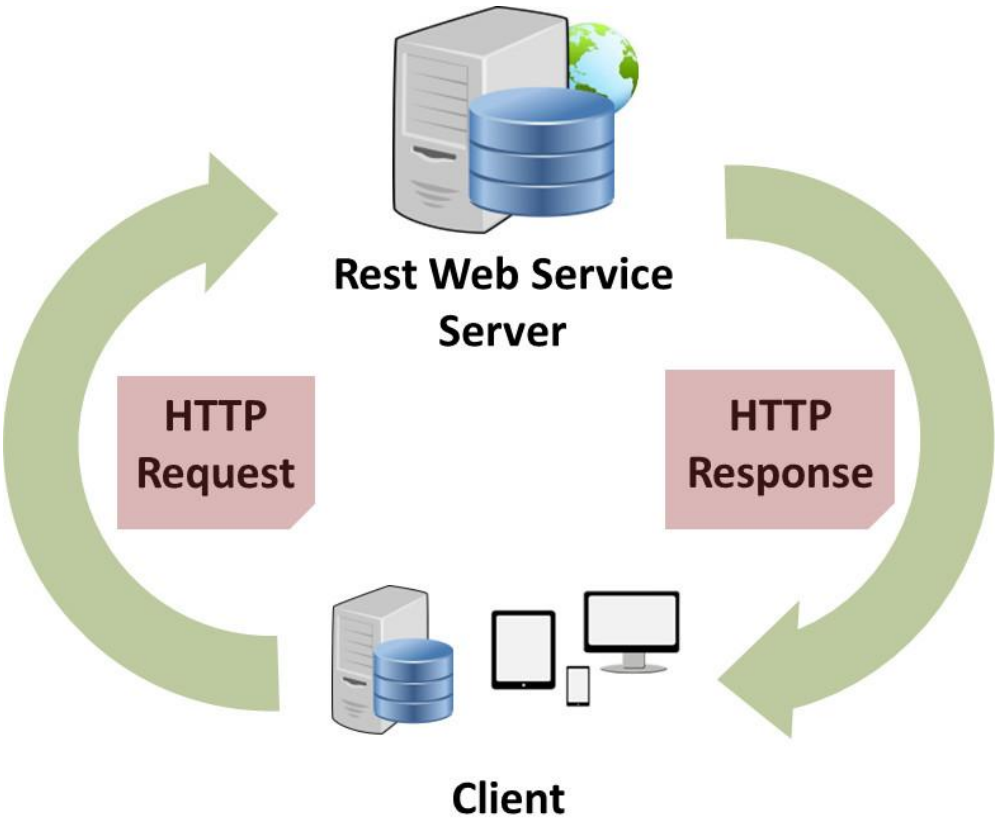
Εικόνα 70: Error στοimport

4. ΣΧΕΔΙΑΣΜΟΣ RESTFUL WEB SERVICES

4.1. Ορισμός

Τα web services είναι μια τεχνολογία που επιτρέπει στις εφαρμογές να επικοινωνούν μεταξύ τους ανεξαρτήτως πλατφόρμας και γλώσσας προγραμματισμού. Ένα web service είναι μια διεπαφή λογισμικού (software interface) που περιγράφει μια συλλογή από λειτουργίες οι οποίες μπορούν να προσεγγιστούν από το δίκτυο μέσω πρότυπων μηνυμάτων XML. Χρησιμοποιεί πρότυπα βασισμένα στη γλώσσα XML για να περιγράψει μία λειτουργία (operation) προς εκτέλεση και τα δεδομένα προς ανταλλαγή με κάποια άλλη εφαρμογή. Μια ομάδα από web services οι οποίες αλληλεπιδρούν μεταξύ τους καθορίζει μια εφαρμογή web services.Το βασικότερο χαρακτηριστικό των web services είναι ότι αποτελούν εφαρμογές που εκτελούνται στον server στον οποίο φιλοξενούνται και όχι στον υπολογιστή του client που τις καλεί. Ο πελάτης(client), αφού εντοπίσει τη Web Service, την καλεί, αυτή εκτελείται στον server (μπορεί κατά την εκτέλεσή της να αποκτά πρόσβαση σε δεδομένα που βρίσκονται και σε άλλους servers), και τελικά επιστρέφει στον πελάτη(client) τα ζητούμενα αποτελέσματα.

Τα RESTfulwebservices είναι μια κατηγορία webservices που έχουν δημιουργηθεί για να λειτουργούν καλύτερα στο web. Το REST είναι μια αρχιτεκτονική μέθοδος που καθορίζει περιορισμούς όπως η ομοιομορφία διεπαφής, η οποία αν εφαρμοστεί σε μια web υπηρεσία προσφέρει καλύτερη απόδοση, επεκτασιμότητα και ευελιξία τροποποίησης. Συν τοις άλλοις, στα RESTfulwebservicesτα δεδομένα και η λειτουργικότητα θεωρούνται πόροι και είναι προσβάσιμα μέσω των Uniform Resource Identifiers (URIs), με άλλα λόγια linksστο διαδίκτυο. Οι πόροι ενεργοποιούνται χρησιμοποιώντας ένα σύνολο απλών και σωστά καθορισμένων λειτουργιών. Η μορφή RESTπεριορίζει την αρχιτεκτονική στη μορφή πελάτη- εξυπηρετητή (client- server)και έχει σχεδιαστεί για να χρησιμοποιεί ένα πρωτόκολλο τύπου HTTP. Πελάτες και εξυπηρετητές ανταλάσσουν αναπαραστάσεις των πόρων χρησιμοποιώντας μια τυποποιημένη διεπαφή και πρωτόκολλο. Τα RESTfulwebservices δεν απαιτούν τίποτα παραπάνω από την αποστολή και «κατανάλωση» αιτήσεων HTTP. Καθώς δεν υπάρχει επίσημος τρόπος τεκμηρίωσης εισόδου και εξόδου των αιτήσεων, η επικύρωση θα πρέπει να υλοποιηθεί μέσω κώδικα.Η δημιουργία των RESTfulwebservice μπορεί να υλοποιηθεί σε οποιαδήποτε γλώσσα προγραμματισμού για web. Χαρακτηριστικές είναι η php, ηjava, η .netκαιη python.(oracle)



Εικόνα 71: Rest Web Service

4.2. Php

4.2.1 Συγγραφή Αρχείων

Η γλώσσα που επιλέξαμε να χρησιμοποιησουμε για την υλοποίηση του RESTfulWebService είναι η php. Ο λόγος για τον οποίο επιλέξαμε τη συγκεκριμένη γλώσσα είναι η σχετικά σύντομη εκμάθησή και συγγραφή της.

Το πρώτο αρχείο που δημιουργήσαμε είναι το **LoginDoctor.php**. Η διαδικασία που ακολουθήσαμε για τη συγγραφή του είναι η εξής:

- Δημιουργία νέου αρχείου .txt
- Γράψιμο κώδικα php
- Αποθήκευση του αρχείου ως .php
- Αποθήκευση του νέου τροποποιημένου αρχείου στο φάκελο htdocs και συγκεκριμένα στο directory «C:\xampp\htdocs»

Η λογική που ακολουθείται στον κώδικα είναι η εξής:

- Επιχείρηση σύνδεσης στον server
- Επιλογή της επιθυμητής βάσης δεδομένων του server, η οποία στην περίπτωσή μας είναι η hospital
- Ορισμός μεταβλητών που χρησιμοποιούνται. Πιο συγκεκριμένα στην περίπτωσή μας υπάρχει ένας πίνακας «doctors», στον οποίο αποθηκεύονται όλα τα στοιχεία του γιατρού που βρέθηκε, μία μεταβλητή «username»στην οποία αποθηκεύεται μέσω POST το usernameπου έχει δώσει ο χρήστης κατά τη διαδικασία εισαγωγής του στο σύστημα , μία μεταβλητή «password»στην οποία αποθηκεύεται μέσω POSTτο password που έχει δώσει επίσης ο χρήστης κατά τη διαδικασία εισαγωγής του και μια μεταβλητή «result»στην οποία αποθηκεύεται το αποτέλεσμα το sqlqueryπου επιθυμούμε. Στη συγκεκριμένη περίπτωση η μεταβλητή «result» θα περιέχει όλα τα στοιχεία του γιατρού με username και password που έχουν καθοριστεί από το χρήστη.
- Έλεγχος για το αποτέλεσμα επιστροφής του sqlquery. Σε περίπτωση που είναι μεγαλύτερο του 0, σημαίνει πως βρέθηκε γιατρός με τα στοιχεία που καθορίσαμε. Κατόπιν λοιπόν ακολουθεί parsetou αποτελέσματος στα αντίστοιχα πεδία του πίνακα doctors. Οπρογραμματιστής ορίζει τα ονόματα των πεδίων του πίνακα και τα αντιστοιχεί σε κάθε πεδίο της βάσης. Για

παράδειγμα με τη γραμμή *$doctors["DoctorId"] = $row["DoctorId"];*,ουσιαστικά στο πεδίο DoctorId του πίνακα doctors, βάζουμε τη την εγγραφή που υπάρχει στο πεδίο DoctorId της βάσης hospitalκαι πίνακα doctor. Κατά τον ίδιο τρόπο γεμίζουμε και τα υπόλοιπα πεδία και θέτουμε το πεδίο «success»της μεταβλητής responseπου υπάρχει defaultως 1.

- Σε περίπτωση που το αποτέλεσμα επιστροφής του sqlquery είναι μικρότερο του 0, σημαίνει πως δεν βρέθηκε γιατρός με τα στοιχεία που δόθηκαν από το χρήστη και συνεπώς θέτουμε το πεδίο «success» της μεταβλητής response που υπάρχει default ως 0.Επίσης επιστρέφεται μήνυμα στο χρήστη για «Μη έγγυρα στοιχεία».

Με ακριβώς ίδια λογική υλοποιείται και το αρχείο **LoginNurse.php**, με τη διαφορά πως αυτή τη φορά το sqlquery αναφέρεται στον πίνακα nurse της βάσηςκαι τα τελικά αποτελέσματα αποθηκεύονται στον πίνακα «nurses». Ανάλογα με το αν βρέθηκε νοσηλευτής με username και password τις τιμές που δόθηκαν από το χρήστη συμπληρώνεται και το πεδίο success με 0 για αποτυχία και 1 για επιτυχία και στην πρώτη περίπτωση επιστρέφεται επίσης μήνυμα «Μη έγκυρα στοιχεία».(wikihow)(dide.flo.sch)(androidhive)(coderanch)

LoginDoctor.php | LoginNurse.php

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["doctors"] = array();
 $username = $_POST['username'];
 $password = $_POST['password'];

 $result = mysql_query("select * FROM doctor WHERE (UsernameDoctor = '$username' && PasswordDoctor = '$password')")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $doctors["DoctorId"] = $row["DoctorId"];
        $doctors["LastName"] = $row["LnameDoctor"];
        $doctors["FirstName"] = $row["FnameDoctor"];
        $doctors["Username"] = $row["UsernameDoctor"];
        $doctors["Password"] = $row["PasswordDoctor"];
        $doctors["Speciality"] = $row["SpecialityDoctor"];

        // push a row into final response array
        array_push($response["doctors"], $doctors);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "Μη έγκυρα στοιχεία.";
    // echo no users JSON
    print json_encode($response);
}
```

Εικόνα 72: LoginDoctor.php

Στη συνέχεια παρατίθεται το αρχείο **AimatologikesExetaseis.php**, στο οποίο με παρόμοια με την παραπάνω λογική επιλέγονται τα πεδία ExamId και NameExam από τον πίνακα examination, όπου το LabId = 1(Αιματολογικό Εργαστήριο) αλφαβητικά. Στην ουσία επιλέγουμε αλφαβητικά όλες τις εξετάσεις που παρέχονται από το Αιματολογικό Εργαστήριο με αλφαβητική σειρά. Το συγκεκριμένο αρχείο χρησιμοποιείται όταν ο χρήστης τύπου «Γιατρός» επιλέξει να στείλει παραπεμπτικό για εξέταση σε κάποιο εργαστήριο. Σε αντίθεση με τα παραπάνω .php αρχεία, εδώ δεν υπάρχει κάποιο POST καθώς η αναζήτηση γίνεται βάσει του LabId, το οποίο είναι ήδη γνωστό(1 για Αιματολογικό εργαστήριο).

Παρομοίως υλοποιούνται τα αρχεία **AktinologikesExetaseis.php**, **BioxhmikesExetaseis.php**και**MikrobiologikesExetaseis.php**με κλειδιά στα sqlqueriesLabId = 4, LabId = 3 και LabId = 2 αντιστοίχως.

AimatologikesExetaseis.php | AvailableDiagnosis.php | AvailableMedicine.php | ClinicsRelate

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["Aimatologikes"] = array();


 $result = mysql_query("SELECT ExamId,NameExam
                        FROM examination where (
                        LabId = 1) ORDER BY NameExam ")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $Aimatologikes["ExamId"] = $row["ExamId"];
        $Aimatologikes["NameExam"] = $row["NameExam"];
        // push a row into final response array
        array_push($response["Aimatologikes"], $Aimatologikes);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;

    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 73: AimatologikesExetaseis.php

Στη συνέχεια παρατίθεται το αρχείο **AvailiableDiagnosis.php**, στο οποίο επιλέγονται όλα τα πεδία του πίνακα diagnosisμε αλφαβητική σειρά σύμφωνα με την ονομασία. Σε αυτό το αρχείο επίσης δεν υπάρχει POSTδεδομένων καθώς το sql query δεν εκτελείται βάσει κάποιου κλειδιού. Το συγεκριμένο αρχείο καλείται όταν κάποιος χρήστης τύπου «Γιατρός» επιλέξει να προσθέσει κάποια διάγνωση για τον ασθενή. Μέσω του συγκεκριμένου λοιπόν αρχείου φορτώνονται στο σύστημα όλες οι διαθέσιμες διαγνώσεις που είναι αποθηκευμένες στο serverώστε ο χρήστης να επιλέξει 1 ή περισσότερες προς προσθήκη.

Παρομοίως υλοποιείται και το αρχείο **AvailiableMedicine.php**κατά το οποίο φορτώνονται στο σύστημα όλα τα διαθέσιμα από την κεντρική βάση

(server)φάρμακα και ο γιατρός μπορεί να προσθέσει κάποιο-α στην φαρμακευτική αγωγή του ασθενή.

AvailableDiagnosis.php | AvailableMedicine.php | ClinicsRelatedToDoctor.php | ClinicsRelatedToN

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["ADiagnosis"] = array();

 $result = mysql_query("SELECT *
                        FROM  diagnosis
                        ORDER BY NameDiagnosis ")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $ADiagnosis["id"] = $row["DiagnosisId"];
        $ADiagnosis["name"] = $row["NameDiagnosis"];

        // push a row into final response array
        array_push($response["ADiagnosis"], $ADiagnosis);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "Δεν βρέθηκαν διαγνώσεις";
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 74: AvailiableDiagnosis.php

Στη συνέχεια παρουσιάζεται το αρχείο**ClinicsRelatedToDoctor.php**, στο οποίο επιστρέφονται όλες οι κλινικές που σχετίζονται με έναν συγκεκριμένο γιατρό, δίνοντας σε POSTως κλειδί το idτου γιατρού.

Με παρόμοιο τρόπο υλοποιείται το αρχείο **ClinicsRelatedToNurse.php**, το οποίο επιστρέφει τις κλινικές που σχετίζονται με έναν συγκεκριμένο νοσηλευτή.

ClinicsRelatedToDoctor.php | ClinicsRelatedToNurse.php | GetCommentId.php | GetPatientId.php | GetReferenceId.php | GetVitalSignId.php | LabResults1.php | LabR

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["doctorClinics"] = array();
 $DoctorId = $_POST['Id'];

//convert post data to int
 $DoctorIdInt = (int)$DoctorId;

 $result = mysql_query("select NameClinic FROM doctorclinic, clinic WHERE (DoctorId = $DoctorIdInt && doctorclinic.ClinicId = clinic.ClinicId)")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $doctorClinics["NameClinic"] = $row["NameClinic"];


        // push a row into final response array
        array_push($response["doctorClinics"], $doctorClinics);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "Μη έγκυρα στοιχεία.";
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 75: ClinicsRelatedToDoctor.php

Κατόπινπαρατίθεταιτοαρχείο**GetCommentId.php**, κατάτοοποίοεπιστρέφεταιτοπεδίοCommentId (primarykey) όλωντωνεγγραφώντουπίνακαcomment.

Μεπαρόμοιοτρόπουλοποιούνταιτααρχεία**GetPatientId.php, GetReferenceId.php**και**GetVitalSignId.php**, ώστεναεπιτραφείτο primary key τωνεγγραφώντωνπινάκων patient, reference και vitalsigns αντίστοιχα.

GetCommentId.php | GetPatientId.php | GetReferenceId.php | GetVitalSignId.php | LabResults1.p

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["CommId"] = array();

 $result = mysql_query("select CommentId
                        from comment")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $CommId["CommentId"] = $row["CommentId"];

        // push a row into final response array
        array_push($response["CommId"], $CommId);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 76: GetCommentId.php

Στησυνέχειαπαρατίθεταιτοαρχείο**LabResults1.php**, κατάτοοποίοδίνονταςσανκλειδίατο PatientId καιτο ReferenceId εκτελείταιτο sql query με inner join μεταξύτωνπινάκωντηςβάσης labresultex1, reference, referencelab, lab, referenceexamination και examination. Πρακτικά με αυτόν τον τρόπο επιστρέφονται τα αποτελέσματα της έτοιμης **εξέτασης αίματος** κάποιου συγκεκριμένου ασθενή.

Κατά τον ίδιο τρόπο λειτουργούμε και στα αρχεία **LabResults2.php, LabResults3.php**και**LabResults4.php,** για τα απότελέσματα της Ουρίας, Αμυλάσης Ούρων και Μαστογραφίας αντιστοίχως.

LabResults1.php | PatientAllergies.php | PatientComments.php | PatientDiagnosis.php | PatientHospitalization.php | PatientMedicine.php

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["LabResultExam1"] = array();
 $PatientId = $_POST['PatientId'];
 $ReferenceId = $_POST['ReferenceId'];


 $result = mysql_query("SELECT RGB, HGB, HT, WBC, NEUT, NEU, LYMPH, FysiologikhTimhRGB,
                            FysiologikhTimhHGB, FysiologikhTimhHT, FysiologikhTimhWBC,
                            FysiologikhTimhNEUT, FysiologikhTimhNEU, FysiologikhTimhLYMPH,
                            DateReady, lab.NameLab as a, examination.NameExam as b, reference.DateReference as c
                        FROM labresultex1
                        INNER JOIN reference
                        INNER JOIN referencelab
                        INNER JOIN lab
                        INNER JOIN referenceexamination
                        INNER JOIN examination ON
                        reference.ReferenceId = referencelab.ReferenceId &&
                        reference.ReferenceId = referenceexamination.ReferenceId &&
                        reference.ReferenceId = labresultex1.ReferenceId &&
                        reference.ReferenceId = $ReferenceId &&
                        reference.PatientId = $PatientId &&
                        referencelab.LabId = lab.LabId &&
                        referenceexamination.ExaminationId = examination.ExamId &&
                        examination.ExamId = labresultex1.ExaminationId")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $LabResultExam1["RGB"] = $row["RGB"];
        $LabResultExam1["HGB"] = $row["HGB"];
        $LabResultExam1["HT"] = $row["HT"];
        $LabResultExam1["WBC"] = $row["WBC"];
        $LabResultExam1["NEUT"] = $row["NEUT"];
        $LabResultExam1["NEU"] = $row["NEU"];
        $LabResultExam1["LYMPH"] = $row["LYMPH"];
        $LabResultExam1["FysiologikhTimhRGB"] = $row["FysiologikhTimhRGB"];
        $LabResultExam1["FysiologikhTimhHGB"] = $row["FysiologikhTimhHGB"];
        $LabResultExam1["FysiologikhTimhHT"] = $row["FysiologikhTimhHT"];
        $LabResultExam1["FysiologikhTimhWBC"] = $row["FysiologikhTimhWBC"];
        $LabResultExam1["FysiologikhTimhNEUT"] = $row["FysiologikhTimhNEUT"];
        $LabResultExam1["FysiologikhTimhNEU"] = $row["FysiologikhTimhNEU"];
        $LabResultExam1["FysiologikhTimhLYMPH"] = $row["FysiologikhTimhLYMPH"];
        $LabResultExam1["DateReady"] = $row["DateReady"];
        $LabResultExam1["NameLab"] = $row["a"];
        $LabResultExam1["NameExam"] = $row["b"];
        $LabResultExam1["DateReference"] = $row["c"];

        // push a row into final response array
        array_push($response["LabResultExam1"], $LabResultExam1);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 77: LabResults1.php

Παρατίθεται στη συνέχεια το αρχείο **PatientAllergies.php**, κατά το οποίο δίνοντας σαν κλειδί μέσω POSTτο PatientId, υλοποιείται innerjoinμεταξύ των πινάκων icd10, allergy, allergyicd10 και patientallergyώστε να επιστραφούν στο χρήστη(γιατρό-νοσηλευτή) τα πλήρη στοιχεία για τις αλλεργίες κάποιου συγκεκριμένου ασθενή. Τα στοιχεία αυτά είναι το όνομα, η περιγραφή και η κωδικοποίηση icd10 των αλλεργιών του ασθενή.

Με παρόμοια λογική(innerjoinόλων των απαραίτητων πινάκων και κλειδί το PatientId) είναι γραμμένα τα αρχεία **PatientComment.php, PatientDiagnosis.php, PatientHospitalization.php, PatientMedicine.php, PatientSymptoms.php, PatientVitalSigns.php**και **SurgeriesRelatedToPatient.php**, δίνοντάς μας πληροφορίες για την εικόνα του ασθενή.

PatientAllergies.php | PatientProf.php | PatientReferences.php | PatientsOfSpecificClinic.php

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["Allergies"] = array();
 $PatientId = $_POST['PatientId'];


 $result = mysql_query("select NameAllergy, DescriptionAllergy, EncodeDisease
                        from icd10 inner join allergy inner join allergyicd10 inner join patientallergy on
                        (allergyicd10.AllergyId = allergy.AllergyId &&
                            allergyicd10.Icd10Id = icd10.Icd10Id &&
                            patientallergy.AllergyId = allergy.AllergyId &&
                            patientallergy.PatientId = $PatientId) ")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $Allergies["NameAllergy"] = $row["NameAllergy"];
        $Allergies["DescriptionAllergy"] = $row["DescriptionAllergy"];
        $Allergies["Encode"] = $row["EncodeDisease"];

        // push a row into final response array
        array_push($response["Allergies"], $Allergies);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "Ο ασθενής δεν έχει αλλεργίες.";
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 78: PatientAllergies.php

Κατόπιν παρατίθεται το αρχείο **PatientProf.php**, στο οποίο δίνοντας σαν key μέσω POST το PatientId, επιστρέφονται όλα τα στοιχεία του προφίλ του ασθενή από τον πίνακα patienttης βάσης.

PatientProf.php | PatientReferences.php | PatientsOfSpecificClinic.php | InsertComment.php

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["PatientProfile"] = array();
 $PatientId = $_POST['PatientId'];


 $result = mysql_query("SELECT * FROM patient WHERE PatientId = $PatientId")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $PatientProfile["Fname"] = $row["Fname"];
        $PatientProfile["Lname"] = $row["Lname"];
        $PatientProfile["FatherFname"] = $row["FatherFname"];
        $PatientProfile["MotherFname"] = $row["MotherFname"];
        $PatientProfile["DateBirth"] = $row["DateBirth"];
        $PatientProfile["Address"] = $row["Address"];
        $PatientProfile["City"] = $row["City"];
        $PatientProfile["Country"] = $row["Country"];
        $PatientProfile["PostCode"] = $row["PostCode"];
        $PatientProfile["Job"] = $row["Job"];
        $PatientProfile["Sex"] = $row["Sex"];
        $PatientProfile["PhoneNumber"] = $row["PhoneNumber"];
        $PatientProfile["Afm"] = $row["Afm"];
        $PatientProfile["AMKA"] = $row["AMKA"];
        $PatientProfile["MaritalStatus"] = $row["MaritalStatus"];
        $PatientProfile["DateHospitalization"] = $row["DateHospitalization"];
        $PatientProfile["DateOffHospitalization"] = $row["DateOffHospitalization"];
        $PatientProfile["Url"] = $row["Url"];
        $PatientProfile["Email"] = $row["Email"];
        $PatientProfile["Fund"] = $row["Fund"];
        $PatientProfile["Smoking"] = $row["Smoking"];
        $PatientProfile["Alcohol"] = $row["Alcohol"];
        $PatientProfile["Diet"] = $row["Diet"];



        // push a row into final response array
        array_push($response["PatientProfile"], $PatientProfile);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "Μη έγκυρα στοιχεία.";
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 79: PatientProf.php

Παρατίθεται παρακάτω το αρχείο **PatientReferences.php**, στο οποίο δίνοντας σαν key μέσω POST το PatientId, επιστρέφονται όλα τα referencesπου σχετίζονται με το συγκεκριμένο ασθενή και έχουν τιμή status = 1, πράγμα που σημαίνει πως τα αποτελέσματά τους είναι έτοιμα.

C:\xampp\htdocs\PatientReferences.php - Notepad++ [Administrator]

Αρχείο Επεξεργασία Εύρεση Προβολή Κωδικοποίηση Γλώσσα Ρυθμίσεις Μακροεντολή Εκτέλε

PatientReferences.php | PatientsOfSpecificClinic.php | InsertComment.php

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["References"] = array();
 $PatientId = $_POST['PatientId'];


 $result = mysql_query("SELECT ReferenceId
                        FROM reference where (
                        PatientId=$PatientId  && status=1)")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $References["ReferenceId"] = $row["ReferenceId"];
        // push a row into final response array
        array_push($response["References"], $References);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no products found
    $response["success"] = 0;
    $response["message"] = "Ο ασθενής δεν έχει έτοιμες εξετάσεις.";
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 80: PatientReferences.php

Παρατίθεται εν συνεχεία το αρχείο **PatientsOfSpecificClinic.php**, στο οποίο αφότου ο χρήστης (νοσηλευτής/γιατρός) επιλέξει μία από τις κλινικές με τις οποίες σχετίζεται, του επιστρέφονται όλοι οι ασθενεις που συνδέονται με τη συγκεκριμένη κλινική, δηλαδή όλοι οι ασθενείς όλων των θαλάμων που ανήκουν στην επιλεγμένη κλινική. Δίνεται σαν key μέσω POST η μεταβλητή ChosenClinic και κατόπιν βάσει αυτού επιστρέφονται τα πεδία PatientId, Fname&Lname από τους πίνακες patientward&patient της βάσης.

PatientsOfSpecificClinic.php | InsertComment.php

```php
<?php

// Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

// Select database
    mysql_select_db("hospital") or die(mysql_error());

// array for JSON response
 $response["patientsOfClinic"] = array();
 $ChosenClinic = $_POST['ChosenClinic'];

 $result = mysql_query("SELECT patient.PatientId, Fname, Lname
                        FROM patientward, patient
                        WHERE WardId IN (
                            SELECT WardId
                            FROM ward
                            WHERE ClinicId = (
                                SELECT ClinicId
                                FROM clinic
                                WHERE NameClinic =  '$ChosenClinic' )
                        ) && patientward.PatientId = patient.PatientId")
            or die(mysql_error());

// check for empty result
if (mysql_num_rows($result) > 0) {
    while ($row = mysql_fetch_array($result)) {
        $patientsOfClinic["PatientId"] = $row["PatientId"];
        $patientsOfClinic["Fname"] = $row["Fname"];
        $patientsOfClinic["Lname"] = $row["Lname"];

        // push a row into final response array
        array_push($response["patientsOfClinic"], $patientsOfClinic);
    }

    // success
    $response["success"] = 1;
    // echoing JSON response
    print json_encode($response);

} else {
    // no patients found
    $response["success"] = 0;
    $response["message"] = "Δεν υπάρχουν ασθενείς σε αυτή την Κλινική.";
    // echo no users JSON
    print json_encode($response);
}
?>
```

Εικόνα 81: PatientsOfSpecificClinic.php

Ο τελευταίος τύπος αρχείου php που δημιουργήσαμε είναι αυτός που υλοποιεί τις εισαγωγές σε πίνακες της κεντρικής βάσης hospital με τιμές που δίνονται από τους χρήστες της εφαρμογής. Παραθέτουμε το παράδειγμα του αρχείου **InsertComment.php**, στο οποίο στέλνονται μέσω POST πληροφορίες σχετικές με την ημερομηνία που έγινε το σχόλιο από το νοσηλευτή και την ίδια πληροφορία του σχολίου. Κατόπιν οι συγκεκριμένες πληροφορίες εισάγονται σαν μια νέα γραμμή στον πίνακα comment. Αν η default μεταβλητή result επιστρέψει true, σημαίνει πως η εισαγωγή ήταν επιτυχής, διαφορετικά επιστρέφεται μήνυμα στο χρήστη «Oops! An error occurred.»

Με παρόμοιο τρόπο είναι γραμμένα και τα υπόλοιπα αρχεία που εκτελούν insert στη βάση hospital, **InsertDiagnosis.php, InsertLabResultEx1.php, InsertLabResultEx2.php, InsertLabResultEx3.php, InsertLabResultEx4.php, InsertMedicine.php, InsertPatient.php, InsertPatientComment.php, InsertPatientVitalSign.php, InsertPatientWard.php, InsertReference.php, InsertReferenceLab.php, InsertRefExam.phpInsertVitalSigns.phpκαιInsertNurseComment.php**

InsertComment.php

```php
<?php

/*
 * Following code will create a new product row
 * All product details are read from HTTP Post Request
 */


    // Connect to database server
    mysql_connect("localhost", "root", "") or die (mysql_error());

    // Select database
    mysql_select_db("hospital") or die(mysql_error());




    // check for required fields

    $DateComment = $_POST['DateComment'];
    $About = $_POST['About'];

    // mysql inserting a new row
    $result = mysql_query("INSERT INTO comment(DateComment, About) VALUES('$DateComment',  '$About')");

    // check if row inserted or not
    if ($result) {
        // successfully inserted into database
        $response["success"] = 1;
        $response["message"] = "Comment successfully added.";

        // echoing JSON response
        echo json_encode($response);
    } else {
        // failed to insert row
        $response["success"] = 0;
        $response["message"] = "Oops! An error occurred.";

        // echoing JSON response
        echo json_encode($response);
    }

?>
```

Εικόνα 82: InsertComment.php

4.2.2. ΚλήσηΑρχείωνμέσωκώδικα Java

Για να μπορέσουμε να συνδέσουμε την εφαρμογή με τον server, θα πρέπει από τον κώδικα Java που υλοποιούμε την Android εφαρμογή, να επιχειρήσουμε σύνδεση στο server στέλνοντας ένα αίτημα τύπου HTTP.Το αίτημα αυτό στον κώδικα αναπαρίσταται με μία μεταβλητή τύπου **HttpPost**την οποία θέτουμε ίση με την τιμή *newHttpPost("http://10.0.2.2:8888/xxx.php"),* όπου xxx.phpτο εκάστοτε αρχείο .phpπου χρειαζόμαστε από αυτά που περιγράφηκαν αναλυτικά παραπάνω.Επίσης σε περίπτωση που χρειάζεται να γίνει κάποια αναζήτηση σε πίνακα-ες της βάσης μας ανάλογα με κάποιο key, θα πρέπει να «στείλουμε» με κάποιο τρόπο αυτό το κλειδί. Η συγκεκριμένη διαδικασία υλοποιείται με τη βοήθεια δεδομένων τύπου *NameValuePair<Key, Value>,* όπου στο πεδίοKey εισάγεται η ονομασία που επιθυμούμε να έχει το κλειδί μας και στο πεδίο Value η τιμή του. Το αποτέλεσμα που θα λάβουμε από την αίτηση αυτή στο serverεισάγεται μη parsαρισμένο στη μεταβλητή response τύπου **HttpResponse**. Θα πρέπει να σημειωθεί πως για να είναι επιτυχές το αίτημα προς το serverθα πρέπει να είναι ενεργοποιημένες οι επιλογές Apache&MySQLστο εργαλείο Xampp. Επίσης πρέπει να αναφέρουμε πως όλο αυτό το κομμάτι γίνεται στο Backgroundμέσω AsyncTaskκαι σε περίπτωση αποτυχίας σύνδεσης πετιέται κατάλληλο exception. Το AsyncTask είναι μία έτοιμη κλάση που προσφέρει το androidγια σωστή διαχείριση των διαφόρων νημάτων του προγράμματος. Η κλάση αυτή χρησιμοποιείται ευρέως για αιτήματα προς το server, διότι εξ αιτίας του μονονηματικού μοντέλου που ακολουθούν οι εφαρμογές Android, μέχρι να ολοκληρωθεί ο χρόνος απόκρισης, η οθόνη του χρήστη είναι κενή. Έτσι θα πρέπει να αποφύγουμε την εκτέλεση μεγάλων λειτουργιών στο βασικό νήμα και θα πρέπει για το λόγο αυτό λειτουργίες όπως η πρόσβαση στο διαδίκτυο, που είναι κάπως χρονοβόρες, να τις υλοποιήσουμε στο παρασκήνιο (Background), μέσω AsyncTask.

Για να γίνει πιο αντιληπτή η συγκεκριμένη βασική λειτουργία παρουσιάζουμε ενδεικτικά ένα απόσπασμα από τον κώδικα Javaπου υλοποιήσαμε. Στο συγκεκριμένο απόσμασμα ανάλογα με το statustou χρήστη(γιατρός/νοσηλευτής) γίνεται η αντίστοιχη αίτηση στο server,κατά την οποία υλοποιείται η επικύρωση των στοιχείων (username&password)που έδωσε ο χρήστης. Στην ουσία ο χρήστης όταν επιχειρεί να κάνει loginστο σύστημα, εισάγει από το πληκτρολόγιο το status(γιατρός/νοσηλευτής), username&password. Ανάλογα λοιπόν με το statuskαλείται το αντίστοιχο .phpαρχείο και σε περίπτωση που μιλάμε για γιατρό γίνεται αναζήτηση στον πίνακα των γιατρών στη βάση του serverμε τα στοιχεία που δόθηκαν, αλλιώς γίνεται αντίστοιχη αναζήτηση στον πίνακα των νοσηλευτών. Με

παρόμοιο τρόπο υλοποιούνται όλες οι αιτήσεις προς το serverγια όλα τα αρχεία .phpπου δημιουργήσαμε.(AndroidGreece)

```
protected String doInBackground(String... message) {

    HttpPost request = new HttpPost();
    HttpResponse response = null;
    String result = "";

    // Try to connect using Apache HttpClient Library
    try {
        HttpClient httpclient = new DefaultHttpClient();
        if(status.equalsIgnoreCase("Γιατρός"))
            request = new HttpPost("http://10.0.2.2:8888/LoginDoctor.php");
        else
            request = new HttpPost("http://10.0.2.2:8888/LoginNurse.php");

        // Add keys for searching
        List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(2);
        nameValuePairs.add(new BasicNameValuePair("username",username));
        nameValuePairs.add(new BasicNameValuePair("password",password));
        request.setEntity(new UrlEncodedFormEntity(nameValuePairs));

        response = httpclient.execute(request);
        }

    catch (Exception e) {result = "aaaaaaaa";}

    // response code
    try {
        BufferedReader rd = new BufferedReader(new InputStreamReader(
                response.getEntity().getContent()));
        String line = "";
        while ((line = rd.readLine()) != null) {
            result = result + line ;
        }
    } catch (Exception e) {result = "bbbbbbbbbbbbbbbb";}
    return result;
}
```

Εικόνα 83: Αίτημα και απάντηση στον και από τον server

4.3. Κωδικοποίηση JSON

Το εργαλείο **JSON**(JavaScriptObjectNotation) είναι ένα ελαφρύ πρότυπο ανταλλαγής δεδομένων. Είναι εύκολο για τους ανθρώπους να το διαβάσουν και γράψουν. Είναι εύκολο για τις μηχανές να το αναλύσουν (parse) και να το παράγουν (generate). Είναι βασισμένο πάνω σε ένα υποσύνολο της γλώσσαςπρογραμματισμού JavaScript, StandardECMA-262 Έκδοση 3η - Δεκέμβριος 1999. Το JSON είναι ένα πρότυπο κειμένου το οποίο είναι τελείως ανεξάρτητο από γλώσσες προγραμματισμού αλλά χρησιμοποιεί πρακτικές (conventions) οι οποίες είναι γνωστές στους προγραμματιστές της οικογένειας προγραμματισμού C, συμπεριλαμβανομένων των C, C++, C#, Java, JavaScript, Perl, Python, και πολλών άλλων. Αυτές οι ιδιότητες κάνουν το JSON μια ιδανική γλώσσα προγραμματισμού ανταλλαγής δεδομένων.(json)

Το JSON είναι χτισμένο σε δύο δομές:

- Μια συλλογή από ζευγάρια ονομάτων/τιμών. Σε διάφορες γλώσσες προγραμματισμού, αυτό θεωρείται ως ένα Object, καταχώριση, δομή, λεξικό, πίνακα hash (hashtable), λίστα κλειδιών, ή associative πίνακα.
- Μία ταξινομημένη λίστα τιμών. Στις περισσότερες γλώσσες προγραμματισμού, αυτό θεωρείται ως ένας πίνακας (array), διάνυσμα, λίστα, ή ακολουθία.

Αυτά είναι τα universaldatastructures. Ουσιαστικά όλες οι μοντέρνες γλώσσες προγραμματισμού τα υποστηρίζουν με τον έναν ή τον άλλον τρόπο. Είναι ένα πρότυπο δεδομένων το οποίο είναι εύκολα μεταβαλλόμενο με γλώσσες προγραμματισμού οι οποίες επίσης είναι βασισμένες σε αυτές τις δομές.

Όσο αφορά την εφαρμογή που δημιουργούμε μπορούμε να παρατηρήσουμε πως στα αρχεία .phpπου χρησιμοποιούμε η επιστροφή των δεδομένων είναι σε μορφή πινάκων. Για να μπορέσουμε να χωρίσουμε(parse) αυτά τα δεδομένα, κάνουμε χρήση του εργαλείου JSONστον κώδικα Java. Ουσιαστικά δημιουργούμε ένα νέο αντικείμενο **JSONObject** με όνομα jsonστο οποίο βάζουμε το αποτέλεσμα που επιστρέφει ως απάντηση ο server. Στις περισσότερες περιπτώσεις το αποτέλεσμα αυτό περιλαμβάνει τη μεταβλητή «**message**», τη μεταβλητή«**success**»και τον **πίνακαποτελεσμάτων** που σε κάθε αρχείο .phpείναι διαφορετικό. Η γενική ιδέα είναι αρχικά να θέσουμε έναν **integer**ίσο μεjson.getInt("success"), έναν **JSONArray**ίσο με json.getJSONArray("xx"),όπου xxo πίνακας που επιστρέφει σαν απάντηση το .phpαρχείο,και ένα **String**ίσο με json.getString("message").

Αρχικά ελέγχουμε τον **integer** και σε περίπτωση που είναι ΄1΄, σημαίνει πως έχουμε επιτυχία και υπάρχουν δεδομένα που επέστρεψε η βάση. Σε αυτή την περίπτωση λοιπόν διατρέχουμε με forlooptον πίνακα τύπου **JSONArray**που ορίσαμε παραπάνω και θέτουμε για κάθε στοιχείο του, μια νέα μεταβλητή τύπου

JSONObjectστην οποία βάζουμε κάθε στοιχείο. Στη συνέχεια κάνουμε parseτα στοιχεία που περιλαμβάνει κάθε τέτοιο υποστοιχείο με getκαι τα εισάγουμε σε νέες μετάβλητέςή λίστες, ώστε να τα διαχειριστούμε με τον τρόπο που επιθυμούμε στο μέλλον.Στην περίπτωση που ο integereίναι ΄0΄, εμφανίζουμε σε μορφή toastτο μήνυμα, δηλαδή τη μεταβλητή **String**.Η παραπάνω διαδικασία υλοποιείται στη μέθοδοonPostExecuteπάλι χρησιμοποιώντας την κλάση AsyncTask που αναφέραμε στην προηγούμενη ενότητα.

Ενδεικτικά παρουσιάζεται παρακάτω απόσπασμα κώδικα Java, στο οποίο έχει προηγηθεί η κλήση του αρχείου PatientsOfClinic.php. Ηαπάντηση του serverστην αίτηση αυτή είναι ουσιαστικά ένας πίνακας PatientsOfClinic, οοποίος περιλαμβάνει όλους τους ασθενείς της κλινικής που επέλεξε ο χρήστης (γιατρός/νοσηλευτής). Για κάθε ασθενή περιλαμβάνονται τα πεδία Fname, Lnameκαι PatientId. Δημιουργούμε λοιπόν 3 συνδεδεμένες λίστες PatientId, PatientFnameκαιPatientLname κάθε μία από τις οποίες θα περιέχει όλα τα PatientId, Fname και Lname αντίστοιχα όλων των ασθενών που σχετίζονται με την επιλεγμένη κλινική.Παρόμοια διαδικασία ακολουθείται σε όλες τις περιπτώσεις που έχουμε αίτηση στο serverκαι λαμβάνουμε απάντησή του.

```
protected void onPostExecute(String result) {

        try{
            JSONObject json = new JSONObject(result);
            int success = json.getInt("success");
            PatientId = new LinkedList<Integer>();
            PatientFname = new LinkedList<String>();
            PatientLname = new LinkedList<String>();

            // if there are clinics
            if(success == 1){
                JSONArray patientsOfClinic = json.getJSONArray("patientsOfClinic");
                for(int i=0;i<patientsOfClinic.length();i++){
                    JSONObject json_data = patientsOfClinic.getJSONObject(i);
                    PatientId.add(json_data.getInt("PatientId"));
                    PatientFname.add(json_data.getString("Fname"));
                    PatientLname.add(json_data.getString("Lname"));
                }
            }
            //if there are not patients in the specific clinic
            else{
                String message = json.getString("message");
                //show toast for not correct data
                Context context = getApplicationContext();
                int duration = Toast.LENGTH_LONG;
                Toast toast = Toast.makeText(context, message, duration);
                toast.show();
                //initialize arrays in order to be sent at List View as empty

            }
        }catch(JSONException e){
            String message = "error!!!!!!";
            //show toast for Error
            Context context = getApplicationContext();
            int duration = Toast.LENGTH_LONG;
            Toast toast = Toast.makeText(context, message, duration);
            toast.show();
        }

    }
```

Εικόνα 84: Json parsing

5. ΤΟΠΙΚΗΒΑΣΗ SQLITE

5.1. Γενικές Πληροφορίες

Το SQLite είναι μία βάση δεδομένων ανοιχτού κώδικα που υποστηρίζει σχεσιακές δυνατότητες της βάσης όπως η σύνταξη SQL, συναλλαγές και έτοιμα statements. Η SQLite απαιτεί περιορισμένη μνήμη κατά το χρόνο εκτέλεσης(περίπου 250 KByte), πράγμα που την καθιστά ικανή να ενσωματωθεί σε άλλους χρόνους εκτέλεσης.

Επίσης υποστηρίζει τους τύπους δεδομένων TEXT(σαν το String στη Java) και INTEGER(σαν το long στη Java) και REAL(σαν το double στη Java). Όλοι οι υπόλοιποι τύποι θα πρέπει να μετατραπούν σε έναν από τους παραπάνω πριν σωθούν στην τοπική βάση.

Η SQLite είναι ενσωματωμένη σε κάθε συσκευή που τρέχει λειτουργικό Android. Η χρήση τοπικής SQLite στο Android δεν απαιτεί κάποια διαδικασία εγκατάστασης ή κάποια άδεια διαχείρισης. Η πρόσβαση σε μια SQLite περιλαμβάνει πρόσβαση στο σύστημα αρχείων, διαδικασία που μπορεί να είναι χρονοβόρα. Για το λόγο αυτό συνιστάται να εκτελούνται οι λειτουργίες της βάσης με ασύγχρονο τρόπο. Εαν η Android εφαρμογή μας δημιουργεί μια βάση δεδομένων SQLite, θα πρέπει να γνωρίζουμε πως είναι αποθηκευμένη bydefault στο directoryDATA/data/APP_NAME/databases/FILENAME.(vogella) (hmkcode)

Κάποια από τα βασικά πλεονεκτήματα της SQLite είναι:

- **Ταχύτητα**
    - Σε πολλές περιπτώσεις είναι το λιγότερο 2-3 φορές πιο γρήγορη από MySQL/PostgreSQL.
    - Γρήγορο php interface
    - Δεν χρησιμοποιούνται sockets και/ή TCP/IP
- **Λειτουργικότητα**
    - Sub-selects, Triggers, Transactions, Views
    - Παραπάνω από 2TB χώρου αποθήκευσης δεδομένων
    - Μικρό memory footprint
    - Αυτοτέλεια, καθώς δεν υπάρχουν εξωτερικές εξαρτήσεις.
    - Ατομική δέσμευση και επαναφορά της προστασίας της ακεραιότητας των δεδομένων
    - Εύκολα μετακινήσημη
- **Ασφάλεια**
    - Κάθε χρήστης έχει τη δική του εντελώς ανεξάρτητη βάση δεδομένων. Κάθε εφαρμογή αποθηκεύει τοπικά τα δικά της στοιχεία και η βάση δεδομένων SQLite χρησιμοποείται αποκλειστικά από τη συγκεκριμένη εφαρμογή, πράγμα που εξασφαλίζει την απουσία σύγχησης δεδομένων μεταξύ των διαφόρων δεδομένων που υπάρχουν στη συσκευή και επομένως ασφάλεια δεδομένων.

Στην εφαρμογή που υλοποιήσαμε η γενική ιδέα για τη χρήση της τοπικής βάσης είναι η χρήση της για προσωρινή αποθήκευση δεδομένων που δίνονται από το χρήστη. Σε περίπτωση που εκτελείται αίτημα στο server για γέμισμα κάποιων πεδίων ορισμένων πινάκων της κεντρικής βάσης hospitalκαι η σύνδεση είναι ανεύφικτη, θα πρέπει τα δεδομένα που δόθηκαν να αποθηκευτούν κάπου. Πέρασε αρχικά η σκέψη να αποθηκεύονται σε ένα ή περισσότερα αρχεία, όμως το σενάριο αυτό ναυάγησε καθώς τα αρχεία είναι εξωτερικός τύπος δεδομένων και δύσκολα υποστηριζόταν από την έκδοση Android 4.3 στην οποία είναι γραμμένη η

androideφαρμογή μας. Με κάποιες αλλαγές στο αρχείο AndroidManifest.xmlπαρόλα αυτά θα μπορούσε να γίνει χρήση εξωτερικών στοιχείων, όμως και πάλι δεν θα εξασφαλιζόταν 100% η ασφάλεια των δεδομένων. Για τους λόγους αυτούς λοιπόν επιλέξαμε να χρησιμοποιήσουμε την SQLitetοπική βάση που διαθέτει το Android . Για τη χρήση της δεν χρειάστηκε κάποια περίπλοκη διαδικασία και με αυτό τον τρόπο εξασφαλίζουμε με σιγουριά το απόρρητο και την ασφάλεια των δεδομένων.Δημιουργήσαμε στη Javaμια νέα κλάση με όνομα **MySQLiteHelper,** η οποία κάνει extend την έτοιμη κλάση**SQLiteOpenHelper**και μέσω αυτής περιγράφουμε τα στοιχεία της τοπικής βάσης μας και δημιουργούμε συναρτήσεις για την δημιουργία πινάκων, εισαγωγή δεδομένων στους πίνακες, ανάγνωση δεδομένων από τους πίνακες και διαγραφή δεδομένων από τους πίνακες.

5.2. Πίνακες

5.2.1. Δημιουργία πινάκων

Η δημιουργία των πινάκων της τοπικής SQLite βάσης υλοποιείται με τη μέθοδο onCreate, η οποία παρατίθεται παρακάτω. Ουσιαστικά κάνουμε Create των πινάκων που επιθυμούμε και κατόπιν τους εισάγουμε ομαδικά στη βάση εκτελώντας τα ερωτήματα SqlCreate που δημιουργήσαμε.

```
1   @Override
2     public void onCreate(SQLiteDatabase db) {
3         // SQL statement to create patientdiagnosis table
4         String CREATE_patientdiagnosis_TABLE = "CREATE TABLE if not exists patientdiagnosis ( " +
5                 "PatientDiagnosisId INTEGER PRIMARY KEY AUTOINCREMENT, " +
6                 "PatientId INTEGER, " +
7                 "DiagnosisId INTEGER, " +
8                 "DateDiagnosis TEXT )";
9
10        String CREATE_diagnosis_TABLE = "CREATE TABLE if not exists diagnosis ( " +
11                "DiagnosisId INTEGER PRIMARY KEY AUTOINCREMENT, " +
12                "NameDiagnosis TEXT )";
13
14        String CREATE_medication_TABLE = "CREATE TABLE if not exists medicine ( " +
15                "MedicationId INTEGER PRIMARY KEY AUTOINCREMENT, " +
16                "NameMedication TEXT )";
17
18        String CREATE_patientmedication_TABLE = "CREATE TABLE if not exists patientmedicine ( " +
19                "PatientMedicationId INTEGER PRIMARY KEY AUTOINCREMENT, " +
20                "PatientId INTEGER, " +
21                "MedicationId INTEGER, " +
22                "Dosage TEXT, " +
23                "StartMedication TEXT, " +
24                "StopMedication TEXT )";
25
26        String CREATE_comment_TABLE = "CREATE TABLE if not exists comment ( " +
27                "CommentId INTEGER PRIMARY KEY AUTOINCREMENT, " +
28                "DateComment TEXT, " +
29                "About TEXT )";
30
31        String CREATE_vitalsigns_TABLE = "CREATE TABLE if not exists vitalsigns ( " +
32                "MeasurementId INTEGER PRIMARY KEY AUTOINCREMENT, " +
33                "DateMeasurement TEXT, " +
34                "Temperature TEXT , " +
35                "Pressure TEXT, " +
36                "Breathing TEXT, " +
37                "Pulses TEXT, " +
38                "SpO2 TEXT, " +
39                "Kfp TEXT, " +
40                "Emptying TEXT, " +
41                "Diabetes TEXT)";
```

```
43          String CREATE_patient_TABLE = "CREATE TABLE if not exists patient ( " +
44                  "PatientId INTEGER PRIMARY KEY AUTOINCREMENT, " +
45                  "Fname TEXT, " +
46                  "Lname TEXT , " +
47                  "FatherFName TEXT, " +
48                  "MotherFName TEXT, " +
49                  "DateBirth TEXT, " +
50                  "Address TEXT , " +
51                  "City TEXT, " +
52                  "Country TEXT, " +
53                  "PostCode TEXT, " +
54                  "Job TEXT, " +
55                  "Sex TEXT , " +
56                  "PhoneNumber TEXT, " +
57                  "Afm TEXT, " +
58                  "AMKA TEXT, " +
59                  "MaritalStatus TEXT, " +
60                  "DateHospitalization TEXT, " +
61                  "Ward Integer, " +
62                  "Url TEXT, " +
63                  "Email TEXT, " +
64                  "Fund TEXT, " +
65                  "Smoking TEXT, " +
66                  "Alcohol TEXT, " +
67                  "Diet TEXT)";
68
69          String CREATE_patientvitalsigns_TABLE = "CREATE TABLE if not exists patientvitalsigns ( " +
70                  "PatientVitalSignsId INTEGER PRIMARY KEY AUTOINCREMENT, " +
71                  "PatientId INTEGER, " +
72                  "VitalSignsId INTEGER )";
```

```
73
74          String CREATE_patientcomment_TABLE = "CREATE TABLE if not exists patientcomment ( " +
75                  "PatientCommentsId INTEGER PRIMARY KEY AUTOINCREMENT, " +
76                  "PatientId INTEGER, " +
77                  "CommentId INTEGER )";
78
79          String CREATE_patientward_TABLE = "CREATE TABLE if not exists patientward ( " +
80                  "PatientWardId INTEGER PRIMARY KEY AUTOINCREMENT, " +
81                  "PatientId INTEGER, " +
82                  "WardId INTEGER )";
83
84          // create patientdiagnosis table
85          db.execSQL(CREATE_patientdiagnosis_TABLE);
86          db.execSQL(CREATE_diagnosis_TABLE);
87          db.execSQL(CREATE_medication_TABLE);
88          db.execSQL(CREATE_patientmedication_TABLE);
89          db.execSQL(CREATE_comment_TABLE);
90          db.execSQL(CREATE_vitalsigns_TABLE);
91          db.execSQL(CREATE_patient_TABLE);
92          db.execSQL(CREATE_patientvitalsigns_TABLE);
93          db.execSQL(CREATE_patientcomment_TABLE);
94          db.execSQL(CREATE_patientward_TABLE);
95      }
```

Εικόνα 85: Μέθοδος OnCreate

5.2.2. Γέμισμα δεδομένων πινάκων

Για το γέμισμα των πινάκων που δημιουργήθηκαν παραπάνω χρησιμοποιούμε τις συναρτήσεις **add**, οι οποίες παίρνουν σαν ορίσματα τα στοιχεία που θέλουμε να εισάγουμε στον εκάστοτε πίνακα.

```
public void add_patientdiagnosis(int a, int b, String c){
public void add_diagnosis(String c){
public void add_medication(String c){
public void add_patientMedication(int a, int b, String c, String d, String e){
public void add_comment(String a,String b){
public void add_vitalsigns(String a,String b,String c,String d, String e,String f, String g,String h, String i){
public void add_patient(String a,String b,String c,String d, String e,String f, String g,String h, String i,
public void add_patientvitalsigns(int a, int b){
public void add_patientcomment(int a, int b){
public void add_patientward(int a, int b){
```

Εικόνα 86: Συναρτήσεις add

Παραθέτουμε στη συνέχεια το «σώμα» αυτού του είδους των συναρτήσεων και συγκεκριμένα αυτό της add_diagnosis(). Κατά τον ίδιο τρόπο υλοποιούνται και όλες οι υπόλοιπες.

```
public void add_diagnosis(String c){
    // 1. get reference to writable DB
    SQLiteDatabase db = this.getWritableDatabase();

    // 2. create ContentValues to add key "column"/value
    ContentValues values = new ContentValues();
    values.put("NameDiagnosis", c);

    // 3. insert
    db.insert("diagnosis", // table
            null, //nullColumnHack
            values); // key/value -> keys = column names/ values = column values

    // 4. close
    db.close();
}
```

Εικόνα 87: add_diagnosis

5.2.3. Ανάγνωση δεδομένων πινάκων

Για την ανάγνωση των στοιχείων των πινάκων που δημιουργήθηκαν παραπάνω χρησιμοποιούμε τις συναρτήσεις **get**, οι οποίες δεν παίρνουν κάποιο όρισμα.

```
// Get All PatientDiagnosis
public List<String> getAllPatientDiagnosis() {

// Get All PatientMedication
public List<String> getAllPatientMedication() {

// Get All Diagnosis (names+ids) order by name
public List<String> getAllDiagnosis() {

// Get All Medication (names+ids) order by name
public List<String> getAllMedication() {

// Get All Comments
public List<String> getAllComment() {

// Get All VitalSigns
public List<String> getAllVitalSigns() {

// Get All Patient
public List<String> getAllPatients() {

// Get All PatientDiagnosis
public List<String> getAllPatientVitalSigns() {

// Get All PatientComments
public List<String> getAllPatientComments() {

// Get All PatientWard
public List<String> getAllPatientWard() {
```

Εικόνα 88: Συναρτήσεις get

Παραθέτουμε στη συνέχεια το «σώμα» αυτού του είδους των συναρτήσεων και συγκεκριμένα αυτό της getAllDiagnosis(). Κατά τον ίδιο τρόπο υλοποιούνται και όλες οι υπόλοιπες.

```
// Get All Diagnosis (names+ids) order by name
public List<String> getAllDiagnosis() {
    List<String> Diagnosis = new LinkedList<String>();

    // 1. build the query
    String query = "SELECT  * FROM " + "diagnosis"+" ORDER BY NameDiagnosis";

    // 2. get reference to writable DB
    SQLiteDatabase db = this.getWritableDatabase();
    Cursor cursor = db.rawQuery(query, null);

    //3. Create Object
    if (cursor.moveToFirst()) {
        do {

            String diagId = cursor.getString(0);
            String diagName = cursor.getString(1);


          //Add to return list
           Diagnosis.add(diagId+","+diagName);
        } while (cursor.moveToNext());
    }
    return Diagnosis;
}
```

Εικόνα 89: getAllDiagnosis

5.2.4. Διαγραφή δεδομένων πινάκων

Για τη διαγραφή των στοιχείων των πινάκων που δημιουργήθηκαν παραπάνω χρησιμοποιούμε τις συναρτήσεις **delete**, οι οποίες δεν παίρνουν κάποιο όρισμα.

```
public void deleteAllPatientDiagnosis()
public void deleteAllDiagnosis()
public void deleteAllPatientMedication()
public void deleteAllMedication()
public void deleteAllComment()
public void deleteAllVitalSigns()
public void deleteAllPatients()
public void deleteAllPatientVitalSigns()
public void deleteAllPatientComments()
public void deleteAllPatientWard()
```

Εικόνα 90: Συναρτήσεις delete

Παραθέτουμε στη συνέχεια το «σώμα» αυτού του είδους των συναρτήσεων και συγκεκριμένα αυτό της deleteAllDiagnosis(). Κατά τον ίδιο τρόπο υλοποιούνται και όλες οι υπόλοιπες.

```
public void deleteAllDiagnosis()
{
    // 1. build the query
    String query = "DELETE FROM diagnosis";
    // 2. get reference to writable DB
    SQLiteDatabase db = this.getWritableDatabase();
    db.execSQL(query);
    db.close();
}
```

Εικόνα 91: deleteAllDiagnosis

5.3. Χρήση στην εφαρμογή

Η ιδέα για το πώς ακριβώς θα χρησιμοποιήσουμε την τοπική βάση που δημιουργήσαμε είναι η εξής:

Επιχειρούμε σύνδεση στο serverκαι σε περίπτωση αποτυχίας σύνδεσης αποθηκεύουμε τοπικά τα δεδομένα που δόθηκαν, ώστε σε περίπτωση που πατηθεί το κουμπί «refresh» να επιχειρήσουμε συγχρονισμό της τοπικής βάσης με την κεντρική καιό,τι έχει αποθηκευτεί τοπικά να μεταφερθεί στην κεντρική βάση και κατόπιν, προς ελευθέρωση των πόρων του συστήματος , να διαγραφεί από την τοπική.

Για λόγους διευκόλυνσης κατά τη διαδικασία εκτέλεσης της εφαρμογής, στην περίπτωση των γιατρών έχει υλοποιηθεί ο έλεγχος για σύνδεση στο server, όμως θεωρήσαμε επιτυχή σύνδεση και συνεπώς η τοπική βάση δεν θα προσπελαστεί.

Αντιθέτως, στην περίπτωση των νοσηλευτών η τοπική αποθήκευση λαμβάνει χώρα ούτως ή άλλως. Ουσιαστικά τα δεδομένα που δίνει ο νοσηλευτής αποθηκεύονται μόνο στην τοπική βάση χωρίς προσπάθεια σύνδεσης με την κεντρική. Η μεταφορά αυτών των δεδομένων στην κεντρική βάση γίνεται μόνο όταν πατηθεί από το χρήστη το κουμπί «refresh»στη σελίδα που γίνεται το login.

Όπως προαναφέρθηκε η διαφορά υλοποίησης στις 2 αυτές περιπτώσεις έγινε καθαρά για πρακτικούς λόγους, δηλαδή να μη χρειάζεται να κλείνουμε τη σύνδεση, ώστε να δείχνουμε τη χρήση της τοπικής βάσης. Πρακτικά λοιπόν με αυτό τον τρόπο δείχνουμε με μία κάπως χειροκίνητημέθοδο τη λειτουργία της εφαρμογής σε περίπτωση επιτυχούς και μη επιτυχούς σύνδεσης μαζί.

Παρουσιάζονται παρακάτω κάποια ενδεικτικά αποσπάσματα του κώδικα Javaγια να γίνει κάπως πιο κατανοητή αυτή η λειτουργικότητα.

Αρχικά παρατίθεται ο κώδικας όπου ένας γιατρός επιλέγει κάποια από τις διαθέσιμες διαγνώσεις προς προσθήκη για έναν ασθενή. Ουσιαστικά στην περίπτωση που **υπάρχει** σύνδεση γίνεται νέο insertστους κατάλληλους πίνακεςτης κεντρικής βάσης με κλήση της μεθόδου InsertDiagnosis, **αλλιώς**φορτώνονται τα απαραίτητα δεδομένα στον πίνακα patientdiagnosistης τοπικής βάσης. Και στις 2 περιπτώσεις ο χρήστης ενημερώνεται με toastγια την κατάσταση του συστήματος(Επιτυχής/Αποτυχημένη Σύνδεση στο server)

```
//if we have connection diagnosis will be loaded from server database and new row
//patientdiagnosis will be stored to server database on add button
if(connection){
    CustomList4 adapter = new CustomList4 (ChooseDiagnosis.this, DiagnosisName, DiagnosisId);
    this.display.setAdapter(adapter);
    allTiks = adapter.getAllTiks();
    allTiksId = adapter.getAllTiksId();
    //Toast.makeText(ChooseDiagnosis.this,allTiks.size()+"\n" +allTiksId.size(), Toast.LENGTH_LONG).show();
    this.add.setOnClickListener(new OnClickListener(){
        @Override
        public void onClick(View v) {
            Toast.makeText(ChooseDiagnosis.this,"Επιτυχής Σύνδεση - Αποθήκευση διάγνωσης στην κεντρική βάση(server)..",
                    Toast.LENGTH_LONG).show();
            String DateDiagnosis = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")
                                        .format(Calendar.getInstance().getTime());
            for (int i=0; i<allTiks.size();i++){
                if (allTiks.get(i).isChecked()) {
                    Integer DiagnId = allTiksId.get(i);
                    //Toast.makeText(ChooseDiagnosis.this,PatientId+"\n"+DiagnId+"\n"+DateDiagnosis,
                    //Toast.LENGTH_LONG).show();
                    new InsertDiagnosis(PatientId, DiagnId, DateDiagnosis).execute("");
                   // db.add_patientdiagnosis(PatientId, DiagnId, DateDiagnosis);
                }
            }

        }
    });

}
```

```
//if we don't have connection diagnosis will be loaded from local database and new row
//patientdiagnosis will be added  to local database on add button in order to be stored in server
//database when connection is availiable on refresh button
else{
    CustomList4 adapterNC = new CustomList4 (ChooseDiagnosis.this, DiagnosisNameNC, DiagnosisIdNC);
    this.display.setAdapter(adapterNC);
    allTiks = adapterNC.getAllTiks();
    allTiksId = adapterNC.getAllTiksId();
    this.add.setOnClickListener(new OnClickListener(){
        @Override
        public void onClick(View v) {
            Toast.makeText(ChooseDiagnosis.this,"Αποτυχημένη Σύνδεση - Αποθήκευση διάγνωσης στην τοπική βάση..",
                    Toast.LENGTH_LONG).show();
            String DateDiagnosis = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(Calendar.getInstance().getTime());
            for (int i=0; i<allTiks.size();i++){
                if (allTiks.get(i).isChecked()) {
                    Integer DiagnId = allTiksId.get(i);
                    Toast.makeText(ChooseDiagnosis.this,"Προσθήκη στην τοπική βάση..", Toast.LENGTH_SHORT).show();
                    db.add_patientdiagnosis(PatientId, DiagnId, DateDiagnosis);
                }
            }

        }
    });

}
```

Εικόνα 92: Λειτουργικότητα σε λειτουργία Γιατρού, σε περίπτωση επιτυχούς και αποτυχούς σύνδεσης

Στη συνέχεια παρατίθεται ο κώδικας όπου ένας νοσηλευτής εισάγει κάποιο σχόλιο για έναν ασθενή. Το στοιχείο που εισάγει μαζί με την ημ/νία σχολίου που λαμβάνεται αυτόματα από το σύστημα αποθηκεύονται τοπικά στον πίνακα commenttης SQLite και ο χρήστης ενημερώνεται για αποτυχία σύνδεσης με toast.

```
/** Called when the user clicks the Add button
 * a new comment added in local Database*/
public void AddButton(View view) {

    comment = ed1.getText().toString();

    Toast.makeText(NurseCommentTab.this,"Αποτυχημένη σύνδεση - Αποθήκευση Σχολίου στην Τοπική
            //  +comment+"\n"+date
            , Toast.LENGTH_LONG).show();

    db.add_comment(date, comment);
    new GetCommentId(PatientId).execute("");
}
```

Εικόνα 93: Λειτουργικότητα σε περίπτωση αποτυχίας σύνδεσης σε λειτουργία Νοσηλευτή

Με παρόμοιο τρόπο υλοποιούνται όλες οι περιπτώσεις κατά τις οποίες γιατρός ή νοσηλευτής επιθυμεί να προσθέσει κάποια στοιχεία για τον επιλεγμένο ασθενή.

Κατά την εκκίνηση της εφαρμογής, στη δεύτερη σελίδα σύνδεσης του χρήστη, υπάρχει το πράσινο κουμπί «refresh», μέσω του οποίου υλοποιείται συγχρονισμός των 2 βάσεων. Όποιος χρήστης έχει πρόσβαση στην εφαρμογή πατάει για ασφάλεια το συγκεκριμένο κουμπί κατά τη σύνδεσή του ώστε όσα δεδομένα υπάρχουν στην τοπική βάση να μεταφερθούν στην κεντρική.

Η θέση του συγκεκριμένου κουμπιού αρχικά ήταν σε κάθε σελίδα της εφαρμογής όπου γινόταν τοπική αποθήκευση, ώστε ο χρήστης να μπορεί ανά πάσα στιγμή να επιχειρεί μέσω αυτού σύνδεση στο serverκαι μεταφορά δεδομένων(συγχρονισμό).

Στη συνέχεια η θέση του συγκεκριμένου κουμπιού άλλαξε και πλέον είναι στην οθόνη που ο χρήστης κάνει login. Πρέπει να σημειωθεί πως κατά τη διαδικασία αποσύνδεσης ο χρήστης πετιέται σε αυτή την οθόνη και υπάρχει σχετικό toastπου τον προτρέπει να κάνει refresh. Για παράδειγμα στο σενάριο πως ένας νοσηλευτής εισάγει νέο ασθενή, κάποιο σχόλιο για υπάρχον ασθενή και κάποιες μετρήσεις ζωτικών σημείων υπάρχοντος ασθενή η εφαρμογή μας λειτουργεί κατ’αυτόν τον τρόπο:

- Login χρήστη
- Επιλογή εισαγωγής νέου ασθενή
- Επιλογή εύρεσης ασθενή από τους ήδη υπάρχοντες στη βάση
- Εγγραφή σχολίου και ζωτικών σημείων
- Αποσύνδεση και συγχρονισμός ώστε οι νέες εγγραφές να μεταφερθούν στην κεντρική βάση

```
/** Called when the user clicks the "Ανανέωση" button */
@SuppressLint("ShowToast")
public void Refresh(View view) {
    Toast.makeText(this, "Συγχρονισμός με κεντρική βάση(server)", Toast.LENGTH_LONG).show();

    //Add New Patient to Server
    pat = db.getAllPatients();
    for (int i=0; i<pat.size();i++){
        String delims = "[,]";
        String[] tokens = new String[24];
        tokens = pat.get(i).split(delims);
        AddNewPatient a = new AddNewPatient();
        a.new InsertPatient(tokens[1], tokens[2], tokens[3], tokens[4], tokens[5], tokens[6],
                            tokens[7], tokens[8], tokens[9], tokens[10], tokens[11], tokens[12],
                            tokens[13], tokens[14], tokens[15], tokens[16], tokens[18],
                            tokens[19], tokens[20], tokens[21], tokens[22], tokens[23]).execute("");

        a.new GetPatientId(Integer.parseInt(tokens[17])).execute("");
        }
    db.deleteAllPatients();


    //Add New VitalSignMeasurement to Server
    vitalS = db.getAllVitalSigns();
    for (int i=0; i<vitalS.size();i++){
        String delims = "[,]";
        String[] tokens = new String[10];
        tokens = vitalS.get(i).split(delims);
        NurseVitalSignsTab v = new NurseVitalSignsTab();
        v.new InsertVitalSign(tokens[1], Float.parseFloat(tokens[2]), Integer.parseInt(tokens[3]),
                Integer.parseInt(tokens[4]), Integer.parseInt(tokens[5]), Float.parseFloat(tokens[6]),
                    Float.parseFloat(tokens[7]), Integer.parseInt(tokens[8]), Float.parseFloat(tokens[9])).execute("");
    }
    db.deleteAllVitalSigns();


//Add New PatientVitalSign to Server
PatientVitalS = db.getAllPatientVitalSigns();
for (int i=0; i<PatientVitalS.size();i++){
    String delims = "[,]";
    String[] tokens = new String[10];
    tokens = PatientVitalS.get(i).split(delims);
    NurseVitalSignsTab v = new NurseVitalSignsTab();
    v.new InsertPatientVitalSign(Integer.parseInt(tokens[0]), Integer.parseInt(tokens[1])).execute("");
    }
db.deleteAllPatientVitalSigns();

//Add new comment to server
Comments = db.getAllComment();
for (int i=0; i<Comments.size();i++){
    String delims = "[,]";
    String[] tokens = new String[3];
    tokens = Comments.get(i).split(delims);
    NurseCommentTab n = new NurseCommentTab();
    n.new InsertComment(tokens[1],tokens[2]).execute("");
    //new GetCommentId(PatientId).execute("");
    }
db.deleteAllComment();

//Add new patientComment to server
PatientComments = db.getAllPatientComments();
for (int i=0; i<PatientComments.size();i++){
    String delims = "[,]";
    String[] tokens = new String[10];
    tokens = PatientComments.get(i).split(delims);
    NurseCommentTab n = new NurseCommentTab();
    n.new InsertPatientComment(Integer.parseInt(tokens[0]),Integer.parseInt(tokens[1])).execute("");

    }
db.deleteAllPatientComments();
```

```
//Add new PatientDiagnosis
Diagnosis = db.getAllPatientDiagnosis();
for (int i=0; i<Diagnosis.size();i++){
    String delims = "[ ]";
    String[] tokens = new String[5];
    tokens = Diagnosis.get(i).split(delims);
    ChooseDiagnosis c = new ChooseDiagnosis();
    c.new InsertDiagnosis(Integer.parseInt(tokens[0]), Integer.parseInt(tokens[1]), tokens[2]).execute("");
}
db.deleteAllPatientDiagnosis();

//Add new PatientMedicine
Medicine = db.getAllPatientMedication();
for (int i=0; i<Medicine.size();i++){
    String delims = "[,]";
    String[] tokens = new String[6];
    tokens = Medicine.get(i).split(delims);
    ChooseMedicine m = new ChooseMedicine();
    m.new InsertMedicine(Integer.parseInt(tokens[0]), Integer.parseInt(tokens[1]), tokens[2], tokens[3],
            tokens[4]).execute("");
}
db.deleteAllPatientMedication();

}
```

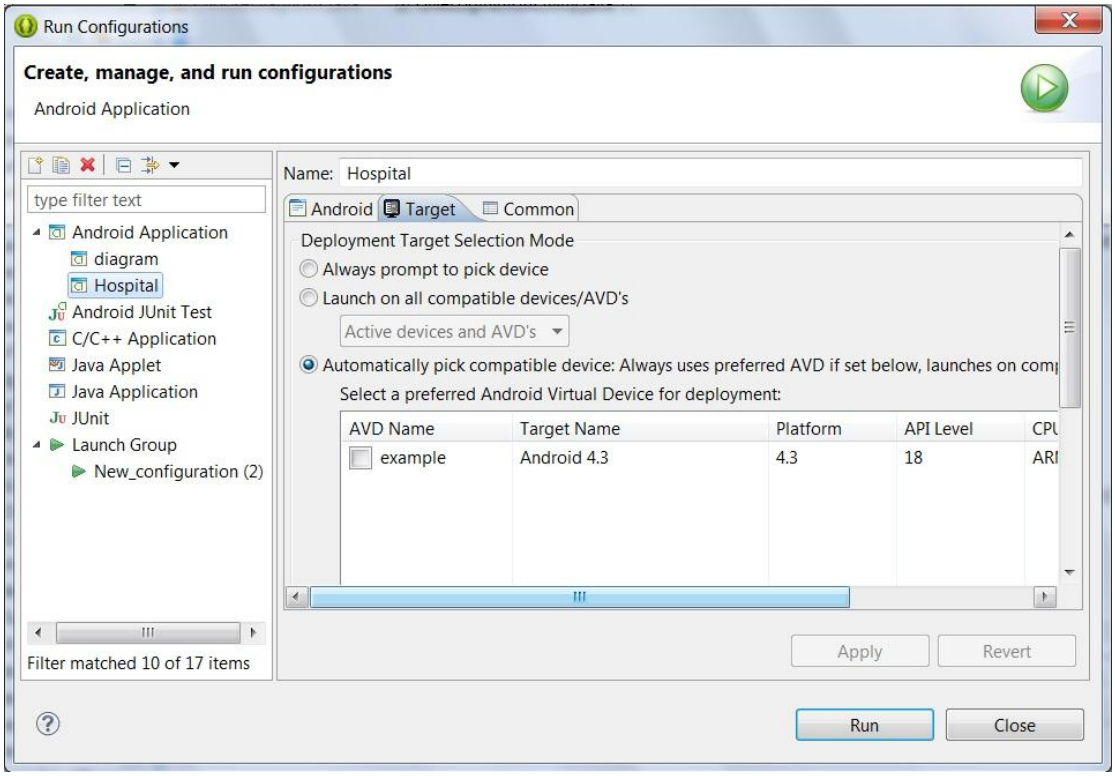
Εικόνα 94: Κώδικας για refreshbutton

6. ΠΑΡΟΥΣΙΑΣΗ ΕΦΑΡΜΟΓΗΣ

Στο παρόν κεφάλαιο παρουσιάζεται βήμα προς βήμα η εφαρμογή, η οποία καθορίσαμε να τρέχεισε έκδοση Android 4.3JellyBean – ApiLevel 18 στη συσκευή NexusOne(3.7’’, 480 x 800: hdpi) με μνήμη: 512 , VMHeap: 32 και InternalStorage : 200 MiB, μέσω του AndroidVirtualDeviceManager. Το AVDManagerπαρέχει μια γραφική διεπαφή στο χρήστη μέσω της οποίας μπορεί να δει και να διαχειριστεί τα AVDs(συσκευές) που απαιτούνται για να τρέξει το AndroidEmulator.

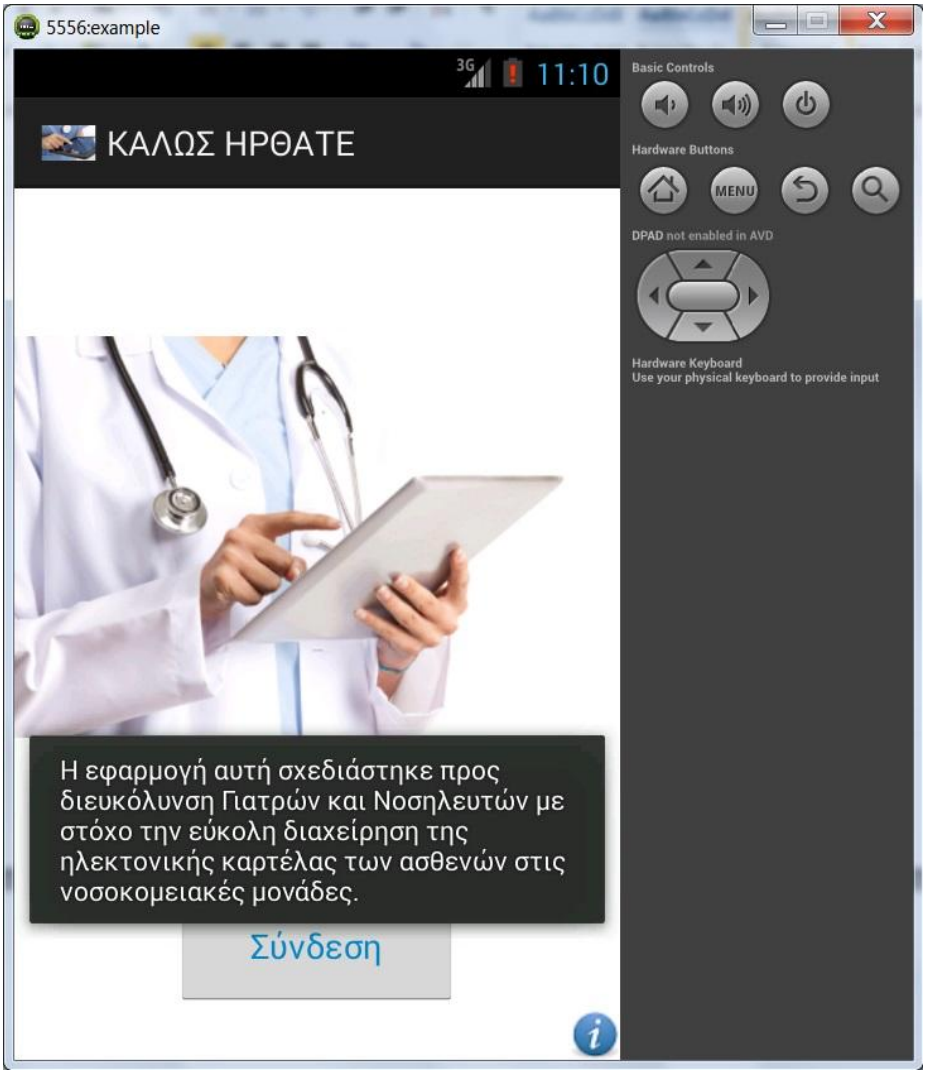
Ο κώδικας υλοποιήθηκε σε Javaμέσω του εργαλείου eclipseκαι του AndroidSdk, το οποίο παρέχει τις βιβλιοθήκες APIκαι τα απαραίτητα εργαλεία για την κατασκευή, δοκιμή και εντοπισμό σφαλμάτων στις εφαρμογές Android. Στο σύνολο υλοποιήθηκαν 29 κλάσεις, κάποιες από τις οποίες είναι απλές κλάσεις και άλλες που είναι τύπου androidactivity. Κάθε κλάση τύπου activityκατά τη δημιουργία της δημιουργεί αυτόματα με αντίστοιχο όνομα ένα αρχείο .xmlστο οποίο ο developerφτιάχνει το γραφικό περιβάλλον της κάθε activity, πρακτικά δηλαδή σχεδιάζει τα γραφικά κομμάτια που θα περιλαμβάνει κάθε σελίδα. Σε σύνολο, στη συγκεκριμένη εφαρμογή υπάρχουν 21 αρχεία .xml, που αποθηκεύονται στο φάκελο layouttου project. Επίσης στο φάκελο drawable-mdpiυπάρχουν όλες οι εικόνες(στο σύνολο 19) που χρησιμοποιήθηκαν για την εφαρμογή και είναι σε μορφή .png.

Για να τρέξουμε την εφαρμογή μας, στο πάνω μέρος των εργαλείων του eclipseεπιλέγουμε το RunConfigurationsκαι πετάγεται παράθυρο στο οποίο επιλέγουμε το όνομα του projectπου επιθυμούμε να τρέξουμε και στο πεδίο Targetεπιλέγουμε το όνομα της συσκευής AVDπου επιθυμούμε να είναι το περιβάλλον. Κατόπιν πατάμε το κουμπί run.



Εικόνα 95: Run

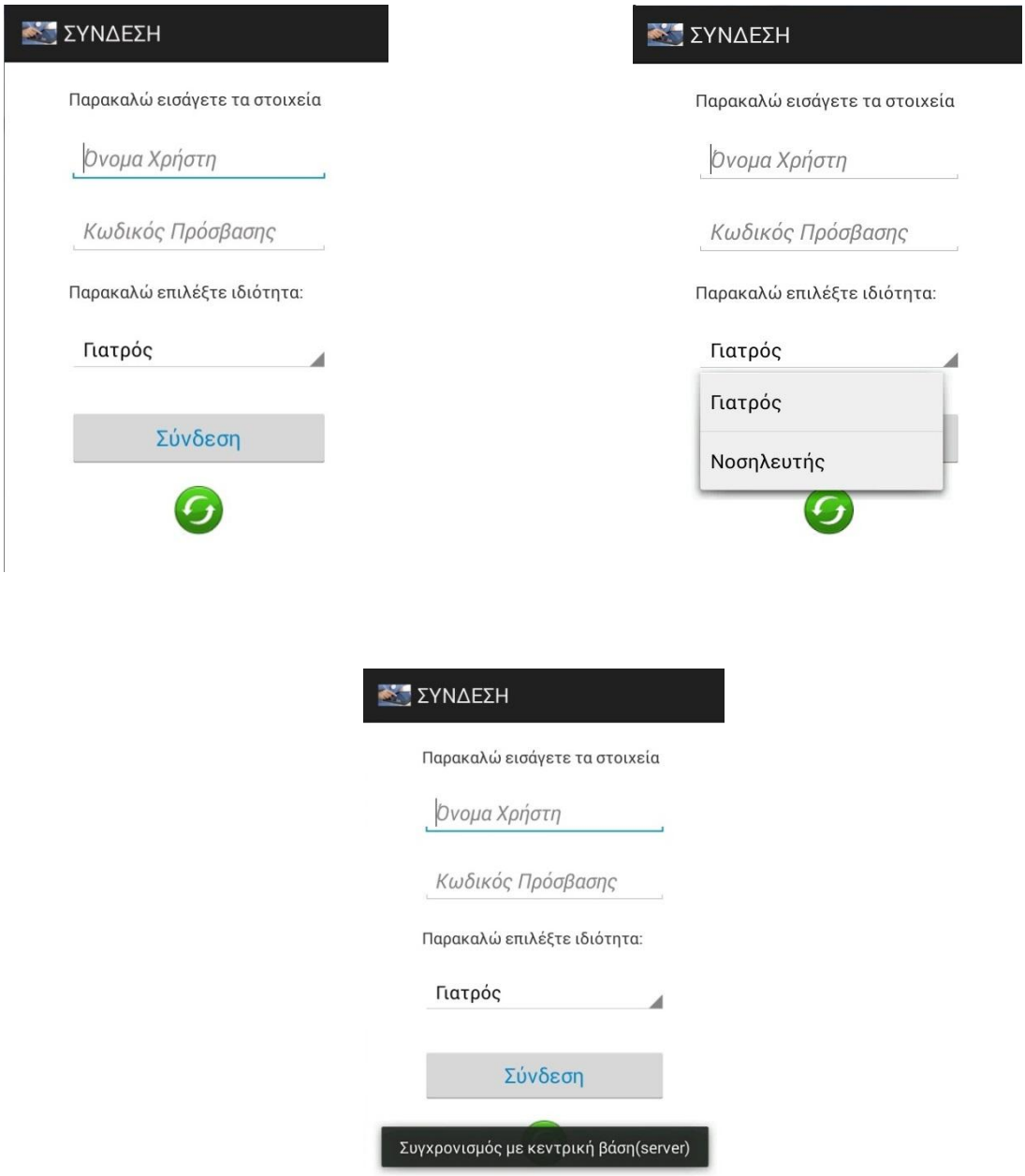
Ανοίγει ο emulator(προσωμοιτής μας), τον ξεκλειδώνουμε και εμφανίζεται η πρώτη σελίδα της εφαρμογής. Στο πάνω μέρος της σελίδας αυτής υπάρχει αριστερά το εικονίδιο της εφαρμογής και ο τίτλος της συγκεκριμένης οθόνης «ΚΑΛΩΣ ΗΡΘΑΤΕ». Στο κέντρο της σελίδας υπάρχει μία εικόνα που παραπέμπει το χρήστη στο είδος της εφαρμογής, ένα κεντρικό κουμπί «Σύνδεση» και ένα πιο μικρό κουμπί στο κάτω δεξιά μέρος της οθόνης σχετικό με πληροφορίες για το σύστημα.Αν ο χρήστης πατήσει το δεύτερο κουμπί, εμφανίζεται για μερικά δευτερόλεπτα ένα μήνυμα τύπου toastπου παρουσιάζει γενικού περιεχομένου πληροφορίες για την εφαρμογή.



Εικόνα 96: WelcomePage

Στη συνέχεια ο χρήστης πατάει το κεντρικό κουμπί «Σύνδεση» και έτσι μεταβαίνει στη δεύτερη σελίδα κατά την οποία η εφαρμογή τον προτρέπει να εισάγει τα στοιχεία του. Στη συγκεκριμένη σελίδα υπάρχουν 2 edittext στοιχεία, ένα για εισαγωγή ονόματος χρήστη και ένα για κωδικό πρόσβασης, στοιχεία τα οποία προς διευκόλυνση του χρήστη φαίνονται σαν hints στα αντίστοιχα πεδία. Ακολουθεί μήνυμα για να εισαχθεί ιδιότητα και ο χρήστης σε αυτή την περίπτωση μέσω spinnerεπιλέγει την ιδιότητα που επιθυμεί Γιατρός/Νοσηλευτής. Το spinnerbydefaultέχει προεπιλεγμένη την ιδιότητα Γιατρός. Επίσης υπάρχουν 2 ακόμη κουμπιά «Σύνδεση» και ένα ImageButton που γραφικά παραπέμπει στη λειτουργία refresh. Με το κουμπί«Σύνδεση» ο χρήστης μεταβαίνει ανάλογα με την

ιδιότητά του στην κατάλληλη σελίδα, ενώ με το πράσινο κουμπί επιχειρείται συγχρονισμός της τοπικής με την κεντρική βάση, ώστε ό,τι στοιχεία τυχόν υπήρχαν αποθηκευμένα στην τοπική να περαστούν στην κεντρική. Καλό θα ήταν όποτε ο χρήστης μεταβαίνει στη συγκεκριμένη σελίδα(είτε κατά την πρώτη επαφή με την συσκευή είτε κατά την αποσύνδεση) να πατάει για σιγουριά το συγκεκριμένο κουμπί. Σε περίπτωση που ο χρήστης το πατήσει εμφανίζεται ενημερωτικό μήνυμα για σύνδεση με την κεντρική βάση και συγχρονισμό.

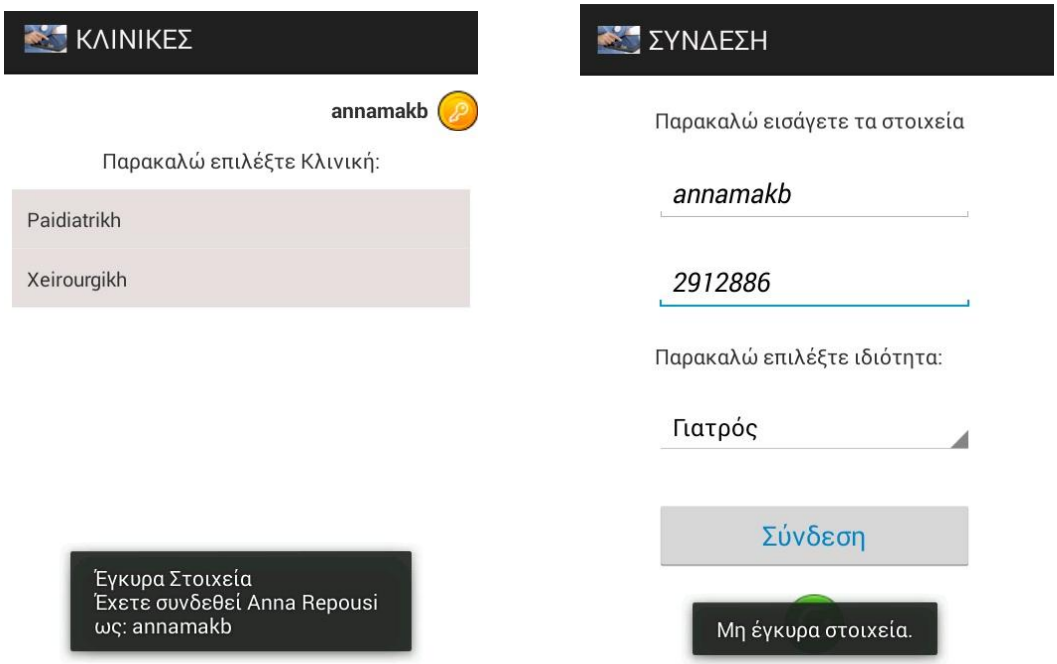


Εικόνα 97: Login Page

6.1. Οθόνες Γιατρού

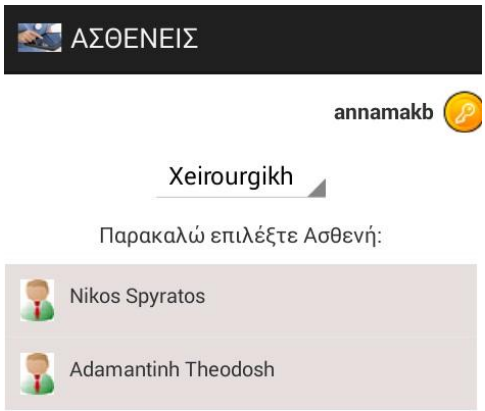
Στο παρόν κεφάλαιο θεωρούμε πως ο χρήστης κάνει login ως **γιατρός** με στοιχεία «annamakb» (username) & «2912888» (password). Αφότου ο χρήστης δώσει τα συγκεκριμένα στοιχεία γίνεται αίτηση στο server μέσω .php αρχείου για αναζήτηση γιατρού με τα συγκεκριμένα στοιχεία.Σε περίπτωση που δεν βρεθεί γιατρός με αυτά τα στοιχεία επιστρέφεται μήνυμα λάθους, το οποίο εμφανίζεται με μορφή toast στο χρήστη. Γνωρίζουμε ως developertου συστήματος πως τα στοιχεία που πληκτρολόγησε ο χρήστης είναι έγκυρα και με το κουμπί σύνδεση μεταβαίνουμε στην επόμενη σελίδα. Για να δείξουμε και τι εμφανίζεται σε περίπτωση μη εγκυρων στοιχείων πληκτρολογούμε κάποια λάθος και εμφανίζουμε την αντίστοιχη οθόνη.

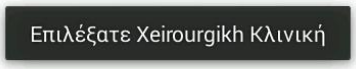
Σε περίπτωση έγκυρου loginη οθόνη που μεταβαίνει ο γιατρός τον ενημερώνει για όλες τις κλινικές με τις οποίες σχετίζεται και τον προτρέπει να επιλέξει κάποια από αυτές. Να σημειωθεί επίσης πως πάνω δεξιά εμφανίζεται το usernameμε το οποίο εισήχθη ο γιατρός και δίπλα του ένα κουμπάκι αποσύνδεσης από το σύστημα. Σε περίπτωση αποσύνδεσης, ο χρήστης μεταβαίνει στη σελίδα Loginπου περιγράψαμε πριν. Τα 2 αυτά στοιχεία (username&κουμπί αποσύνδεσης) θα υπάρχουν και στις επόμενες οθόνες που ακολουθούν έχοντας ακριβώς την ίδια λειτουργικότητα, δηλαδή μετάβαση στο LoginPage.



Εικόνα98: ΈγκυροLogin: ΜετάβασηστοClinicsPage(α), ΜηέγκυροLogin: ΠαραμονήστοLoginPage(β)

Στη συνέχεια, από τις κλινικές που παρατίθενται,ο γιατρός επιλέγει τη Χειρουργική Κλινική. Να σημειωθεί πως ο λόγος για τον οποίο φαίνονται στα greeklish οι κλινικές και γενικότερα όσα στοιχεία προκύπτουν από την κεντρική βάση, είναι πως για κάποιο λόγο με το phpMyAdminδεν καταφέραμε να παίρνουμε ελληνικά. Επιλέγοντας τη χειρουργική, μεταβαίνουμε σε νέα οθόνη στην οποία εμφανίζονται όλοι οι ασθενείς της συγκεκριμένης κλινικής. Ουσιαστικά μεταβαίνουμε στην κεντρική βάση και ψάχνουμε τους ασθενείς όλων των θαλάμων που ανήκουν στη συγκεκριμένη κλινική. Στη συγκεκριμένη σελίδα υπάρχει ένα spinnerμέσω του οποίου ο γιατρός μπορεί ανά πάσα στιγμή να μεταβεί σε όποια από τις υπόλοιπες κλινικές σχετίζεται για να βρει τους ασθενείς της. Σε περίπτωση που κάποια κλινική δεν έχει ασθενείς, η απάντηση του servereίναι ένα σχετικό μήνυμα το οποίο και πετάμε σαν ενημερωτικό toastστο χρήστη.



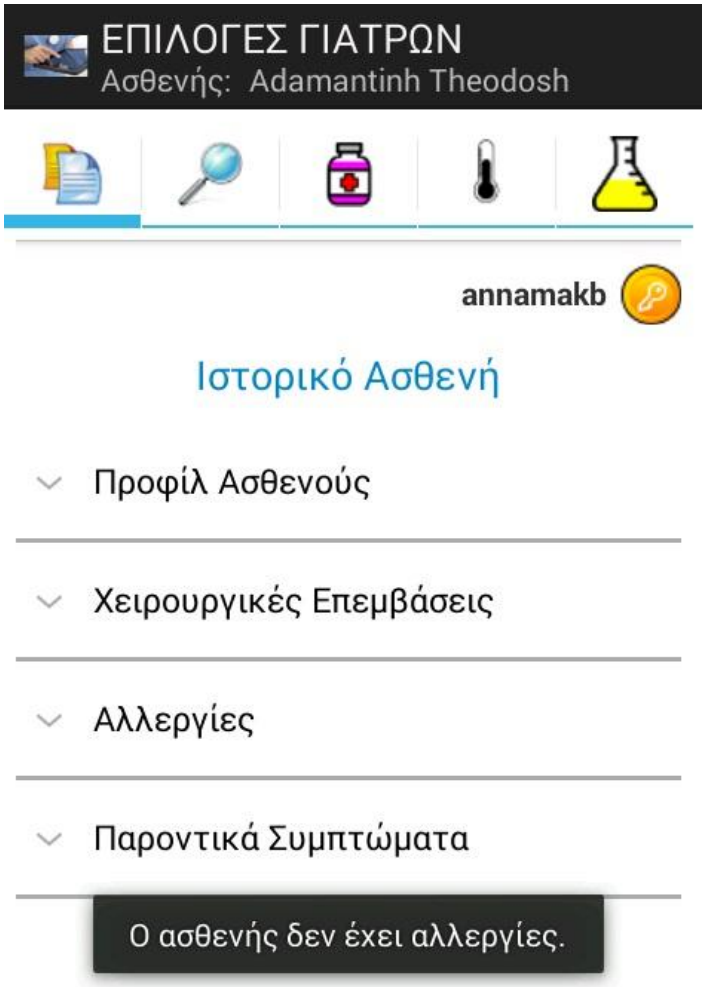


Εικόνα99: PatientsPage(α), Spinner(β), Αλλαγήκλινικήςμέσωspinner(γ), Toastσεπερίπτωση απουσίας ασθενών από κλινική(δ)

Θεωρούμε πως ο γιατρός επιλέγει τον ασθενή AdamantiniTheodosiκαι μετά από αυτή του την επιλογή γίνεται μετάβαση σε νέα οθόνη στην οποία παρουσιάζονται με μορφή tabsόλες οι λειτουργίες που μπορεί να εκτελέσει στο συγκεκριμένο ασθενή. Είναι προεπιλεγμένο το πρώτο tabπου σχετίζεται με το ιστορικό του ασθενή και σε περίπτωση που δεν υπάρχουν χειρουργία ή αλλεργίες ή παροντικά συμπτώματα πετιέται ανάλογο ενημερωτικό toast. Επιπλέον στο πάνω μέρος της σελίδας εμφανίζεται το όνοματεπώνυμο του ασθενή που εξετάζουμε, ώστε ο γιατρός να γνωρίζει ανά πάσα στιγμή σε ποιον ασθενή αναφέρεται.

Το **πρώτο tab**, που είναι και προεπιλεγμένο, ουσιαστικά περιέχει μια expandablelist, στην οποία εμφανίζονται κατηγοριοποιημένα τα στοιχεία του ασθενή(προφίλ, χειρουργικές επεμβάσεις, αλλεργίες και παροντικά συμπτώματα). Ανοίγοντας το «Προφίλ Ασθενούς», φορτώνονται από τη βάση όλα τα προσωπικά στοιχεία του ασθενή και παρατίθενται με μορφή λίστας, συμπεριλαμβανομένου και της κλινικής, θαλάμου, κρεβατιού στο οποίο νοσηλεύεται. Ανοίγοντας το «Χειρουργικές Επεμβάσεις», παρουσιάζονται όλα τα χειρουργεία που έχει υποστεί, συνοδευόμενα από την ονομασία τους, την ημ/νία διεξαγωγής τους και κάποια σχόλια που τυχόν έχουν γίνει, δεδομένα τα οποία προκύπτουν επίσης από την

κεντρική βάση. Ανοίγοντας το «Αλλεργίες», φαίνονται όλες οι αλλεργίες που παρουσιάζει ο ασθενής. Στην περίπτωση της AdamantinhTheodosiδεν υπάρχουν κάποιες , όμως από τον κώδικα έχει καθοριστεί να παρουσιάζονται πάλι σε μορφή λίστας με την περιγραφή τους και κάποιες ακόμη λεπτομέρειες. Τέλος, όσο αφορά τα παροντικά συμπτώματα, ουσιαστικά αναφερόμαστε σε συμπτώματα τα οποία παρουσιάζει ο ασθενής όσο διάστημα νοσηλεύεται στο νοσοκομείο. Ο λόγος ύπαρξης της συγκεκριμένης πληροφορίας είναι η όσο το δυνατόν πιο ενδελεχής πληροφόρηση του γιατρού για το προφίλ του ασθενή που παρακολουθείται. Για κάθε σύμπτωμα δίνονται πληροφορίες για την ονομασία, κάποιο σχόλιο και την κωδικοποίηση του στο Icd10.

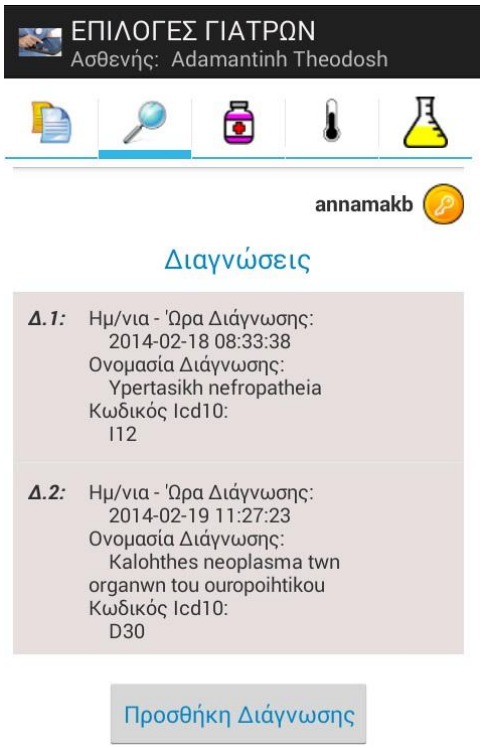


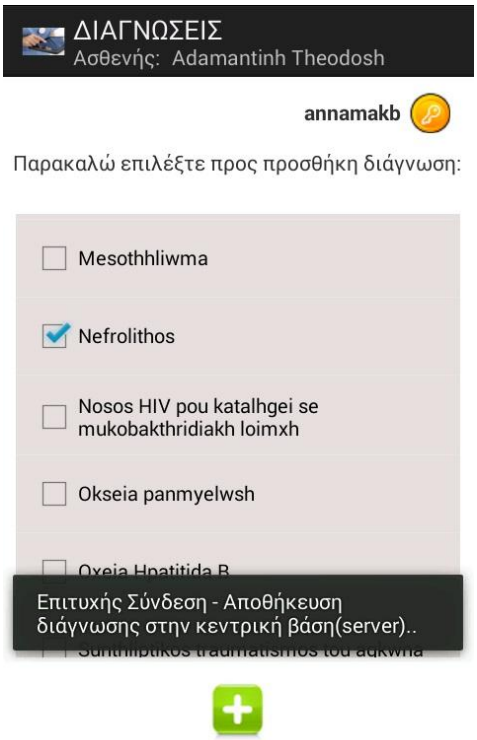
Εικόνα 98: DoctorTabs

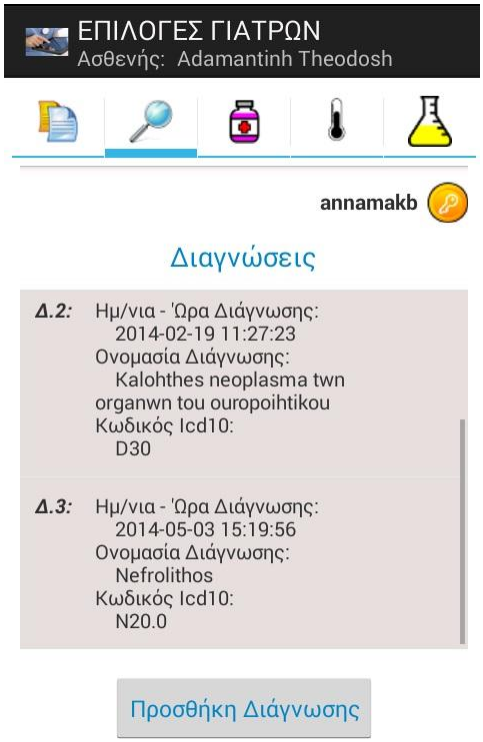
Εικόνα 99: Προφίλ Ασθενούς

Εικόνα 100: Χειρουργικές Επεμβάσεις(α), Παροντικά Συμπτώματα(β)

Το **δεύτερο tab**σχετίζεται με τις διαγνώσεις. Παρουσιάζονται στο γιατρό οι διαγνώσεις που έχουν ήδη γίνει καθ' όλη τη διάρκεια νοσηλείας του ασθενή συνοδευόμενες από την ημ/νία που έγιναν και την κωδικοποίηση σύμφωνα με το Icd10. Έτσι ο γιατρός μπορεί να δει τι έχει διαγνωσθεί μέχρι τώρα για το συγκεκριμένο ασθενή. Ο λόγος ύπαρξης παραπάνω της μίας διάγνωσης είναι καθαρά ρεαλιστικός, καθώς στην πραγματικότητα ο ασθενής μπορεί στην αρχή της νοσηλείας του να πάσχει από κάποια ασθένεια και να προκύψει κάποια άλλη στην πορεία ή να πάσχει από παραπάνω της μίας. Πατώντας το κουμπί «Προσθήκη Διάγνωσης», ο γιατρός μεταβαίνει σε μια νέα σελίδα στην οποία φορτώνονται από την κεντρική βάση όλες οι πιθανές διαγνώσεις. Θεωρήσαμε ευκολότερο προς το χρήστη να έχει πρόσβαση σε όλες τις δυνατές διαγνώσεις επιλέγοντας με τικ προς προσθήκη μία ή και περισσότερες. Φυσικά για πρακτικούς λόγους φορτώθηκε στη βάση περιορισμένος αριθμός διαγνώσεων. Ο γιατρός στο παράδειγμα που τρέχουμε επιλέγει τη διάγνωση Νεφρόλιθος και πατάει το κουμπί add. Κατόπιν για να ελέγξουμε τη σωστή εισαγωγή της νέας διάγνωσης επιλέγουμε ξανά το tabτων διαγνώσεων στο οποίο αυτή τη φορά θα πρέπει να έχει προστεθεί η διάγνωση που μόλις εισήχθη, πράγμα που όντως συμβαίνει.

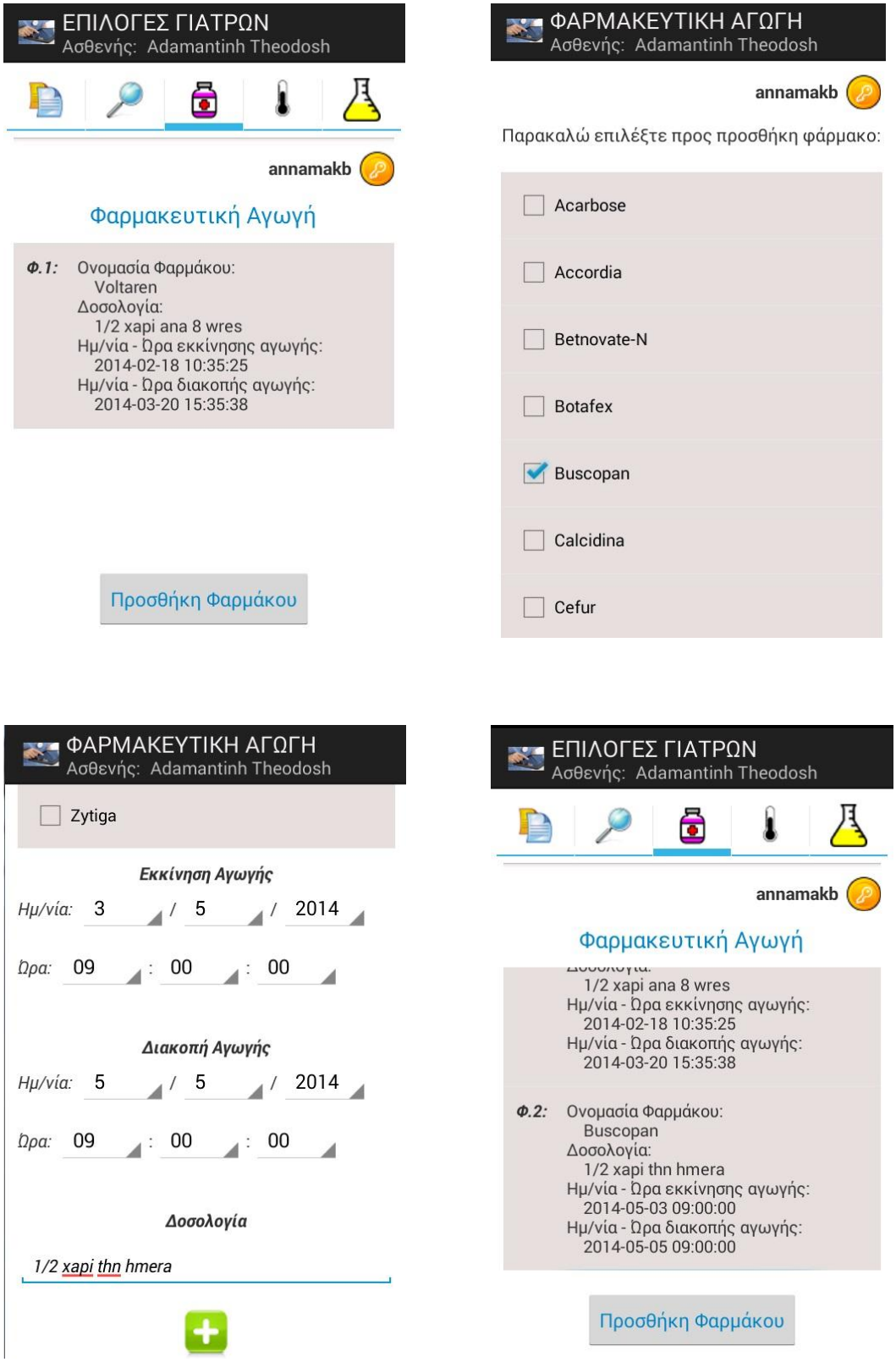






Εικόνα 101: Διαγνώσεις - Αρχική σελίδα(α), Σελίδα Προσθήκης διάγνωσης(β), Επιστροφή στο tab Διαγνώσεων για επαλήθευση(γ)

Το **τρίτο tab**σχετίζεται με τη φαρμακευτική αγωγή. Παρουσιάζονται στο γιατρό τα φάρμακα που ήδη λαμβάνει ο ασθενής καθ' όλη τη διάρκεια νοσηλείας του, συνοδευόμενα από την ημ/νία εκκίνησης, διακοπής και τη δοσολογία. Πατώντας το κουμπί «Προσθήκη Φαρμάκου», ο γιατρός μεταβαίνει σε μια νέα σελίδα στην οποία φορτώνονται από την κεντρική βάση όλα τα διαθέσιμα φάρμακα. Θεωρήσαμε ευκολότερο προς το χρήστη να έχει πρόσβαση σε όλα τα φάρμακα επιλέγοντας με τικ προς προσθήκη ένα ή και περισσότερα. Φυσικά για πρακτικούς λόγους φορτώθηκε στη βάση περιορισμένος αριθμός φαρμάκων. Ο γιατρός στο παράδειγμα που τρέχουμε επιλέγει το φάρμακο Buscopanκαι ορίζει ημερομηνία και ώρα εκκίνησης-διακοπής της αγωγής καθώς και δοσολογία με τη βοήθεια spinnerγια μέρα, μήνα, έτος, ώρα, λεπτά & δευτερόλεπτα και edittextγια εισαγωγή δοσολογίας από το πληκτρολόγιο αντίστοιχα.Στη συνέχεια πατάει το κουμπί add. Για να ελέγξουμε τη σωστή εισαγωγή του νέου φαρμάκου επιλέγουμε ξανά το tabτων φαρμάκων στο οποίο αυτή τη φορά θα πρέπει να έχει προστεθεί το φάρμακο που μόλις εισήχθη με όλες τις extraπληροφορίες, πράγμα που όντως συμβαίνει.



Εικόνα 102: Φάρμακα - Αρχική σελίδα(α), Σελίδα Προσθήκης φαρμάκου(β), Σελίδα Προσθήκης φαρμάκου συνέχεια(γ),Επιστροφή στο tab Φαρμάκων για επαλήθευση(δ)

Το **τέταρτο tab** σχετίζεται με την πορεία του ασθενή. Σε αυτό το tabδίνονται πληροφορίες σε μορφή expandablelist στο γιατρό για μετρήσεις των ζωτικών σημείων του ασθενή, αποτελέσματα από εργαστηριακές εξετάσεις που είναι έτοιμακαι σχόλια που έχουν γίνει από νοσηλευτές και θα ήταν χρήσιμα να τα γνωρίζει ο γιατρός. Επίσης στο κάτω μέρος της οθόνης εμφανίζεται υπάρχει κουμπί για εμφάνιση του θερμομετρικού διαγράμματος.

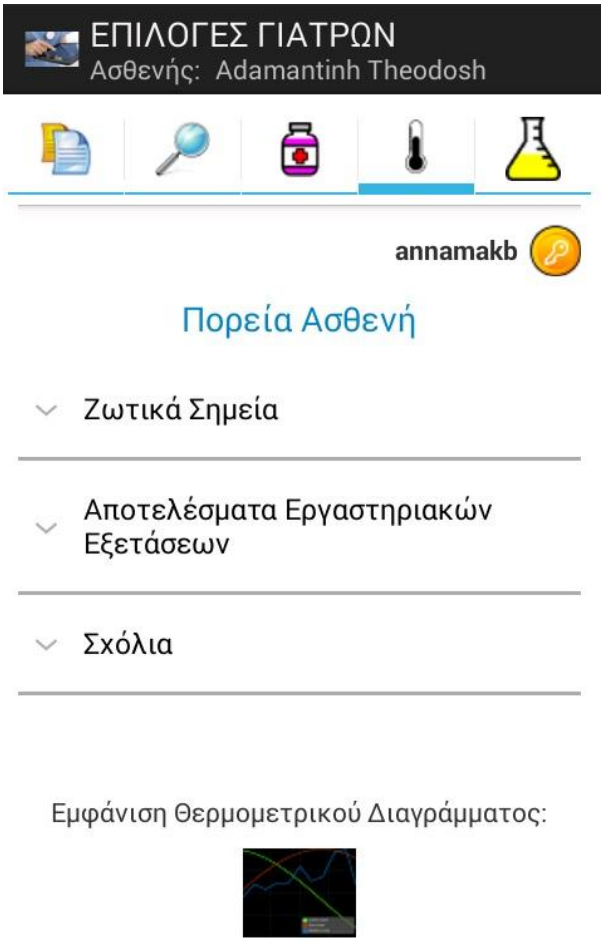
Αν ο γιατρός «ανοίξει» την επιλογή «Ζωτικά Σημεία», μπορεί να δει όλες τις μετρήσεις ζωτικών σημείων που έχουν γίνει στον ασθενή καθ'όλη τη διάρκεια της νοσηλείας του και περιλαμβάνουν τις τιμές πίεσης, αναπνοών, σφυγμών, SpO2(κορεσμός οξυγόνου), ΚΦΠ(κεντρική φλεβική πίεση), κεννώσεων και σακχάρου καθώς και την ακριβή ημ/νία και ώρα που έγινε η μέτρηση. Στα ζωτικά σημεία περιλαμβάνεται και η θερμοκρασία η οποία όμως εμφανίζεται σχηματικά σε θερμομετρικό διάγραμμα και συνεπώς παραλείπεται από τα Ζωτικά σημεία.

Αν «ανοίξει» την επιλογή «Αποτελέσματα Εργαστηριακών Εξετάσεων», εμφανίζονται επίσης σε λίστα τα έτοιμα αποτελέσματα όλων των εξετάσεων που έχει κάνει ο ασθενής, συνοδευόμενες από το εργαστήριο, το όνομα της εξέτασης, τα αποτελέσματα και τις φυσιολογικές τιμές κάθε μέτρησης. Επίσης δίνονται πληροφορίες για την ακριβή ημερομηνία που στάλθηκε το παραπεμπτικό στο εργαστήριο και την ημ/νία που ετοιμάστηκαν τα αποτελέσματα της εξέτασης. Έτσι ο γιατρός μπορεί να ελέγξει την αποδοτικότητα του εργαστηρίου και να εφιστήσει την προσοχή όπου υπάρχουν καθυστερήσεις.Θα πρέπει σε αυτό το σημείο να αναφέρουμε οτι στην κεντρική βάση δεδομένων hospitaléχουμε εισάγει ενδεικτικά 4 εργαστήρια(Αιματολογικό, Μικροβιολογικό, Βιοχημικό και Ακτινολογικό) τα οποία υποστηρίζουν το καθένα από 10 είδη εξετάσεων. Για τα αποτελέσματα του εκάστοτε είδους εξέτασης δημιουργήσαμε στην κεντρική βάση από έναν πίνακα, καθώς κάθε εξέταση έχει τα δικά της πεδία και γι'αυτό δεν μπορούσε να υπάρχει αποκλειστικά ένας πίνακας για να τις καλύπτει όλες. Φτιάξαμε λοιπόν ενδεικτικά 4 πίνακες που αφορούν μία εξέταση από κάθε εργαστήριο(Γενική Αίματος: Αιματολογικό, Ουρία: Βιοχημικό, Αμυλάση Ούρων: Μικροβιολογικό, Μαστογραφία: Ακτινολογικό).

Αν «ανοίξει» την επιλογή «Σχόλια», εμφανίζονται τα σχόλια που έχουν γίνει από τους νοσηλευτές συνοδευόμενα από ακριβή ημ/νίακαι ατο όνομα του νοσηλευτή που έκανε το σχόλιο.

Στην περίπτωση που ο γιατρός πατήσει το κουμπί για εμφάνιση θερμομετρικού διαγράμματος, εμφανίζεται ένα πλήρες διάγραμμα με τις θερμοκρασίες του ασθενή μέχρι και για 5 προηγούμενες μέρες. Κάθε μέρα έχουμε θεωρήσει πως λαμβάνουν χώρα 4 μετρήσεις(ανά 6 ώρες), στις 05:30, 11:30, 17:30

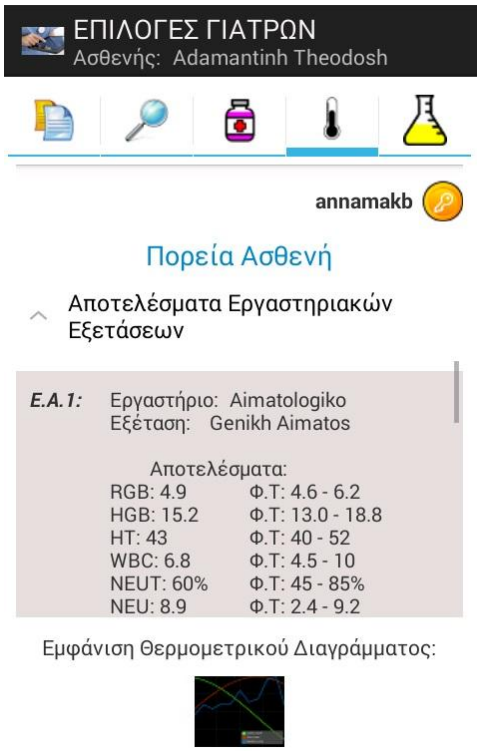
και 23:30. Κάθε γραμμή του θερμομετρικού αφορά διαφορετική μέρα. Με τα πλήκτρα που εμφανίζονται κατω δεξιά ο γιατρός μπορεί να κάνει zoom, ξezoomκαι επιστροφή στο αρχικό μέγεθος. Στο πάνω μέρος της οθόνης εμφανίζεται σχετικός τίτλος που ενημερώνει συνοπτικά και περιεκτικά το χρήστη για το τι ακριβώς βλέπει. Να σημειωθεί επίσης πως ανάλογα με τις μετρήσεις ζωτικών σημείων που έχει ο χρήστης εμφανίζονται και οι ανάλογες μέρες, δηλαδή αν υπάρχουν μετρήσεις για 4,3,2,1 προηγούμενες μέρες θα εμφανιστούν οι αντίστοιχες 4,3,2,1 γραμμές στο θερμομετρικό διάγραμμα. Τέλος σε περίπτωση που ο χρήστης δεν έχει μετρησεις για τουλάχιστον ένα 24ωρο, εμφανίζεται μία γραμμή ευθεία στους 33 βαθμούς και ανάλογο toastπου ενημερώνει το χρήστη για το λόγο που δεν παράγεται το διάγραμμα.

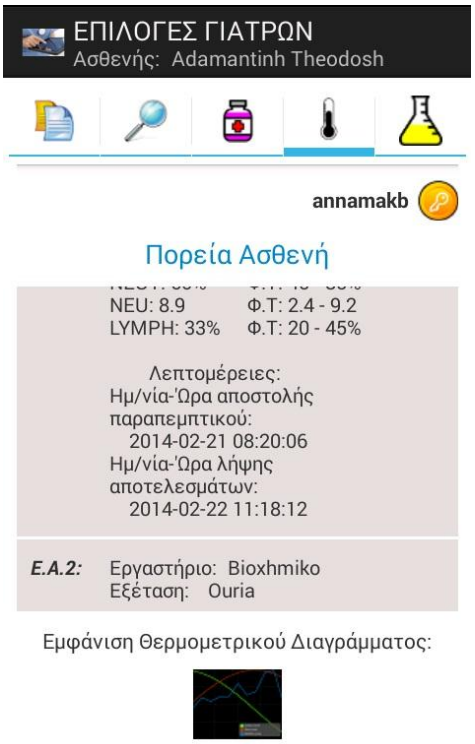


Εικόνα 103: Progress Tab

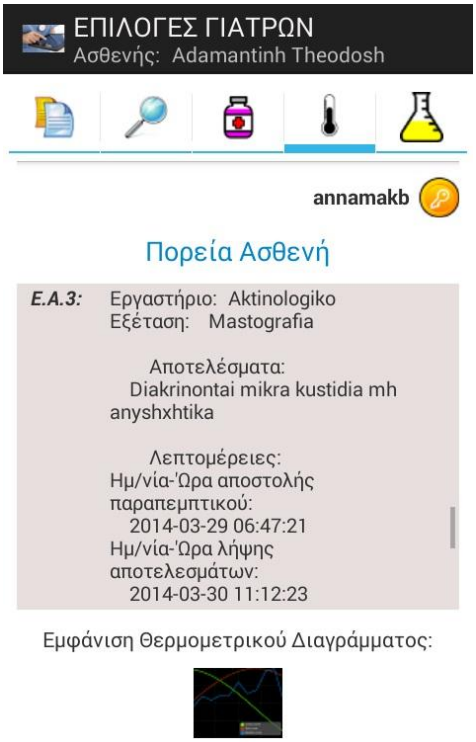
ΕΠΙΛΟΓΕΣ ΓΙΑΤΡΩΝ
Ασθενής: Adamantinh Theodosh
annamakb
Πορεία Ασθενή
Ζωτικά Σημεία
Ζ.Σ.1: Ημ/νία - 'Ωρα Μέτρησης:
2014-04-28 05:30:00
Πίεση: 110
Αναπνοές: 78
Σφυγμοί:90
SpO2: 0.97
ΚΦΠ: 24
Κεννώσεις: 1
Σάκχαρο: 105
Εμφάνιση Θερμομετρικού Διαγράμματος:

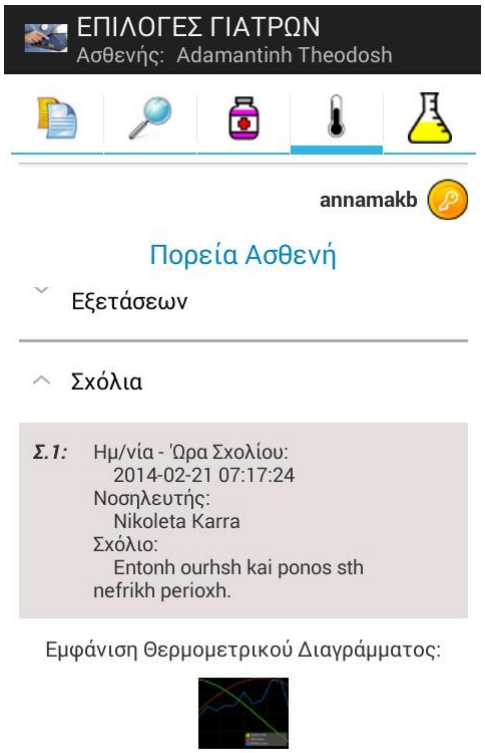
ΕΠΙΛΟΓΕΣ ΓΙΑΤΡΩΝ
Ασθενής: Adamantinh Theodosh
annamakb
Πορεία Ασθενή
Ζ.Σ.2: Ημ/νία - 'Ωρα Μέτρησης:
2014-04-28 11:30:00
Πίεση: 105
Αναπνοές: 80
Σφυγμοί:93
SpO2: 0.98
ΚΦΠ: 25
Κεννώσεις: 2
Σάκχαρο: 107
Ζ.Σ.3: Ημ/νία - 'Ωρα Μέτρησης:
Εμφάνιση Θερμομετρικού Διαγράμματος:


ΕΠΙΛΟΓΕΣ ΓΙΑΤΡΩΝ
Ασθενής: Adamantinh Theodosh
annamakb
Πορεία Ασθενή
Αποτελέσματα Εργαστηριακών Εξετάσεων
Ε.Α.1: Εργαστήριο: Aimatologiko
Εξέταση: Genikh Aimatos
Αποτελέσματα:
RGB: 4.9 Φ.Τ: 4.6 - 6.2
HGB: 15.2 Φ.Τ: 13.0 - 18.8
HT: 43 Φ.Τ: 40 - 52
WBC: 6.8 Φ.Τ: 4.5 - 10
NEUT: 60% Φ.Τ: 45 - 85%
NEU: 8.9 Φ.Τ: 2.4 - 9.2
Εμφάνιση Θερμομετρικού Διαγράμματος:

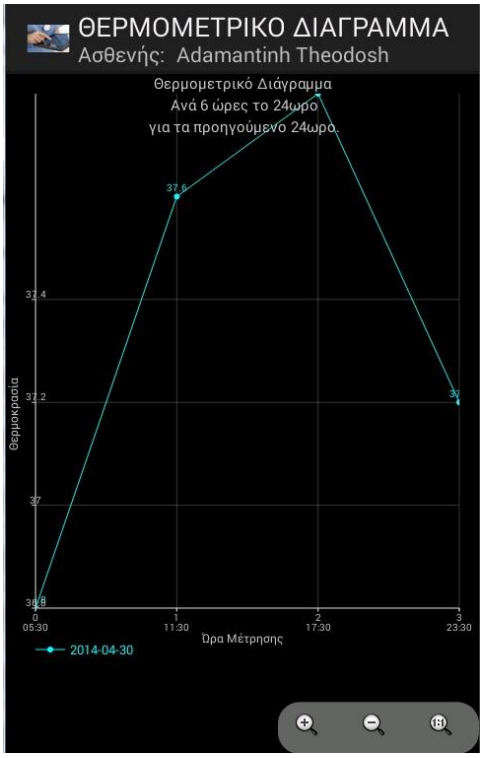

ΕΠΙΛΟΓΕΣ ΓΙΑΤΡΩΝ
Ασθενής: Adamantinh Theodosh
annamakb
Πορεία Ασθενή
NEU: 8.9 Φ.Τ: 2.4 - 9.2
LYMPH: 33% Φ.Τ: 20 - 45%
Λεπτομέρειες:
Ημ/νία-'Ωρα αποστολής παραπεμπτικού:
2014-02-21 08:20:06
Ημ/νία-'Ωρα λήψης αποτελεσμάτων:
2014-02-22 11:18:12
Ε.Α.2: Εργαστήριο: Bioxhmiko
Εξέταση: Ouria
Εμφάνιση Θερμομετρικού Διαγράμματος:

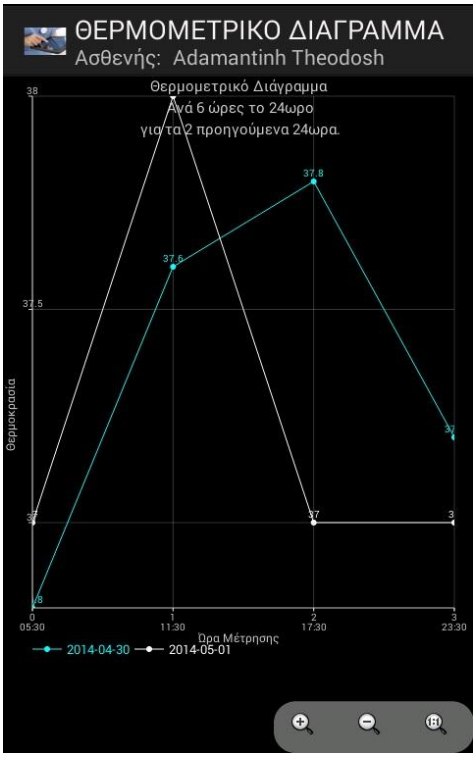
ΕΠΙΛΟΓΕΣ ΓΙΑΤΡΩΝ
Ασθενής: Adamantinh Theodosh
annamakb
Πορεία Ασθενή
Ε.Α.2: Εργαστήριο: Bioxhmiko
Εξέταση: Ouria
Αποτελέσματα:
Ουρία: 20 Φ.Τ: 10-50 mg/dl
Λεπτομέρειες:
Ημ/νία-Ώρα αποστολής παραπεμπτικού:
2014-03-03 08:25:18
Ημ/νία-Ώρα λήψης αποτελεσμάτων:
2014-03-04 06:26:23
Εμφάνιση Θερμομετρικού Διαγράμματος:

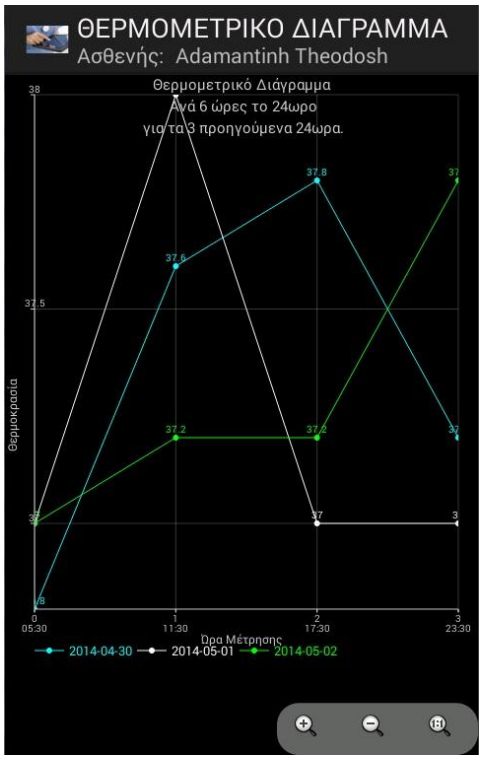

ΕΠΙΛΟΓΕΣ ΓΙΑΤΡΩΝ
Ασθενής: Adamantinh Theodosh
annamakb
Πορεία Ασθενή
Ε.Α.3: Εργαστήριο: Aktinologiko
Εξέταση: Mastografia
Αποτελέσματα:
Diakrinontai mikra kustidia mh anyshxhtika
Λεπτομέρειες:
Ημ/νία-Ώρα αποστολής παραπεμπτικού:
2014-03-29 06:47:21
Ημ/νία-Ώρα λήψης αποτελεσμάτων:
2014-03-30 11:12:23
Εμφάνιση Θερμομετρικού Διαγράμματος:

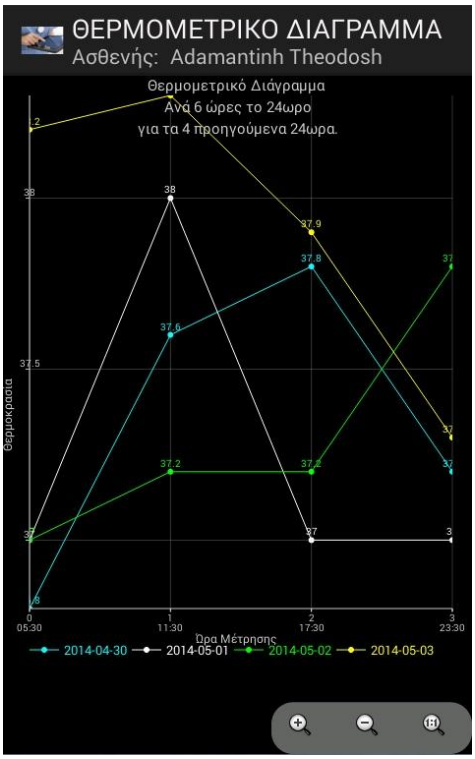

ΕΠΙΛΟΓΕΣ ΓΙΑΤΡΩΝ
Ασθενής: Adamantinh Theodosh
annamakb
Πορεία Ασθενή
Εξετάσεων
Σχόλια
Σ.1: Ημ/νία - Ώρα Σχολίου:
2014-02-21 07:17:24
Νοσηλευτής:
Nikoleta Karra
Σχόλιο:
Entonh ourhsh kai ponos sth nefrikh perioxh.
Εμφάνιση Θερμομετρικού Διαγράμματος:

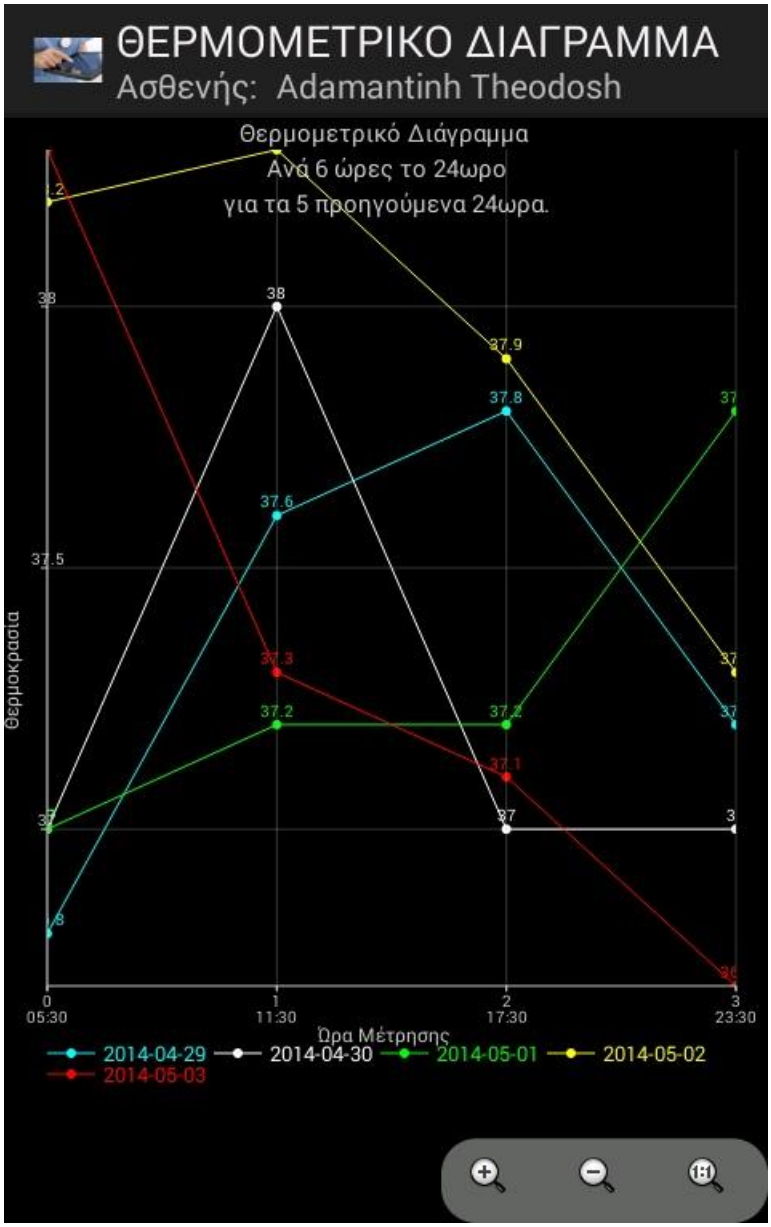
ΘΕΡΜΟΜΕΤΡΙΚΟ ΔΙΑΓΡΑΜΜΑ
Ασθενής: Adamantinh Theodosh
Θερμομετρικό Διάγραμμα
Ανά 6 ώρες το 24ωρο
για τα προηγούμενο 24ωρο.
Δεν μπορεί να δημιουργηθεί θερμομετρικό διάγραμμα, διότι δεν παρέχονται τουλάχιστον 4 μετρήσεις(ένα 24ωρο).


ΘΕΡΜΟΜΕΤΡΙΚΟ ΔΙΑΓΡΑΜΜΑ
Ασθενής: Adamantinh Theodosh
Θερμομετρικό Διάγραμμα
Ανά 6 ώρες το 24ωρο
για τα προηγούμενο 24ωρο.
Θερμοκρασία
Ώρα Μέτρησης
2014-04-30


ΘΕΡΜΟΜΕΤΡΙΚΟ ΔΙΑΓΡΑΜΜΑ
Ασθενής: Adamantinh Theodosh
Θερμομετρικό Διάγραμμα
Ανά 6 ώρες το 24ωρο
για τα 2 προηγούμενα 24ωρα.
Θερμοκρασία
Ώρα Μέτρησης
2014-04-30
2014-05-01

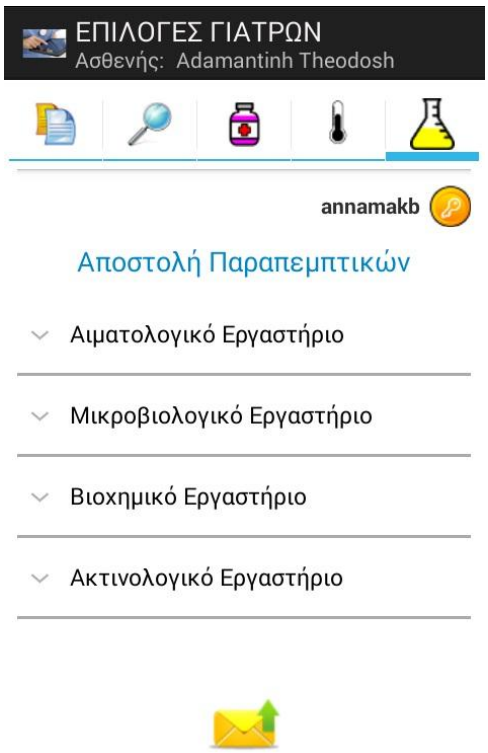

ΘΕΡΜΟΜΕΤΡΙΚΟ ΔΙΑΓΡΑΜΜΑ
Ασθενής: Adamantinh Theodosh
Θερμομετρικό Διάγραμμα
Ανά 6 ώρες το 24ωρο
για τα 3 προηγούμενα 24ωρα.
Θερμοκρασία
Ώρα Μέτρησης
2014-04-30
2014-05-01
2014-05-02

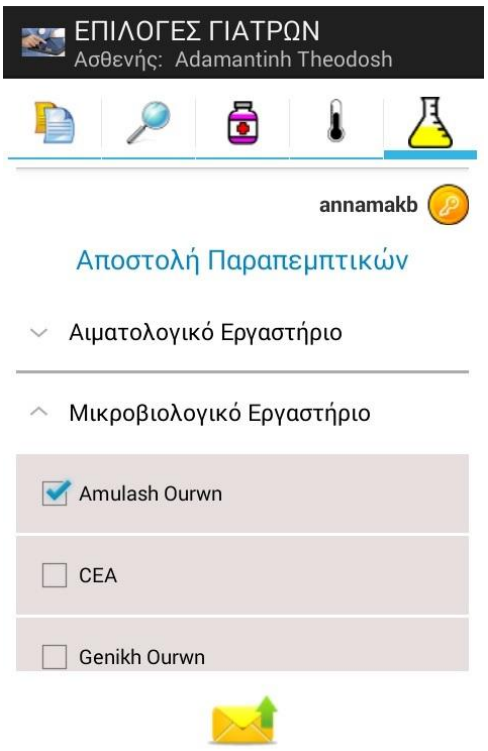

ΘΕΡΜΟΜΕΤΡΙΚΟ ΔΙΑΓΡΑΜΜΑ
Ασθενής: Adamantinh Theodosh
Θερμομετρικό Διάγραμμα
Ανά 6 ώρες το 24ωρο
για τα 4 προηγούμενα 24ωρα.
Θερμοκρασία
Ώρα Μέτρησης
2014-04-30
2014-05-01
2014-05-02
2014-05-03

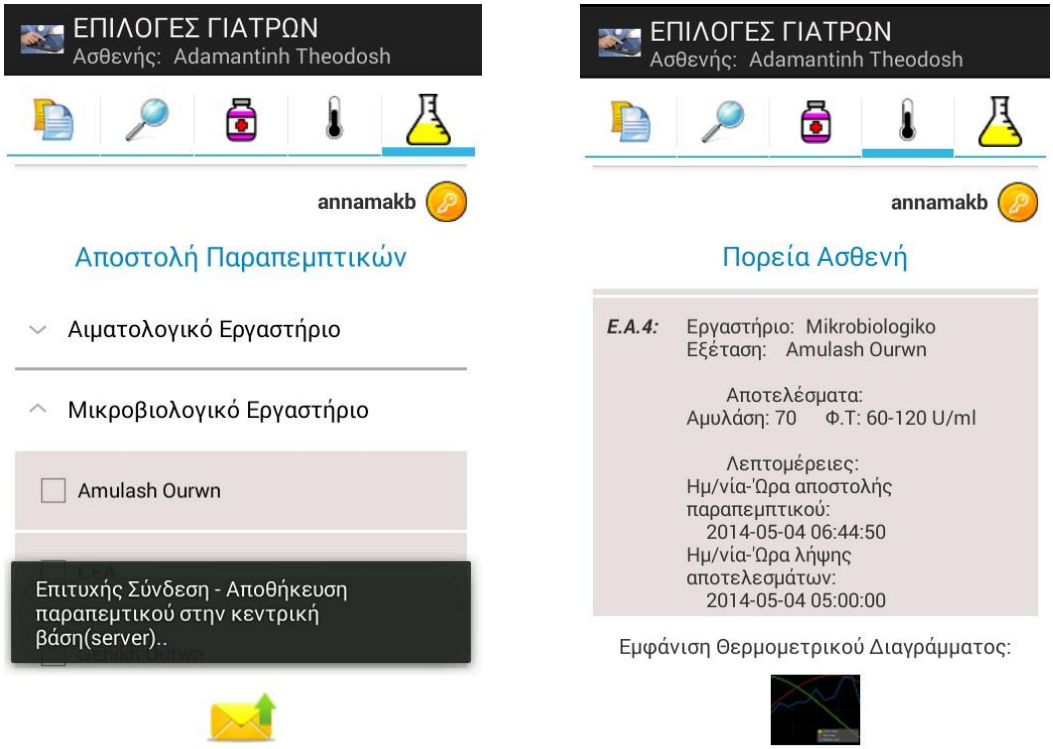


Εικόνα 104: Ζωτικά Σημεία(α) -Ζωτικά Σημεία συνέχεια(β) -Γενική Αίματος(γ) - Γενική Αίματος συνέχεια(δ) - Ουρία(ε) - Μαστογραφία(στ)- Σχόλια(ζ) - Θερμ/κό Διάγραμμα με σφάλμα(η) - Θερμ/κό Διάγραμμα για 1 μέρα(θ) - Θερμ/κό Διάγραμμα για 2 μέρες(θ) - Θερμ/κό Διάγραμμα για 3 μέρες(θ) - Θερμ/κό Διάγραμμα για 4 μέρες(θ) - Θερμ/κό Διάγραμμα για 5 μέρες(θ)

Το **πέμπτο tab**, σχετίζεται με την αποστολή παραπεμπτικών στα εργαστήρια προς διεξαγωγή εξετάσεων. Με χρήση expandablelistεμφανίζονται κατηγοριοποιημένες ανά εργαστήριο οι εξετάσεις για τις οποίες ο χρήστης μπορεί να στείλει παραπεμπτικό-ά. Στην περίπτωσή μας θα δοκιμάσουμε να στείλουμε μια παραπεμπτικό για μια εξέταση αμυλάσης ούρων, η οποία ανήκει στο Μικροβιολογικό εργαστήριο. Ο γιατρός θα τικάρει την επιλογή του και κατόπιν θα πατήσει το εικονίδιο φακελάκι στο τέλος της σελίδας, ώστε να σταλεί το παραπεμπτικό στην κεντρική βάση. Με το πάτημα του κουμπιού εμφανίζεται toastπρος ενημέρωση του χρήστη για την αποστολή παραπεμπτικού. Στη συνέχεια, παίζουμε το ρόλο του εργαστηρίου και συμπληρώνουμε στον πίνακα labresultex3 της κεντρικής βάσης hospitalτα πεδία «AmylashOurwn», «Status» και «DateReady»με τις κατάλληλες τιμές. Επίσης στον πίνακα κάνουμε το statustou συγκεκριμένου παραπεμπτικού 1, ώστε να φανεί πως η εξέταση είναι έτοιμη. Κατόπιν προς επαλήθευση πάμε στο progresstabκαι ανοίγουμε τα «Αποτελέσματα Εργαστηριακών Εξετάσεων», στο οποίο βλέπουμε πως όντως έχουν προστεθεί τα αποτελέσματα της Αμυλάσης Ούρων με τα στοιχεία που συμπληρώθηκαν από το εργαστήριο.





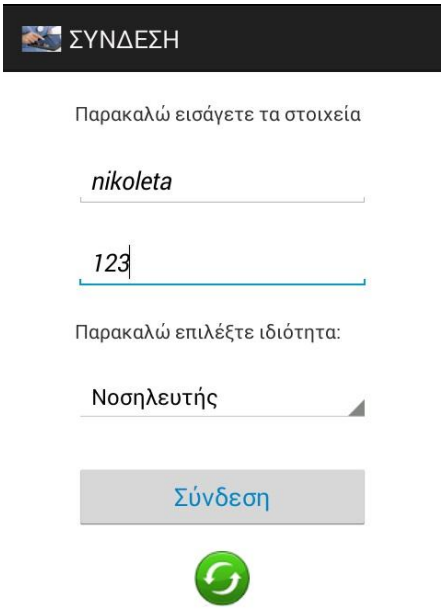


Εικόνα 105: ReferenceTab(α), Επιλογή Εξέτασης(β), Ενημερωτικό toast στο πάτημα του φακέλου(γ), Επαλήθευση(δ)

6.2. Οθόνες Νοσηλευτή

Στην ενότητα αυτή δοκιμάζουμε να τρέξουμε την εφαρμογή ως **νοσηλευτής**με στοιχεία «nikoleta» (username) & «123» (password). Αφότου ο χρήστης δώσει τα συγκεκριμένα στοιχεία γίνεται αίτηση στο server μέσω .php αρχείου για αναζήτηση νοσηλευτή με τα συγκεκριμένα στοιχεία.Σε περίπτωση που δεν βρεθεί νοσηλευτής με αυτά τα στοιχεία επιστρέφεται μήνυμα λάθους, το οποίο εμφανίζεται με μορφή toast στο χρήστη. Γνωρίζουμε ως developertου συστήματος πως τα στοιχεία που πληκτρολόγησε ο χρήστης είναι έγκυρα και με το κουμπί σύνδεση μεταβαίνουμε στην επόμενη σελίδα. Για να δείξουμε και τι εμφανίζεται σε περίπτωση μη έγκυρων στοιχείων πληκτρολογούμε κάποια λάθος και εμφανίζουμε την αντίστοιχη οθόνη.

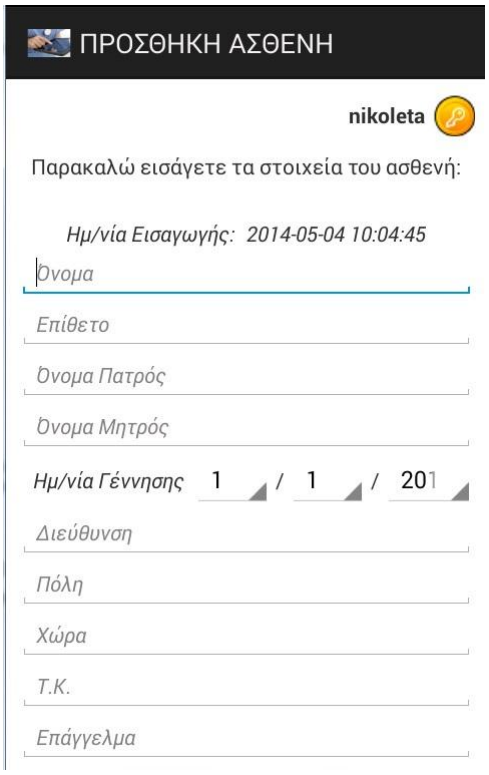
Σε περίπτωση έγκυρου loginη οθόνη που μεταβαίνει ο νοσηλευτής τον ενημερώνει για τις 2 βασικές λειτουργίες που μπορεί να εκτελέσει(εισαγωγή νέου ασθενή και εύρεση ασθενή από τους υπάρχοντες). Οι δύο αυτές λειτουργίες εμφανίζονται στην οθόνη με μορφή κουμπιών(imagebutton) και ανάλογαμε το τι επιθυμεί να κάνει ο χρήστης πατάει και το ανάλογο κουμπί. Να σημειωθεί επίσης πως πάνω δεξιά εμφανίζεται το usernameμε το οποίο εισήχθη ο νοσηλευτής και δίπλα του ένα κουμπάκι αποσύνδεσης από το σύστημα. Σε περίπτωση αποσύνδεσης, ο χρήστης μεταβαίνει στη σελίδα Loginπου περιγράψαμε πριν. Τα 2 αυτά στοιχεία (username&κουμπί αποσύνδεσης) θα υπάρχουν και στις επόμενες οθόνες που ακολουθούν έχοντας ακριβώς την ίδια λειτουργικότητα, δηλαδή μετάβαση στο LoginPage.

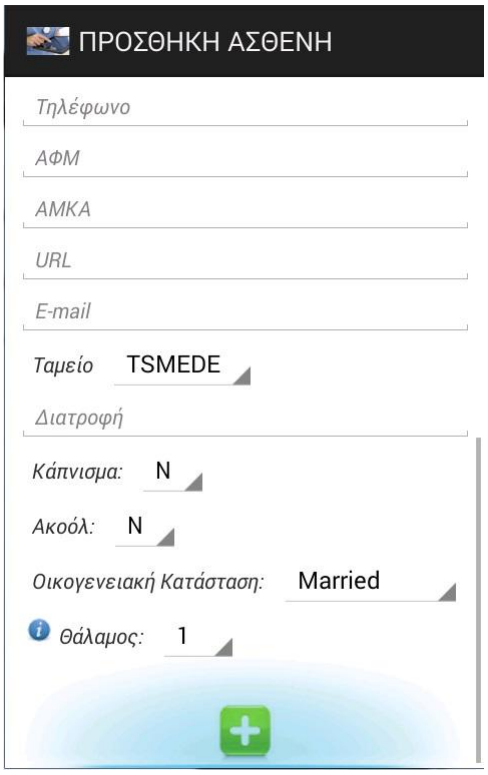


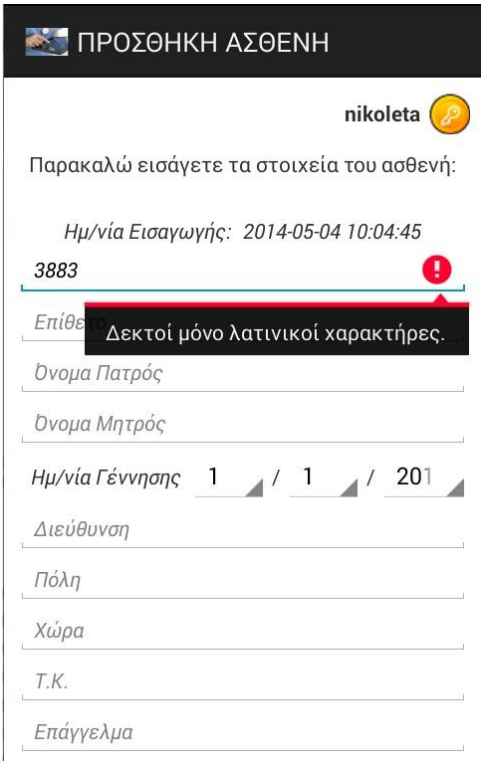
Εικόνα106: ΜηέγκυροLogin: Παραμονήστο Login Page(α), Έγκυρο Login: Μετάβασηστησελίδαβασικώνεπιλογώννοσηλευτή(β), Basic Nurse Page(γ)

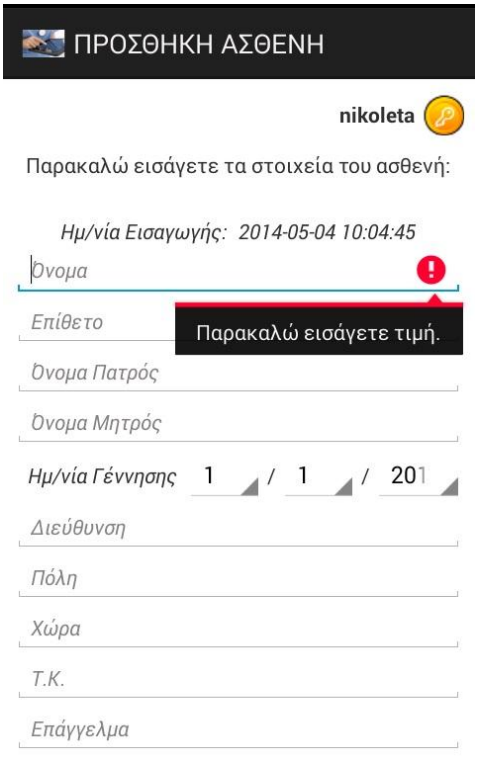
Ο νοσηλευτης κατόπιν πατάει το κουμπί για **εισαγωγή νέου ασθενή** και μεταβαίνει στην επόμενη σελίδα κατά την οποία συμπληρώνει από το πληκτρολόγιο μια φόρμα με τα προσωπικά στοιχεία του νέου ασθενή. Σε αυτή τη σελίδα σε κάθε πεδίο υπάρχουν έλεγχοι για την σωστή εισαγωγή δεδομένων. Για παράδειγμα αν στο πεδίο όνομα εισάγει αριθμητικά δεδομένα πετιέται προειδοποιητικό μήνυμα που τον προτρέπει να εισάγει λατινικούς χαρακτήρες. Ο λόγος για τον οποίο πρέπει να εισαχθούν λατινικοί χαρακτήρες έχει να κάνει με την κεντρική βάση δεδομένων στο server, η οποία έχουμε ήδη αναφέρει πως δεν δέχεται ελληνικά. Αν επίσης δεν εισάγει τιμή στο πεδίο αυτό πετιέται επίσης ενημερωτικό μήνυμα. Με τον ίδιο τρόπο λειτουργεί η εφαρμογή και για τα υπόλοιπα πεδία της φόρμας συμπλήρωσης στοιχείων. Επίσης στο πεδίο, για παράδειγμα του ταχυδρομικού κώδικα, θα πρέπει να εισαχθεί μόνο αριθμητική τιμή, πράγμα που εξασφαλίζεται με ενημερωτικό μήνυμα λάθους. Παρόμοιο μήνυμα πετιέται σε όλα τα πεδία τα οποία πρέπει να συμπληρωθούν από αριθμούς. Επίσης με spinnereισάγονται τιμές στα πεδία φύλο, ταμείο, κάπνισμα, αλκοόλ,

οικογενειακή κατάσταση και θάλαμος. Στο πεδίο url επίσης υπάρχει σχετικό μήνυμα που ενημερώνει το νοσηλευτή για τη μορφή που θα πρέπει να έχει η τιμή που εισάγει. Το ίδιο συμβαίνει και στο πεδίο της ηλεκτρονικής διεύθυνσης e-mail. Επίσης πρέπει να αναφέρουμε πως αριστερά του πεδίου θάλαμος, υπάρχει ένα κουμπί πληροφοριών, το οποίο αν πατηθεί από το χρήστη πετάει ενημερωτικό toast για τους θαλάμους που αντιστοιχούν σε κάθε κλινική, ώστε να διευκολύνεται ο χρήστης. Αφού συμπληρωθεί η φόρμα ο νοσηλευτής πατάει το κουμπί + στο τέλος της σελίδας και ο νέος ασθενής εισέρχεται στην τοπική βάση δεδομένων. Όπως περιγράψαμε σε προηγούμενο κεφάλαιο για πρακτικούς λόγους θεωρήσαμε πως για τους νοσηλευτές η εφαρμογή θα λειτουργήσει όπως θα λειτουργούσε αν δεν υπήρχε επιτυχία σύνδεσης στο serverκαι γι’ αυτό το λόγο όλα τα δεδομένα που δίνονται από το νοσηλευτή θερούμε πως αποθηκεύονται τοπικά. Προκειμένου να αντιλαμβάνεται και ο ίδιος ο χρήστης τι ακριβώς συμβαίνει, όταν πατάει το κουμπί +, πετιέται ενημερωτικό μήνυμα για αποτυχία σύνδεσης και εισαγωγή στην τοπική βάση.

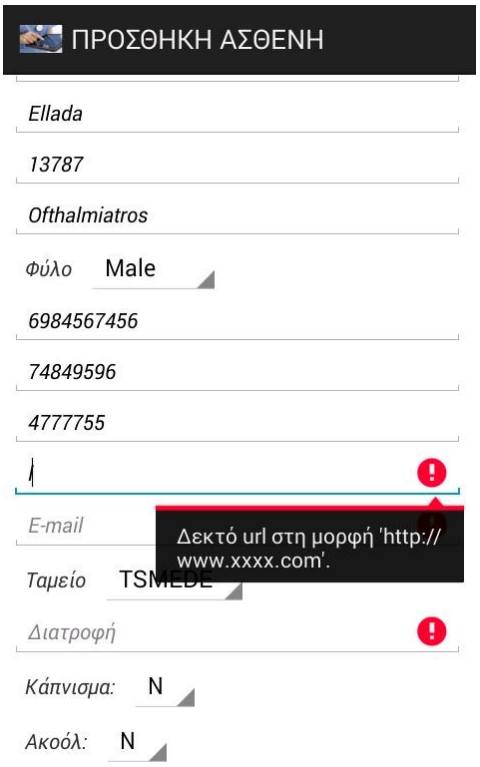


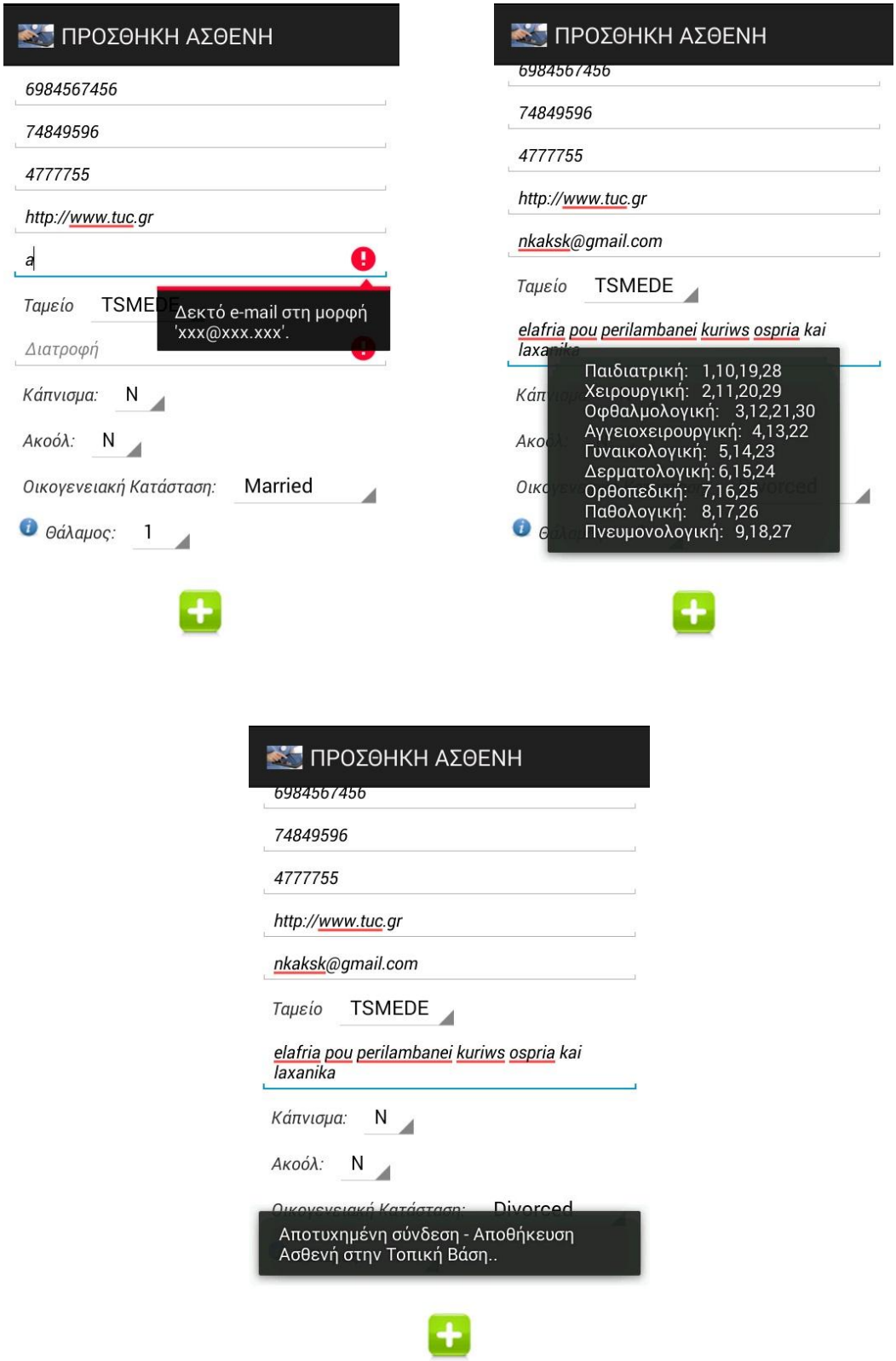



ΠΡΟΣΘΗΚΗ ΑΣΘΕΝΗ
nikoleta
Παρακαλώ εισάγετε τα στοιχεία του ασθενή:
Ημ/νία Εισαγωγής: 2014-05-04 10:04:45
3883
Δεκτοί μόνο λατινικοί χαρακτήρες.
Όνομα Πατρός
Όνομα Μητρός
Ημ/νία Γέννησης 1 / 1 / 201
Διεύθυνση
Πόλη
Χώρα
Τ.Κ.
Επάγγελμα


ΠΡΟΣΘΗΚΗ ΑΣΘΕΝΗ
nikoleta
Παρακαλώ εισάγετε τα στοιχεία του ασθενή:
Ημ/νία Εισαγωγής: 2014-05-04 10:04:45
Όνομα
Επίθετο
Παρακαλώ εισάγετε τιμή.
Όνομα Πατρός
Όνομα Μητρός
Ημ/νία Γέννησης 1 / 1 / 201
Διεύθυνση
Πόλη
Χώρα
Τ.Κ.
Επάγγελμα

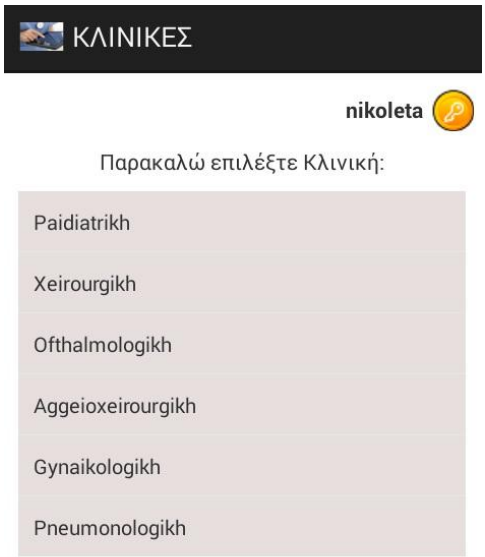
ΠΡΟΣΘΗΚΗ ΑΣΘΕΝΗ
Παρακαλώ εισάγετε τα στοιχεία του ασθενή:
Ημ/νία Εισαγωγής: 2014-05-04 10:04:45
Dionyshs
Papadatos
Theodwros
Anastasia
Ημ/νία Γέννησης 6 / 5 / 196
Anthemiou
Patra
Ellada
lat
Δεκτοί μόνο αριθμητικοί χαρακτήρες.
Επάγγελμα
Φύλο Female

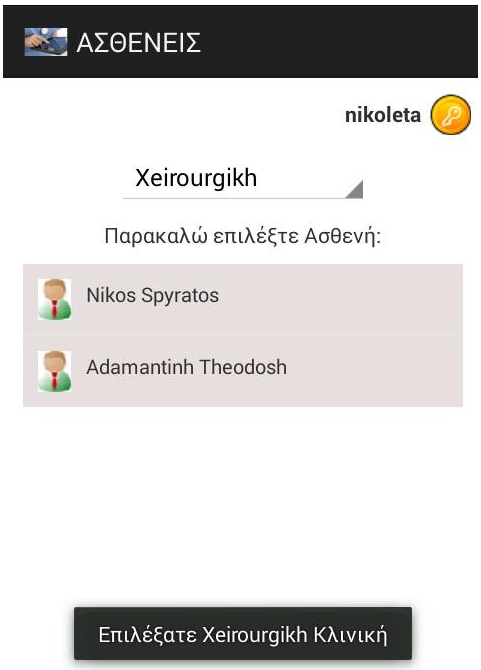

ΠΡΟΣΘΗΚΗ ΑΣΘΕΝΗ
Ellada
13787
Ofthalmiatros
Φύλο Male
6984567456
74849596
4777755
E-mail
Δεκτό url στη μορφή 'http://www.xxxx.com'.
Ταμείο
Διατροφή
Κάπνισμα: N
Ακοόλ: N

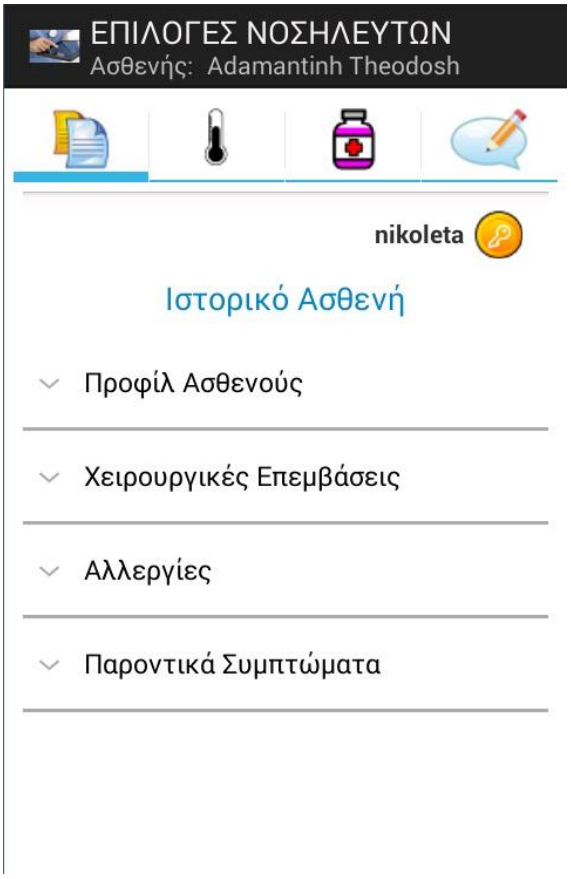


Εικόνα107: Add new patient page(α), Add new patient page συνέχεια(β), Έλεγχοιεγκυρότητας(γ,δ,ε,στ,ζ), Πλήκτρογιαπληροφορίεςκλινικών(η), Προσθήκηνέουασθενή(θ)

Στη συνέχεια, ο χρήστης γυρνάει στην προηγούμενη σελίδα των βασικών επιλογών του νοσηλευτή και πατάει το κουμπί Εύρεση ασθενή. Μεταβαίνει στη σελίδα των κλινικών(ClinicsPage), ο οποία είναι ακριβώς ίδια με τη σελίδα των κλινικών που εμφανίζεται στους γιατρούς και περιγράψαμε πιο πάνω.Ο νοσηλευτής στην περιπτωσή μας θεωρούμε πως επιλέγει χειρουργική κλινική και από τους ασθενείς που νοσηλεύονται σε αυτή επιλέγει την AdamantiniTheodosi. Αφότου την επιλέξει η εφαρμογή τον πεταέι στην επόμενη σελίδα, στην οποία με παρόμοιο τρόπο με τους γιατρούς, εμφανίζονται 4 διαφορετικά tabsγια τις δυνατότητες του νοσηλευτή. Να σημειωθεί πως και σε αυτή τη σελίδα είναι προεπιλεγμένο το **πρώτο tab**, που σχετίζεται με το ιστορικό του ασθενή και έχει την ίδια ακριβώς λειτουργία με αυτή που έχει και για τους γιατρούς. Ο νοσηλευτής μπορεί μέσω αυτού να ενημερωθεί για το ιστορικό του ασθενή.

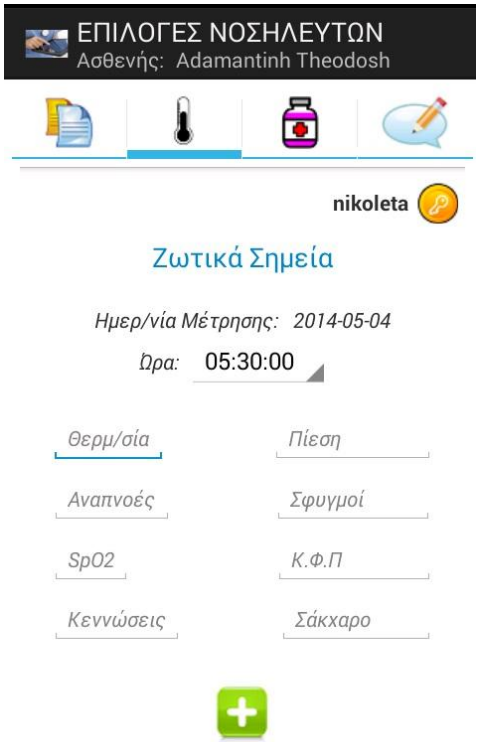


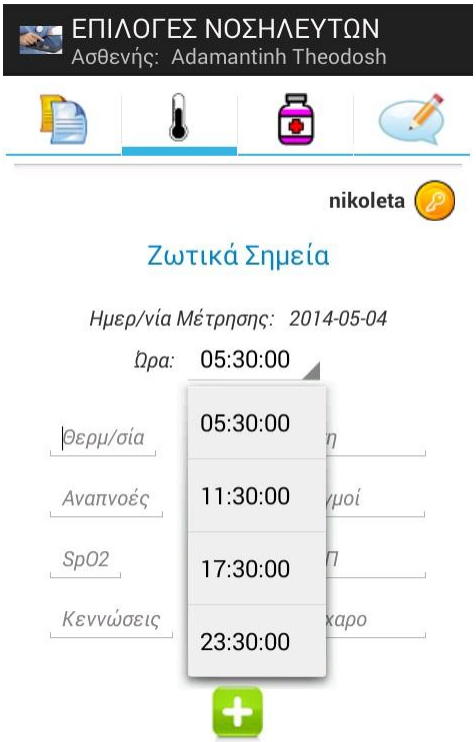


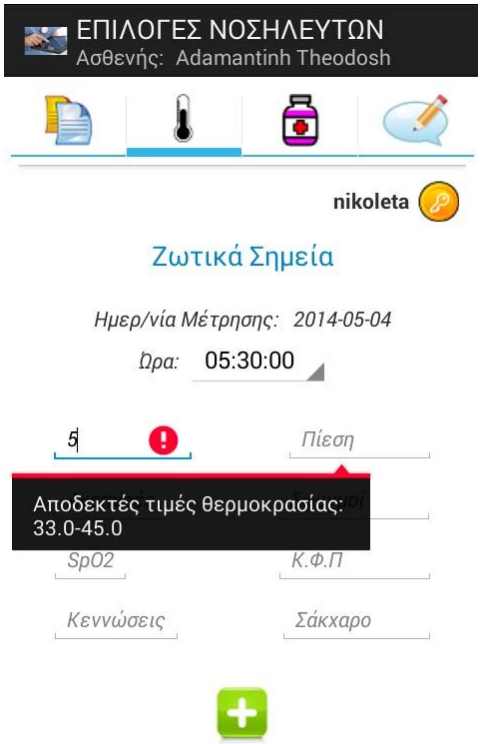


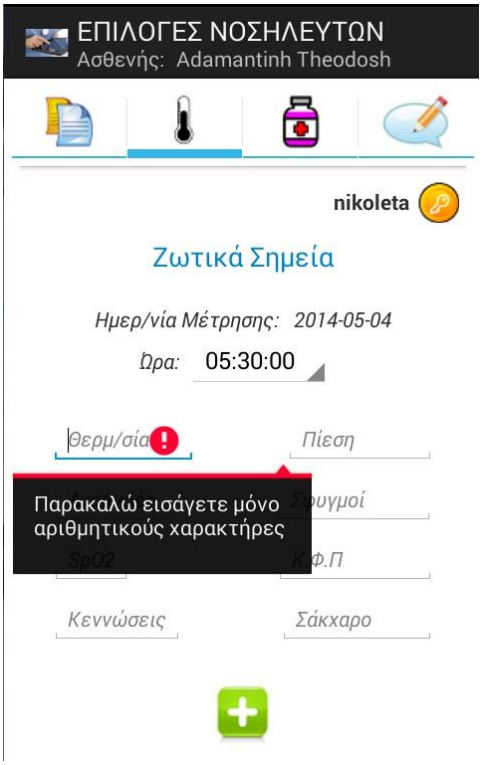
Εικόνα 108: Clinics Page(α), Patients Page(β), Nurse Tabs(γ)

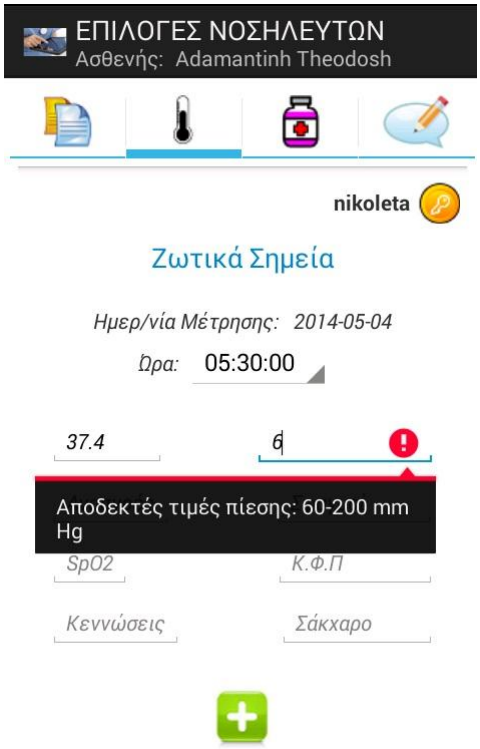
Κατόπιν, στο **δεύτερο tab**, ο νοσηλευτής έχει τη δυνατότητα να εισάγει νέες μετρήσεις ζωτικών σημείων για τον επιλεγμένο ασθενή. Στο πάνω μέρος της οθόνης λαμβάνεται αυτόματα η ημερομηνία και ο νοσηλευτής μπορεί να εισάγει μέσω spinnerτην ώρα (διαθέσιμες: 05:30, 11:30, 17:30, 23:30). Ο λόγος που η εφαρμογή τον «αναγκάζει» να επιλέξει μία από αυτές τις ώρες είναι καθαρά πρακτικός, καθώς οι μετρήσεις ζωτικών πρέπει να λαμβάνονται συγκεκριμένες ώρες ώστε να μπορεί να δημιουργηθεί και το θερμομετρικό διάγραμμα. Κατά τη συμπλήρωση των πεδίων των ζωτικών σημείων, υπάρχουν μηνύματα εγκυρότητας που ενημερώνουν το χρήστη για τις δυνατές τιμές που μπορεί να δώσει για κάθε μέτρηση. Τέλος, πατώντας το κουμπί +, η νέα μέτρηση εισάγεται στην τοπική βάση.

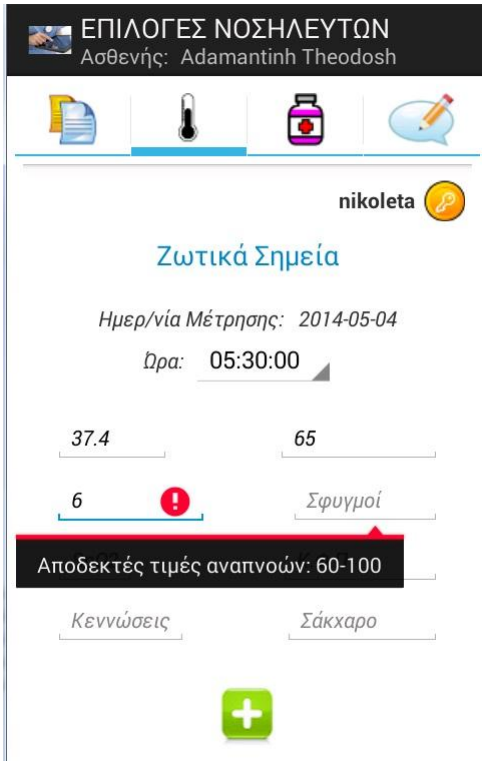

ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
Θερμ/σία
Πίεση
Αναπνοές
Σφυγμοί
SpO2
Κ.Φ.Π
Κεννώσεις
Σάκχαρο


ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
05:30:00
11:30:00
17:30:00
23:30:00
Θερμ/σία
Αναπνοές
SpO2
Κεννώσεις


ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
5
Πίεση
Αποδεκτές τιμές θερμοκρασίας: 33.0-45.0
SpO2
Κ.Φ.Π
Κεννώσεις
Σάκχαρο


ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
Θερμ/σία
Πίεση
Παρακαλώ εισάγετε μόνο αριθμητικούς χαρακτήρες
Κεννώσεις
Σάκχαρο


ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
37.4
6
Αποδεκτές τιμές πίεσης: 60-200 mm Hg
SpO2
Κ.Φ.Π
Κεννώσεις
Σάκχαρο

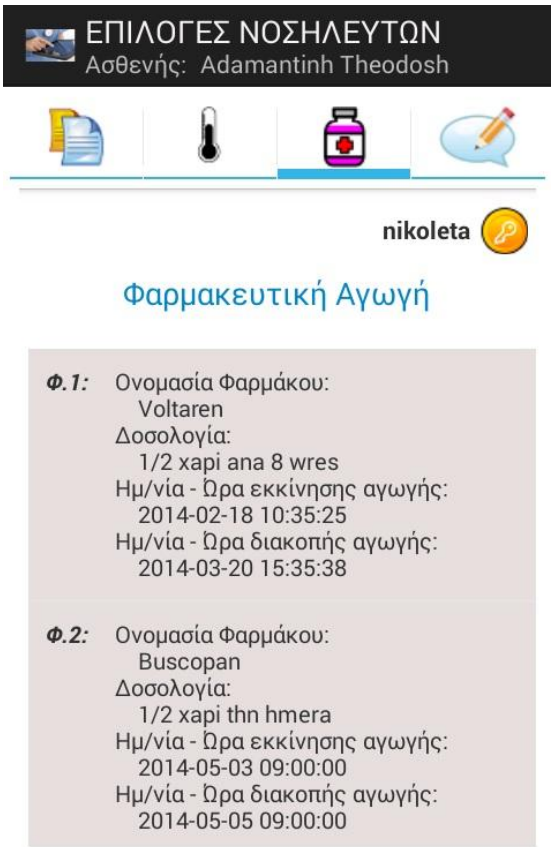

ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
37.4
65
6
Σφυγμοί
Αποδεκτές τιμές αναπνοών: 60-100
Κεννώσεις
Σάκχαρο

ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
37.4
65
65
6
Αποδεκτές τιμές σφυγμών: 70-120
Κεννώσεις
Σάκχαρο

ΕΠΙΛΟΓΕΣ ΝΟΣΗΛΕΥΤΩΝ
Ασθενής: Adamantinh Theodosh
nikoleta
Ζωτικά Σημεία
Ημερ/νία Μέτρησης: 2014-05-04
Ώρα: 05:30:00
37.4
65
65
78
0.
Κ.Φ.Π
Αποδεκτές τιμές SpO2: 0.7-1.0

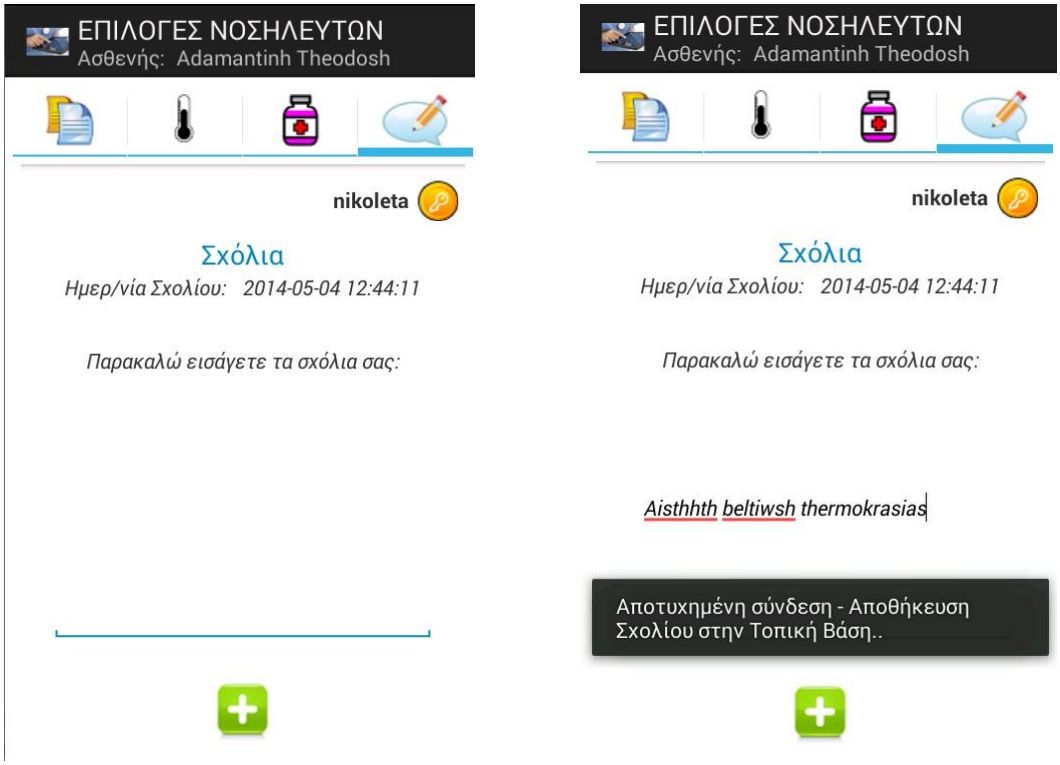
Εικόνα 109: VitalsignsTab(α), spinner για ώρα(β), Μηνύματα εγκυρότητας(γ,δ,ε,στ,ζ,η,θ,ι), Προσθήκη Μέτρησης(κ)

Στο **τρίτο tab**, ο νοσηλευτής έχει πρόσβαση στη φαρμακευτική αγωγή του επιλεγμένου ασθενούς. Ως νοσηλευτής μπορεί μόνο να την αναγνώσει και σύμφωνα με αυτή να χορηγήσει τα κατάλληλα φάρμακα στον ασθενή.



Εικόνα 110: MedicineTab

Στο **τέταρτο** και τελευταίο **tab**, ο νοσηλευτής εισάγει κάποιο σχόλιο για τον επιλεγμένο ασθενή που ενδεχομένως θα φανεί χρήσιμο για τον γιατρό. Στο πάνω μέρος της οθόνης αναγράφεται η ημερομηνία και ώρα σχολίου. Αφού συμπληρωθεί το πεδίο σχολίου, ο νοσηλευτής πατάει το κουμπί + στο κάτω μέρος της οθόνης, μέσω του οποίου το νέο σχόλιο εισάγεται στην τοπική βάση. Παράλληλα, ο χρήστης όπως και στις προηγούμενες περιπτώσεις ενημερώνεται για την αποτυχία σύνδεσης με serverκαι συνεπώς για την τοπική αποθήκευση.



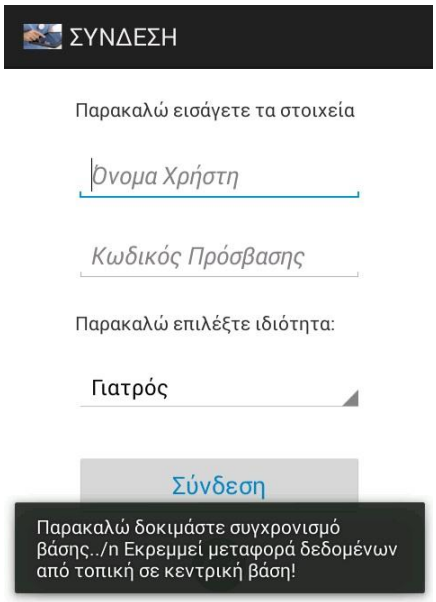
Εικόνα 111: CommentTab(α), Προσθήκη Σχολίου(β)

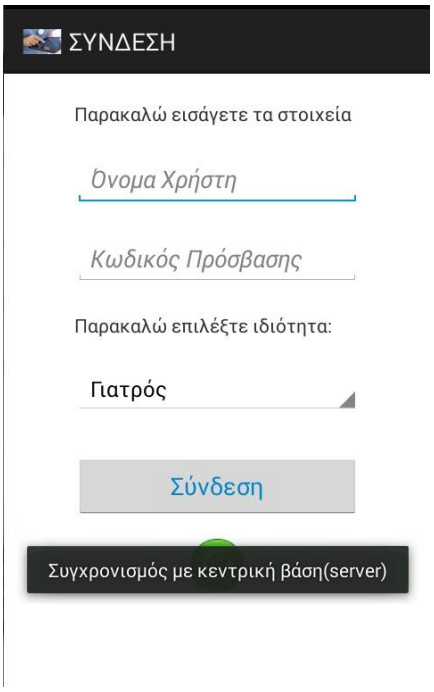
Σε αυτό το σημείο ο νοσηλευτής έχει εισάγει έναν νέο ασθενή, μία μέτρηση ζωτικών σημείων για την ασθενή AdamantiniTheodosikαι ένα σχόλιο για την ίδια ασθενή. Τα συγκεκριμένα όμως δεδομένα δεν έχουν εισαχθεί στο serverπαρά μόνο στην τοπική βάση. Για το λόγο αυτό, όταν ο χρήστης επιχειρήσει να αποσυνδεθεί από το σύστημα μέσω του πορτοκαλί κουμπιού πάνω δεξιά, θα μεταβεί στη σελίδα Loginκαι ταυτόχρονα θα πεταχτεί προειδοποιητικό μήνυμα πως τα δεδομένα δεν έχουν περαστεί στο σύστημα και εκκρεμεί συγχρονισμός με το server. Συνεπως, ο χρήστης θα αντιληφθεί πως στη σελίδα που μεταβαίνει θα πρέπει με κάποιο τρόπο να επιχειρήσει συγχρονισμό. Σε περίπτωση που δεν εισήχθησαν κάποια δεδομένα στην τοπική βάση, η μετάβαση στο LoginPageγίνεται χωρίς να εμφανιστεί κάποιο προειδοποιητικό toast.

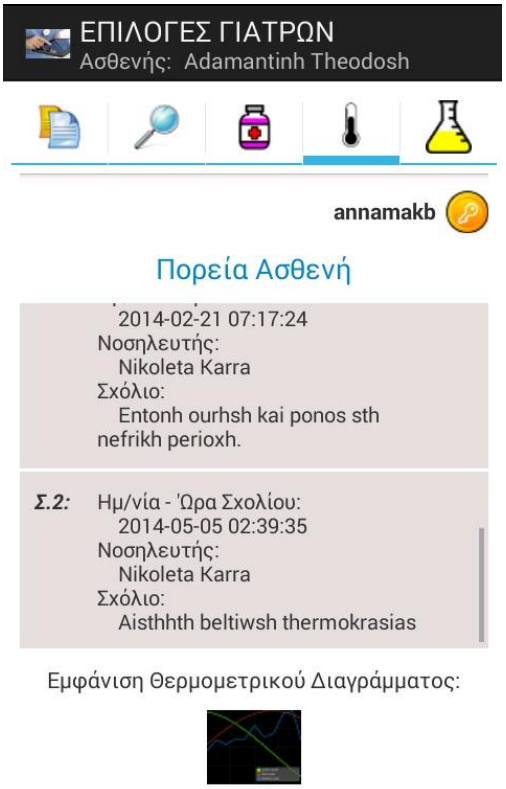
Ο συγχρονισμός γίνεται με το κουμπί τύπου refreshπου βρίσκεται στη συγκεκριμένη οθόνη.Αν πατηθεί, ό,τι δεδομένο υπήρχε στην τοπική μεταφέρεται στους αντίστοιχους πίνακες της κεντρικής βάσης.

Στη συγκεκριμένη περίπτωση, ο νοσηλευτής μας κατά την έξοδό του κάνει συγχρονισμό και κατόπιν για να επαληθεύσουμε τη σωστή εισαγωγή των νέων δεδομένων, μπαίνουμε στο σύστημα ως γιατρός. Ελέγχουμε αρχικά τους ασθενείς

της Παιδιατρικής κλινικής, στην οποία όντως προστέθηκε ο νέος ασθενής. Έπειτα από την Χειρουργική κλινική επιλέγουμε τον ασθενή AdamantiniTheodosikαι μεταβαίνουμε στο ProgressTab, με σκοπό να ανοίξουμε τα πεδία Ζωτικά Σημεία και Σχόλια. Ανοίγοντας τα παρατηρούμε πως όντως προστέθηκαν τα νέα δεδομένα που έβαλε ο νοσηλευτής στην κεντρική βάση.







Εικόνα 112: LogOut(α), Συγχρονισμός(β), Επαλήθευση(γ,δ,ε)

6.3. Αξιοσημείωτα αποσπάσματα κώδικα Java

Στην παρούσα ενότητα παρατίθενται κάποια αποσπάσματα από τον κώδικα Javaπου υλοποιήθηκε και αξίζουν αναφοράς.

Αρχικά λοιπόν παραθέτουμε την Activity**PatientsOfClinic**, στην οποία περιλαμβάνονται 5 μέθοδοι. Στη μέθοδο onCreate() καλείται το αντίστοιχο layoutκαι λαμβάνονται οι επιπλέον πληροφορίες από την προηγουμενη Activityπου θα φανούν χρήσιμες σε αυτή. Στη μέθοδο LogOut(), περιγράφεται τι θα πρέπει να συμβεί σε περίπτωση που ο χρήστης πατήσει το κουμπί της αποσύνδεσης. Στη μέθοδο createSpinnerDropDown(), γίνεται η δημιουργία του spinnerκαι ανάλογα με την επιλεγμένη κλινική φορτώνονται και οι κατάλληλοι ασθενείς. Η διαδικασία αυτή γίνεται μέσω της CustomOnSelectedListener()και της GetData(). Η τελευταία υλοποιεί τη σύνδεση με το serverκαι τη φόρτωση των ασθενών της κλινικής που επιλέχτηκε στο spinneraπό την κεντρική βάση. Στolayouttης συγκεκριμένης σελίδας, οι ασθενείς εμφανίζονται με μορφή λίστας, που κάθε γραμμή είναι ένα εικονίδιο και το ονοματεπώνυμο του εκάστοτε χρήστη. Για να υλοποιηθεί λίστα με αυτό τον τρόπο χρησιμοποιήσαμε μια επιπλέον βοηθητική κλάση **CustomList2**, η οποία περιγράφει επακριβώς πώς θα είναι διαμορφωμένη η λίστα. Κατά αυτόν τον τρόπο στο χρήστη φαίνεται μια οθόνη όπως φαίνεται στην εικόνα 99(PatientsPage), που φαίνεται παραπάνω. Με την ίδια λογική υλοποιήθηκαν και όλες οι σελίδες που περιλαμβάνουν απλού τύπου λίστα.(learn2crack)

PatientsOfClinic.java

```
package com.example.example;

import java.io.BufferedReader;

public class PatientsOfClinic extends Activity {

    private static LinkedList<Integer> PatientId = new LinkedList<Integer>();
    private static LinkedList<String> PatientFname = new LinkedList<String>();
    private static LinkedList<String> PatientLname = new LinkedList<String>();
    String username;
```

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


    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_patients_of_clinic);

    // Get the message from the intent
    Intent intent = getIntent();

    int userId = intent.getExtras().getInt("UserId");
    String[] Clinics = intent.getExtras().getStringArray("Clinics");
    String ChosenClinic = intent.getExtras().getString("ChosenClinic");
    String status = intent.getExtras().getString("Status");
    username = intent.getExtras().getString("username");

    TextView txt = (TextView) findViewById(R.id.User);
    txt.setText(username);
   // Toast.makeText(PatientsOfClinic.this,ChosenClinic+"\n"+status+"\n"+userId , Toast.LENGTH_LONG).show();
    createSpinnerDropDown(Clinics, ChosenClinic, status, userId);
}





/** Called when the user clicks the "LogOut" button */
public void LogOut(View view) {
    Intent intent = new Intent(this, GiveLoginInfoActivity.class);
    startActivity(intent);
}

//Add clinics into spinner dynamically
private void createSpinnerDropDown(String[] a, String b, String c, int d) {

    //we know from UserInfo that the size of Clinics Array is 50
    //we need to know how many are not nulls
    int h=0;
    for(int k=0;k<50;k++){
        if((a[k]!=null)&&(a[k]!=b))
            h=h+1;
    }

    Spinner spinner = (Spinner) findViewById(R.id.spinner2);
    final List<String> list = new ArrayList<String>();

    //first insert the chosen clinic in order to be seemed in spinner
    list.add(b);
    for(int i=0; i<h; i++){
        if (!a[i].equalsIgnoreCase(b))
        list.add(a[i]);
    }

    //create an ArrayAdaptar from the String List
    ArrayAdapter<String> dataAdapter = new ArrayAdapter<String>(this,android.R.layout.simple_spinner_item, list);
    //set the view for the Drop down list
    dataAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    //set the ArrayAdapter to the spinner
    spinner.setAdapter(dataAdapter);
    //attach the listener to the spinner
    spinner.setOnItemSelectedListener(new CustomOnItemSelectedListener(c,d));
}
```

```
public class CustomOnItemSelectedListener implements OnItemSelectedListener {

    //variables
    private String  status;
    private int userId;

    //constructor
    public CustomOnItemSelectedListener(String a, int b){
        status = a;
        userId = b;
    }

    //when an item is selected from spinner
    //we make a toast to know which one is selected and
    //we create the list with patients of the selected item
    //the first time, clinic from previous activity is selected
    public void onItemSelected(AdapterView<?> parent, View view, int pos,
            long id) {

        Toast.makeText(parent.getContext(),
                "Επιλέξατε "+ parent.getItemAtPosition(pos).toString()+" Κλινική",
                Toast.LENGTH_LONG).show();
       //Display message
        TextView text = (TextView)findViewById(R.id.txt);
        text.setText(R.string.PatientMessage);
       //Display PatientList
        ListView list = (ListView)findViewById(R.id.list);
       //All info needed to continue
        new GetData(list,parent.getItemAtPosition(pos).toString(),status,userId).execute("");
    }

    @Override
    public void onNothingSelected(AdapterView<?> arg0) {
        // TODO Auto-generated method stub

    }

}
```

```
private class GetData extends AsyncTask<String, Void, String> {

    //variables
    private ListView display ;
    private String chosenClinic, status;
    private int userId;


    //constructor
    GetData (ListView c,String a,String b,int d){
        display = c;
        chosenClinic = a;
        status = b;
        userId = d;
    }

@SuppressLint("UseValueOf")
    protected String doInBackground(String... message) {

        HttpPost request = new HttpPost();
        HttpResponse response = null;
        String result = "";

        // Try to connect using Apache HttpClient Library
        try {
            // Add your data
            List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(1);
            HttpClient httpclient = new DefaultHttpClient();

            request = new HttpPost("http://10.0.2.2:8888/PatientsOfSpecificClinic.php");

            //send chosenClinic

            nameValuePairs.add(new BasicNameValuePair("ChosenClinic", chosenClinic));
            request.setEntity(new UrlEncodedFormEntity(nameValuePairs));

            response = httpclient.execute(request);
            }




    catch (Exception e) {result = "aaaaaaaa";}

    // response code
    try {
        BufferedReader rd = new BufferedReader(new InputStreamReader(
                response.getEntity().getContent()));
        String line = "";
        while ((line = rd.readLine()) != null) {
            result = result + line ;
        }
    } catch (Exception e) {result = "bbbbbbbbbbbbbbbb";}
    return result;
}
```

```
protected void onPostExecute(String result) {

    try{
        JSONObject json = new JSONObject(result);
        int success = json.getInt("success");
        PatientId = new LinkedList<Integer>();
        PatientFname = new LinkedList<String>();
        PatientLname = new LinkedList<String>();

        // if there are clinics
        if(success == 1){
            JSONArray patientsOfClinic = json.getJSONArray("patientsOfClinic");
            for(int i=0;i<patientsOfClinic.length();i++){
                JSONObject json_data = patientsOfClinic.getJSONObject(i);
                PatientId.add(json_data.getInt("PatientId"));
                PatientFname.add(json_data.getString("Fname"));
                PatientLname.add(json_data.getString("Lname"));
            }
        }
```

```
        //if there are not patients in the specific clinic
        else{
            String message = json.getString("message");
            //show toast for not correct data
            Context context = getApplicationContext();
            int duration = Toast.LENGTH_LONG;
            Toast toast = Toast.makeText(context, message, duration);
            toast.show();
            //initialize arrays in order to be sent at List View as empty

        }
    }catch(JSONException e){
        String message = "error!!!!!!";
        //show toast for Error
        Context context = getApplicationContext();
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, message, duration);
        toast.show();
    }

    final LinkedList<String> PFname = new LinkedList<String>();
    final LinkedList<String> PLname = new LinkedList<String>();


 //put in list for listview
    for(int k=0;k<PatientFname.size();k++){
        if (PatientFname.get(k)!=null){
            PFname.add(PatientFname.get(k));
            PLname.add(PatientLname.get(k));
        }

    }
```

```
CustomList2 adapter = new CustomList2 (PatientsOfClinic.this, PFname, PLname );
this.display.setAdapter(adapter);

// click on a specific clinic and go to next page
this.display.setOnItemClickListener(new AdapterView.OnItemClickListener() {

    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        //Toast.makeText(PatientsOfClinic.this, "Επιλέξατε " +PatientFname[+position]+" "+PatientLname[+position
        Toast.makeText(PatientsOfClinic.this, "Επιλέξατε " +PFname.get(position)+" "+PLname.get(position), Toast.

        Intent intent = null;

        //if doctor, load the tabs for doctor's choices
        //else load the tabs for nurse's choices
        if (status.equalsIgnoreCase("Γιατρός")){
            intent = new Intent(PatientsOfClinic.this, DoctorTabChoices.class);
            intent.putExtra("DoctorId", userId);
        //  intent.putExtra("username", username);
        //  intent.putExtra("PatientName", PFname.get(position)+" "+PLname.get(position));
            //Toast.makeText(PatientsOfClinic.this,Integer.toBinaryString(userId) ,Toast.LENGTH_LONG).show();

            }
        else{
            intent = new Intent(PatientsOfClinic.this, NurseTabChoices.class);
        //  intent.putExtra("username", username);
            intent.putExtra("NurseId", userId);
        //  intent.putExtra("PatientName", PFname.get(position)+" "+PLname.get(position));

            //intent = new Intent(PatientsOfClinic.this, NurseChoices.class);
        }

        //store the items for the next activity
        intent.putExtra("username", username);
        intent.putExtra("PatientName", PFname.get(position)+" "+PLname.get(position));
        intent.putExtra("patientId", PatientId.get(position));
        intent.putExtra("ChosenClinic", chosenClinic);

        //start new activity
        startActivity(intent);
    }

            });

        }
    }
}
```

Εικόνα 113: PatientsOfClinic Activity

*PatientsOfClinic.java | CustomList2.java

```
package com.example.example;

import java.util.LinkedList;

public class CustomList2 extends ArrayAdapter<String>{

    private final Activity context;
    private LinkedList<String> web = new LinkedList<String>();
    private LinkedList<String> web2 = new LinkedList<String>();


    public CustomList2(Activity context,LinkedList<String> web,LinkedList<String> web2) {
        super(context, R.layout.list_single, web);
        this.context = context;
        this.web = web;
        this.web2 = web2;
    }


    //image + 2 arrays
    @Override
    public View getView(int position, View view, ViewGroup parent) {

        LayoutInflater inflater = context.getLayoutInflater();
        View rowView= inflater.inflate(R.layout.list_single, null, true);

        ImageView imageView = (ImageView) rowView.findViewById(R.id.img);
        TextView txtTitle = (TextView) rowView.findViewById(R.id.header);
        txtTitle.setTextSize(13);
        if (web.get(position)!=null){
            imageView.setImageResource(R.drawable.person);
            txtTitle.setText("   "+web.get(position)+" "+web2.get(position));
        }
        return rowView;

    }

}
```

Εικόνα 114: CustomList2

*PatientsOfClinic.java CustomList2.java activity_patients_of_clinic.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:layout_gravity="center_horizontal"
    android:orientation="vertical"
    tools:context=".PatientsOfClinic" >

   <LinearLayout
            android:layout_marginTop="10dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="220dp"
            android:orientation="horizontal" >

                    <TextView
                        android:id="@+id/User"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:layout_gravity="center"
                        android:textSize="14sp"
                        android:textStyle="bold"
                        />

                    <ImageButton
                        android:layout_marginLeft="5dp"
                        android:id="@+id/logout"
                        android:layout_width="28dp"
                        android:layout_height="28dp"
                        android:background="#FFFFFFFF"
                        android:layout_gravity="center"
                        android:src="@drawable/logout"
                        android:onClick="LogOut"  />
    </LinearLayout>


    <Spinner
        android:layout_marginTop="10dp"
        android:id="@+id/spinner2"
        android:layout_gravity="center"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
         />
    <requestFocus />


     <TextView
        android:layout_marginTop="10dp"
        android:id="@+id/txt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center" />


    <ListView
         android:layout_marginTop="10dp"
         android:layout_marginRight="5dp"
        android:layout_marginLeft="5dp"
        android:layout_marginBottom="10dp"
        android:id="@+id/list"
         android:layout_gravity="center"
         android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>

</LinearLayout>
```

Εικόνα115: activity_patients_of_clinic.xml

Ένας άλλος τύπος λίστας που υλοποιήθηκε είναι η expandablelist, η οποία ουσιαστικά είναι «υπολίστα» της λίστας. Ενδεικτικά αναφέρουμε την **HistoryTab**Activity , στην οποία περιλαμβάνεται η κλάση ExpandableListAdapter, την οποία και παραθέτουμε λεπτομερώς. Όσο αφορά τις υπόλοιπες μεθόδους που υπάρχουν είναι παρόμοιες σε μορφή με αντίστοιχες στην ActivityPatientsOfClinicπου περιγράφηκε παραπάνω, γι'αυτό και τις παρουσιάζουμε ελαχιστοποιημένες. Με παρόμοιο τρόπο δημιουργούνται οι expandablelistsκαι σε άλλα σημεία(Activities) της εφαρμογής.

```
activity_history_tab.xml   HistoryTab.java
package com.example.example;

import java.io.BufferedReader;

public class HistoryTab extends ExpandableListActivity{

    String username;
    private static final String HistoryList[] = {"Προφίλ Ασθενούς", "Χειρουργικές Επεμβάσεις", "Αλλεργίες","Παροντικά Συμπτώμ

    private static String HospitalizationDetails;

    private static LinkedList<String> Surgeries = new LinkedList<String>();
    private static LinkedList<String> HeadersSurgeries = new LinkedList<String>();

    private static LinkedList<String> Allergies = new LinkedList<String>();
    private static LinkedList<String> HeadersAllergies = new LinkedList<String>();

    private static LinkedList<String> Symptoms = new LinkedList<String>();
    private static LinkedList<String> HeadersSymptoms = new LinkedList<String>();

    private static LinkedList<String> ProfileElements = new LinkedList<String>();
    private static LinkedList<String> HeadersProfileElements =
            new LinkedList<String> (Arrays.asList("Ονοματεπώνυμο:\t", "Όνομα Πατρός:\t", "Όνομα Μητρός:\t","Ημερ/νία Γέννησης
                    "Διεύθυνση Κατοικίας:\t",   "Πόλη:\t","Χώρα:\t","Ταχυδρομικός Κώδικας:\t","Επάγγελμα:\t", "Φύλο:\t", "Τηλ
                    "Α.Φ.Μ:\t", "ΑΜΚΑ:\t", "Οικ/κή Κατάσταση:\t", "Ημ/νία -'Ωρα\nΕισαγωγής:\t", "Ημ/νία -'Ωρα\nΕξαγωγής:\t",
                    "Email:\t", "Ταμείο:\t", "Κάπνισμα:\t", "Αλκοόλ:\t", "Διατροφή:\t","Λεπτομέρειες\nΝοσηλείας:\t\t\t"));
```

```
    public void onCreate(Bundle savedInstanceState) {

    /** Called when the user clicks the "LogOut" button */
    public void LogOut(View view) {

    private class GetData extends AsyncTask<String, Void, String> {

    private class GetData1 extends AsyncTask<String, Void, String> {

    private class GetData2 extends AsyncTask<String, Void, String> {

    private class GetData3 extends AsyncTask<String, Void, String> {

    private class GetData4 extends AsyncTask<String, Void, String> {
```

```
public class ExpandableListAdapter extends BaseExpandableListAdapter {

        private Context myContext;

        public ExpandableListAdapter(Context context) {
            myContext = context;
        }

        @Override
        public Object getChild(int groupPosition, int childPosition) {
            return null;
        }

        @Override
        public long getChildId(int groupPosition, int childPosition) {
            return 0;
        }
```

```
        @SuppressLint("CutPasteId")
        @Override
        public View getChildView(int groupPosition, int childPosition, boolean isLastChild, View convertView, ViewGroup

            switch (groupPosition){

                case 0:
                    CustomList3 adapter0 = new CustomList3(HistoryTab.this,HeadersProfileElements, ProfileElements);
                    convertView = adapter0.getView(childPosition);
                    break;

                case 1:
                    CustomList3 adapter1 = new CustomList3(HistoryTab.this,HeadersSurgeries, Surgeries);
                    convertView = adapter1.getView(childPosition);
                    break;

                case 2:
                    CustomList3 adapter2 = new CustomList3(HistoryTab.this,HeadersAllergies, Allergies);
                    convertView = adapter2.getView(childPosition);
                    break;


                case 3:
                    CustomList3 adapter3 = new CustomList3(HistoryTab.this,HeadersSymptoms, Symptoms);
                    convertView = adapter3.getView(childPosition);
                    break;


            }

             return convertView;

    }
```

```
        @Override
        public int getChildrenCount(int groupPosition) {
            int a=0;
            if(groupPosition==0)
                a= HeadersProfileElements.size();
            else if(groupPosition==1)
                a=Surgeries.size();
            else if(groupPosition==2)
                a=Allergies.size();
            else
                a=Symptoms.size();
            return a;

        }

        @Override
        public Object getGroup(int groupPosition) {
            return null;
        }

        @Override
        public int getGroupCount() {

            return HistoryList.length;
        }

        @Override
        public long getGroupId(int groupPosition) {
            return 0;
        }



        @Override
        public View getGroupView(int groupPosition, boolean isExpanded,View convertView, ViewGroup parent) {
            if (convertView == null) {
                LayoutInflater inflater = (LayoutInflater) myContext.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                convertView = inflater.inflate(R.layout.group_layout_expandable_list, null);
            }
            TextView groupName = (TextView) convertView.findViewById(R.id.articleHeaderTextView);
            groupName.setText(HistoryList[groupPosition]);

            return convertView;
        }

        @Override
        public boolean hasStableIds() {
            return false;
        }

        @Override
        public boolean isChildSelectable(int groupPosition, int childPosition) {
            return true;
        }


    }
}
```

Εικόνα 116: History Tab Activity

activity_history_tab.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
 android:layout_width="wrap_content"
android:layout_height="match_parent"
android:layout_gravity="center_horizontal"
    android:orientation="vertical">


    <LinearLayout
            android:layout_marginTop="10dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="200dp"
            android:orientation="horizontal" >

                    <TextView
                        android:id="@+id/User"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:layout_gravity="center"
                        android:textSize="14sp"
                        android:textStyle="bold"
                        />

                    <ImageButton
                        android:layout_marginLeft="5dp"
                        android:id="@+id/logout"
                        android:layout_width="28dp"
                        android:layout_height="28dp"
                        android:background="#FFFFFFFF"
                        android:layout_gravity="center"
                        android:src="@drawable/logout"
                        android:onClick="LogOut"  />
    </LinearLayout>


  <TextView
                android:layout_marginTop="10dp"
                android:id="@+id/articleListHeaderTextView"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:text="@string/History"
                android:textColor="#0585C4"
                android:textAppearance="?android:attr/textAppearanceMedium"


                />

<ExpandableListView
                android:id="@android:id/list"
                android:layout_width="300dp"
                android:layout_height="match_parent"
                android:layout_gravity="center"
                android:layout_below="@id/articleListHeaderTextView"
                android:layout_margin="10dp"
                android:divider="@android:color/darker_gray"
                android:childDivider="@android:color/white"
                android:dividerHeight="2dp"
                android:fadingEdge="none">
</ExpandableListView>
</LinearLayout>
```

Εικόνα 117: activity_history_tab.xml

Για την υλοποίηση του θερμομετρικού διαγράμματος χρησιμοποιήθηκε η ActivityThermometricalDiagram. Ανάλογα με το πόσες μετρήσεις έχει ο ασθενής, καλέστηκαν οι ActivitiesThermometricalDiagram1, ThermometricalDiagram2, ThermometricalDiagram3, ThermometricalDiagram4, ThermometricalDiagram5. Ενδεικτικά παραθέτουμε την **ThermometricalDiagram5** και με παρόμοιο τρόπο υλοποιούνται και οι υπόλοιπες.(jandroiddevelopment)

```
*ThermometricalDiagram5.java
package com.example.example;

import java.util.ArrayList;

@SuppressLint("NewApi")
public class ThermometricalDiagram5 extends Activity
{

    private GraphicalView mChart;
    private static ArrayList<String> Hours = new ArrayList<String>();
    private static ArrayList<String> HoursChanged = new ArrayList<String>();

    private static ArrayList<String> Temperatures1 = new ArrayList<String>();
    private static ArrayList<String> Temperatures2 = new ArrayList<String>();
    private static ArrayList<String> Temperatures3 = new ArrayList<String>();
    private static ArrayList<String> Temperatures4 = new ArrayList<String>();
    private static ArrayList<String> Temperatures5 = new ArrayList<String>();
    private static String D1,D2,D3,D4,D5,PatientName;
```

```
@SuppressLint("NewApi")
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //Toast.makeText(ThermometricalDiagram.this, "malaka", Toast.LENGTH_LONG).show();

    setContentView(R.layout.activity_thermometrical_diagram);

    Intent intent = getIntent();
    D1 = intent.getExtras().getString("Date1");
    Hours = intent.getStringArrayListExtra("Hours");
    Temperatures1 = intent.getStringArrayListExtra("Temperatures1");
    D2 = intent.getExtras().getString("Date2");
    Temperatures2 = intent.getStringArrayListExtra("Temperatures2");
    D3 = intent.getExtras().getString("Date3");
    Temperatures3 = intent.getStringArrayListExtra("Temperatures3");
    D4 = intent.getExtras().getString("Date4");
    Temperatures4 = intent.getStringArrayListExtra("Temperatures4");
    D5 = intent.getExtras().getString("Date5");
    Temperatures5 = intent.getStringArrayListExtra("Temperatures5");
    PatientName = intent.getExtras().getString("PatientName");

    getActionBar().setSubtitle("Ασθενής:\t"+PatientName);
    OpenChart();
}
```

```
private void OpenChart()
{
 // Define the number of elements you want in the chart.
 int z[]={0,1,2,3};

 for(int i=0;i<Hours.size();i++){
     HoursChanged.add("\n\n"+Hours.get(i));
     //Toast.makeText(ThermometricalDiagram.this,  HoursChanged.get(i) , Toast.LENGTH_LONG).show();
 }

 // Create XY Series for X Series.
 XYSeries Date1 = new XYSeries(D1);
 XYSeries Date2 = new XYSeries(D2);
 XYSeries Date3 = new XYSeries(D3);
 XYSeries Date4 = new XYSeries(D4);
 XYSeries Date5 = new XYSeries(D5);

 //  Adding data to the X Series.
 for(int i=0;i<4;i++){
     Date1.add(z[i],Double.parseDouble(Temperatures1.get(i)));
     Date2.add(z[i],Double.parseDouble(Temperatures2.get(i)));
     Date3.add(z[i],Double.parseDouble(Temperatures3.get(i)));
     Date4.add(z[i],Double.parseDouble(Temperatures4.get(i)));
     Date5.add(z[i],Double.parseDouble(Temperatures5.get(i)));
     }

 // Create a Dataset to hold the XSeries.
 XYMultipleSeriesDataset dataset=new XYMultipleSeriesDataset();

 // Add X series to the Dataset.
 dataset.addSeries(Date1);
 dataset.addSeries(Date2);
 dataset.addSeries(Date3);
 dataset.addSeries(Date4);
 dataset.addSeries(Date5);
```

```
// Create XYSeriesRenderer to customize XSeries
XYSeriesRenderer Xrenderer1=new XYSeriesRenderer();
Xrenderer1.setColor(Color.CYAN);
Xrenderer1.setPointStyle(PointStyle.CIRCLE);
Xrenderer1.setDisplayChartValues(true);
Xrenderer1.setLineWidth(1);
Xrenderer1.setFillPoints(true);

XYSeriesRenderer Xrenderer2=new XYSeriesRenderer();
Xrenderer2.setColor(Color.WHITE);
Xrenderer2.setPointStyle(PointStyle.CIRCLE);
Xrenderer2.setDisplayChartValues(true);
Xrenderer2.setLineWidth(1);
Xrenderer2.setFillPoints(true);

XYSeriesRenderer Xrenderer3=new XYSeriesRenderer();
Xrenderer3.setColor(Color.GREEN);
Xrenderer3.setPointStyle(PointStyle.CIRCLE);
Xrenderer3.setDisplayChartValues(true);
Xrenderer3.setLineWidth(1);
Xrenderer3.setFillPoints(true);

XYSeriesRenderer Xrenderer4=new XYSeriesRenderer();
Xrenderer4.setColor(Color.YELLOW);
Xrenderer4.setPointStyle(PointStyle.CIRCLE);
Xrenderer4.setDisplayChartValues(true);
Xrenderer4.setLineWidth(1);
Xrenderer4.setFillPoints(true);

XYSeriesRenderer Xrenderer5=new XYSeriesRenderer();
Xrenderer5.setColor(Color.RED);
Xrenderer5.setPointStyle(PointStyle.CIRCLE);
Xrenderer5.setDisplayChartValues(true);
Xrenderer5.setLineWidth(1);
Xrenderer5.setFillPoints(true);
```

```
// Create XYMultipleSeriesRenderer to customize the whole chart
XYMultipleSeriesRenderer mRenderer=new XYMultipleSeriesRenderer();
mRenderer.setChartTitle("Θερμομετρικό Διάγραμμα\n\tΑνά 6 ώρες το 24ωρο\n\tγια τα 5 προηγούμενα 24ωρα.");
mRenderer.setXTitle("\n\nΩρα Μέτρησης");
mRenderer.setYTitle("Θερμοκρασία");
mRenderer.setLegendTextSize(13);
mRenderer.setZoomButtonsVisible(true);
mRenderer.setShowGrid(true);
for(int i=0;i<z.length;i++){
    mRenderer.addXTextLabel(i, HoursChanged.get(i));
}
// Adding the XSeriesRenderer to the MultipleRenderer.
mRenderer.addSeriesRenderer(Xrenderer1);
mRenderer.addSeriesRenderer(Xrenderer2);
mRenderer.addSeriesRenderer(Xrenderer3);
mRenderer.addSeriesRenderer(Xrenderer4);
mRenderer.addSeriesRenderer(Xrenderer5);
LinearLayout chart_container=(LinearLayout)findViewById(R.id.Chart_layout);
```

```
    // Creating an intent to plot line chart using dataset and multipleRenderer
    mChart=(GraphicalView)ChartFactory.getLineChartView(getBaseContext(), dataset, mRenderer);

    //  Adding click event to the Line Chart.
    mChart.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View arg0) {
            // TODO Auto-generated method stub
            SeriesSelection series_selection=mChart.getCurrentSeriesAndPoint();
            if(series_selection!=null){
                int series_index=series_selection.getSeriesIndex();
                String select_series = null;
                if(series_index==0){
                    select_series="X Series";
                }
                else{
                    select_series="Y Series";
                }
            }
        }
    });


    // Add the graphical view mChart object into the Linear layout .
    chart_container.addView(mChart);


   }


}
```

Εικόνα 118: ThermometricalDiagram5

7. ΕΠΙΛΟΓΟΣ

7.1. Συμπεράσματα

Η εφαρμογή που υλοποιήσαμε μπορεί να αποτελέσει ένα από τα ηλεκτρονικά εργαλεία Γιατρών/Νοσηλευτών στη σύγχρονη νοσοκομειακή πραγματικότητα. Μέσω αυτής οι χρήστες στους οποίους απευθύνεται έχουν τη δυνατότητα να ελέγξουν πλήρως την κλινική εικόνα κάθε ασθενή καθώς και να συμπληρώσουν στοιχεία που σχετίζονται με αυτή. Αυτές οι δυνατότητες τους προσφέρονται σε ένα αρκετά σύγχρονο και εύχρηστο προς αυτούς περιβάλλον.

'Ενα από τα βασικά πλεονεκτήματα της συγεκριμένης εφαρμογής είναι το λογισμικό Android, το οποίο είναι από τα πλέον διαδεδομένα σήμερα. Πολλές από τις σύγχρονες φορητές συσκευές κάνουν χρήση του συγκεκριμένου λογισμικού. Παράλληλα, μέσω της εφαρμογής μειώνεται αισθητά ο μεγάλος όγκος γραφικής ύλης που απαιτείται για φακέλους, συνταγογραφήσεις εξετάσεις ασθενών κτλ, καθώς όλες οι πληροφορίες βρίσκονται συγκεντρωμένες στην κεντρική βάση δεδομένων. Επίσης, κατά την εισαγωγή στοιχείων νέου ασθενή ή τιμών ζωτικών σημείων, διατίθενται έλεγχοι εγκυρότητας και ο χρήστης πληροφορείται άμεσα για τις δυνατές τιμές των πεδίων, ώστε να αποφεύγονται πιθανά ανθρώπινα λάθη. Όλες οι λειτουργίες γίνονται αυτοματοποιημένα και συνεπώς εξοικονομείται χρόνος, εφόσον η αναζήτηση φακέλων ασθενή και εξετάσεων η διαδικασία προσθήκης φαρμακευτικής αγωγής, η μελέτη συνολικής πορείας ασθενή και πολλές ακόμη λειτουργίες γίνονται με ένα κλικ στην κινητή συσκευή. Αυτοματοποιημένη είναι επίσης η διαδικασία αποστολής παραπεμπτικών στα εργαστήρια για εξετάσεις των ασθενών. Έτσι, όταν τα αποτελέσματα είναι έτοιμα το σύστημα ενημερώνεται αυτόματα χωρίς επιπλέον κωλυσιεργίες από το προσωπικό. Η άμεση λήψη των αποτελεσμάτων είναι σημαντική καθώς μπορεί να σώσει μέχρι και ανθρώπινες ζωές. Τέλος, η πληροφόρηση που παρέχει το σύστημα για σημαντικές ημερομηνίες, μπορεί να κάνει τη λειτουργία της νοσοκομειακής μονάδας πολύ πιο ελεγχόμενη.

Συνεπώς, συμπεραίνουμε πως η εφαρμογή αυτή θα μπορούσε να φανεί για πολλούς λόγους χρήσιμη στη σωστή και αποδοτική λειτουργία των σύγχρονων νοσοκομείων.

7.2. Μελλοντικές Βελτιώσεις

Στην παρούσα διπλωματική εργασία υλοποιήθηκαν πολλές από τις λειτουργικές απαιτήσεις μιας εφαρμογής που απευθύνεται στο νοσοκομειακό προσωπικό(Γιατρούς και Νοσηλευτές).

Στο μέλλον θα μπορούσαν να γίνουν βελτιώσεις σχετικές είτε με επιπλέον λειτουργικότητες είτε με επέκταση των ήδη υλοποιημένων. Κάποιες από αυτές παραθέτουμε στη συγκεκριμένη ενότητα.

Στη σελίδα που ο χρήστης εισάγει το στοιχεία του (όνομα χρήστη & κωδικός πρόσβασης) θα μπορούσε να υπάρχει έλεγχος από το σύστημα που να ενημερώνει το χρήστη ποιο πεδίο από τα δύο έκανε λάθος, όταν δώσει μη έγκυρα στοιχεία. Επίσης κατά τη διαδικασία εισαγωγής νέου ασθενή, θα μπορούσε να γίνει και εισαγωγή στοιχείων σχετικών με τις χειρουργικές επεμβάσεις, τις αλλεργίες και τα τρέχοντα συμπτώματα και όχι μόνο του προσωπικού προφίλ του ασθενή. Ο νοσηλευτής επίσης θα μπορούσε να έχει τη δυνατότητα πέραν της ανάγνωσης του ιστορικού κάποιου ασθενή, να κάνει και επεξεργασία ορισμένων πεδίων(λειτουργία edit). Επίσης, θα μπορούσε η εφαρμογή να υποστηρίζει και τη λειτουργία διαγραφής κάποιου χρήστη είτε από Γιατρό είτε από Νοσηλευτή. Τέλος, θα μπορούσε να υπάρχει σε κάποιο σημείο καταγεγραμμένη η ακριβής ημερομηνία και ώρα επίσκεψης του γιατρού ή νοσηλευτή στο θάλαμο του ασθενή.

8. ΒΙΒΛΙΟΓΡΑΦΙΑ - ΑΝΑΦΟΡΕΣ

1. Meier, R. (2010). *Professional Android 2 Application Development.* Wrox.
2. Ζορμπά, Ι. (2011). *Ανάπτυξη εφαρμογών σε κινητά και φορητές συσκευές(PDA's) για εξυπηρέτηση ιατρών και νοσηλευτών εντός νοσοκομειακών ιδρυμάτων.*
3. Οικονομόπουλος, Κ. (2013). *Ανάπτυξη εφαρμογής για παρακολούθηση θεμάτων υγείας και θεραπείας σε φορητές συσκευές(PDAS) για Android.*
4. Τερζής, Π. (2013). *Εφαρμογή Κινητού(smartphone) και tablet για ηλεκτρονική καρτέλα ασθενούς σε νοσοκομειακές μονάδες με τεχνολογία WindowsPhone 8.*
5. *gkatzios*. (2013). Retrieved from http://gkatzios.blogspot.com/2013/11/smartphones-tablets.html
6. *AndroidGreece*. (n.d.). Retrieved from http://androidgreece.gr/forum/viewtopic.php?f=4&t=160
7. *androidhive*. (n.d.). Retrieved from http://www.androidhive.info/2012/05/how-to-connect-android-with-php-mysql/
8. *coderanch*. (n.d.). Retrieved from http://www.coderanch.com/t/570282/Android/Mobile/post-data-android-app-php
9. *developer.android*. (n.d.). Retrieved from http://developer.android.com/training/basics/firstapp/index.html
10. *dide.flo.sch*. (n.d.). Retrieved from http://dide.flo.sch.gr/Plinet/Tutorials-PHP-Bible/Tutorials-PHP-Bible-1.html
11. *hmkcode*. (n.d.). Retrieved from http://hmkcode.com/android-simple-sqlite-database-tutorial/
12. *homeomedi*. (n.d.). Retrieved from http://www.homeomedi.com/aimatologikesexetaseis.html
13. *how-to-solve-xampp-problem*. (n.d.). Retrieved from http://el.web.id/how-to-solve-xampp-problem-port-80-443-in-use-by-skype-470
14. *imop*. (n.d.). Retrieved from http://www.imop.gr/urotools-normal-values
15. *imop*. (n.d.). Retrievedfromhttp://www.imop.gr/diagnwstikes-eksetaseis/ouria#Ποια είναι τα αποτελέσματα της εξέτασης;

16. *jandroiddevelopment*. (n.d.). Retrieved from http://jandroiddevelopment.blogspot.gr/2013/08/android-charts-using-chart-engine-part.html

17. *json*. (n.d.). Retrieved from http://www.json.org/json-el.html

18. *learn2crack*. (n.d.). Retrieved from http://www.learn2crack.com/2013/10/android-custom-listview-images-text-example.html

19. *oracle*. (n.d.). Retrieved from http://docs.oracle.com/javaee/6/tutorial/doc/gijqy.html

20. *techrepublic*. (n.d.). Retrieved from http://www.techrepublic.com/blog/smb-technologist/import-and-export-databases-using-phpmyadmin/

21. *uml-diagrams*. (n.d.). Retrieved from http://www.uml-diagrams.org/sequence-diagrams.html

22. *uml-diagrams*. (n.d.). Retrieved from http://www.uml-diagrams.org/use-case-diagrams-examples.html

23. *visual-paradigm*. (n.d.). Retrieved from http://www.visual-paradigm.com/editions/community.jsp

24. *vogella*. (n.d.). Retrieved from http://www.vogella.com/tutorials/AndroidSQLite/article.html

25. *wikihow*. (n.d.). Retrieved from http://www.wikihow.com/Write-PHP-Scripts

26. *wikipedia*. (n.d.). Retrieved from http://el.wikipedia.org/wiki/%CE%94%CE%B9%CE%B5%CE%B8%CE%BD%CE%AE%CF%82_%CE%A3%CF%84%CE%B1%CF%84%CE%B9%CF%83%CF%84%CE%B9%CE%BA%CE%AE_%CE%A4%CE%B1%CE%BE%CE%B9%CE%BD%CF%8C%CE%BC%CE%B7%CF%83%CE%B7_%CE%9D%CE%BF%CF%83%CE%B7%CE%BC%CE%AC%CF%84%CF%89%CE%BD_%CE%BA

27. *wikipedia*. (n.d.). Retrieved from http://el.wikipedia.org/wiki/XAMPP

28. *wikipedia*. (n.d.). Retrieved from http://el.wikipedia.org/wiki/%CE%91%CE%BC%CF%85%CE%BB%CE%AC%CF%83%CE%B7

29. *wikipedia*. (n.d.). Retrieved from http://el.wikipedia.org/wiki/Android

30. *youtube*. (n.d.). Retrieved from http://www.youtube.com/watch?v=UvMZqwyErrs