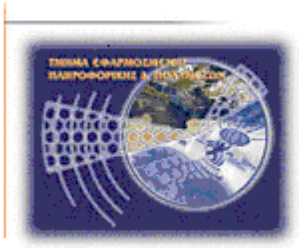
Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων



ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΘΕΜΑ

«Πληροφοριακό σύστημα για νοσοκομείο»

Γεωργουλάκης Μιχάλης Α.Μ. 2515

Πιριντζιής Πέτρος Α.Μ. 2491

Επιβλέπων καθηγητής: Παπαδάκης Νικόλαος

Ευχαριστίες

Με την ολοκλήρωση της πτυχιακής μας εργασίας θα θέλαμε να ευχαριστήσουμε όλους όσους μας βοήθησαν για την πολύτιμη στήριξή τους. Κυρίως οφείλουμε να ευχαριστήσουμε τον επιβλέποντα καθηγητή μας κ. Παπαδάκη Νικόλαο ο οποίος μας στήριξε και μας καθοδήγησε καθ' όλη τη διάρκεια της πτυχιακής εργασίας. Τέλος, οφείλουμε να αφιερώσουμε την πτυχιακή μας εργασία στις οικογένειες μας και σε όλους εκείνους που ήταν δίπλα μας σε όλη αυτή την προσπάθεια μας αλλά και όλα τα χρόνια της φοίτησης μας παρέχοντας μας στήριξη και απεριόριστη κατανόηση.

Abstract

Databases are one of the key tools in managing information and making transactions. Although for several years the databases have penetrated into our daily lives and are an integral part of any complex process, there are still large organizations that do not have a modern and efficient database and also, the corresponding management system. In this work, the management system of a hospital and its database are designed and implemented.

For the development of the database and the system management, tools and languages of the latest technology were used like Adobe Dreamweaver CS6, HTML5, CSS3, JavaScript and AJAX.

The database design begins with a detailed analysis of the system requirements, followed by the creation of an Entiry-Relationship model which is then converted to a Relational Model. The integrity restrictions, functional dependencies are also presented. Finally, the model is converted into canonical form before realized using SQL.

For the database management, a website was developed in HTML5 and PHP. The system provides functions for adding, changing, and finding the various entities of the database and is designed for easy and efficient of a large database.

Σύνοψη

Ένα από τα βασικότερα εργαλεία στην διαχείριση πληροφορίας και στην πραγματοποίηση συναλλαγών αποτελούν οι βάσεις δεδομένων. Παρόλο που εδώ και αρκετά χρόνια οι βάσεις δεδομένων έχουν εισχωρήσει στην καθημερινότητα μας και είναι αναπόσπαστο κομμάτι κάθε πολύπλοκης διαδικασίας, είναι ακόμη συχνό φαινόμενο η ύπαρξη μεγάλων οργανισμών που δεν έχουν μία σύγχρονη και αποδοτική βάση δεδομένων, με το αντίστοιχο σύστημα διαχείρισης. Στην παρούσα εργασία, σχεδιάζεται και υλοποιείται η βάση δεδομένων για ένα νοσοκομείο, καθώς και το σύστημα διαχείρισης της.

Για την ανάπτυξη της βάσης δεδομένων και του συστήματος διαχείρισης χρησιμοποιήθηκαν εργαλεία και γλώσσες τελευταίας τεχνολογίας όπως το Adobe Dreamweaver CS6, HTML5, CSS3 και Javascript.

Πριν τον σχεδιασμό της βάσης γίνεται μία λεπτομερής ανάλυση των αναγκών του συστήματος. Δημιουργείται το μοντέλο οντοτήτων σχέσεων και με εφαρμογή των αντίστοιχων κανόνων αυτό μετατρέπεται στο σχεσιακό μοντέλο. Παρουσιάζονται οι περιορισμοί ακεραιότητας, οι συναρτησιακές εξαρτήσεις και τέλος το μοντέλο μετατρέπεται σε κανονική μορφή, πριν υλοποιηθεί με χρήση SQL.

Για την διαχείριση της βάσης δεδομένων δημιουργήθηκε ένας ιστοχώρος με χρήση HTML5 και PHP. Το σύστημα προσφέρει λειτουργίες για προσθήκη, αλλαγή και εύρεση των διάφορων οντοτήτων της βάσης, ενώ σχεδιάστηκε με στόχο την αποτελεσματική και εύκολη διαχείριση μίας μεγάλης βάσης δεδομένων.

Πίνακας Περιεχομένων

Ευχαριστίες 2

Abstract 3

Σύνοψη 4

Πίνακας Περιεχομένων 5

Πίνακας Εικόνων 11

Κεφάλαιο 1: Εισαγωγή 14

1.1 Κίνητρα και στόχοι της εργασίας 14

1.2 Δομή της εργασίας 15

Κεφάλαιο 2: Βιβλιογραφική επισκόπηση 17

2.1 Η δημιουργία των βάσεων δεδομένων – Ιστορική αναδρομή 17

2.2 Μοντέλα συστημάτων βάσεων δεδομένων 17

2.2.1 Μοντέλο πλοήγησης 17

2.2.2 Σχεσιακό μοντέλο 18

2.2.3 Το μοντέλο οντοτήτων σχέσεων 19

2.3 Στατικές vs Δυναμικές Ιστοσελίδες 19

2.3.1 Στατικές Ιστοσελίδες 19

2.3.2 Δυναμικές Ιστοσελίδες 20

2.4 Εργαλεία Ανάπτυξης 20

2.4.1 SmartDraw 20

2.4.2 XAMPP 20

2.4.3 MySQL workbench 21

2.4.4 Adobe Dreamweaver CS6 22

2.5 Τεχνολογίες 22

2.5.1 HTML5 22
2.5.2 PHP 23
2.5.3 CSS3 23
2.5.4 JavaScript 24
2.5.5 AJAX (Asynchronous JavaScript and XML) 24
Κεφάλαιο 3: Μοντέλο οντοτήτων συσχετίσεων - μετατροπή σε σχεσιακό μοντέλο 25
3.1 Μοντέλο Οντοτήτων - Συσχετίσεων 25
3.1.1 Θεωρία 25
3.1.2 Εφαρμογή του μοντέλου οντοτήτων-συσχετίσεων 29
3.2 Κανόνες Μετατροπής στο σχεσιακό μοντέλο. 34
3.2.1 Εφαρμογή 1ου κανόνα (Ισχυρές οντότητες) 35
3.2.2 Εφαρμογή 2ου κανόνα (Ασθενείς οντότητες) 36
3.2.3 Εφαρμογή 3ου κανόνα (Πλειότιμα γνωρίσματα) 37
3.2.4 Εφαρμογή 4ου κανόνα (Συσχετίσεις M-N) 37
3.2.5 Εφαρμογή 5ου κανόνα (Συσχετίσεις 1-1) 37
3.2.6 Εφαρμογή 6ου κανόνα (Συσχετίσεις 1-N) 38
3.2.7 Συνολικά 38
3.3 Περιορισμοί Ακεραιότητας 39
3.4 Συναρτησιακές Εξαρτήσεις 39
3.5 Κανονικές Μορφές 44
3.5.1 Πρώτη Κανονική Μορφή (1NF) 44
3.5.2 Δεύτερη Κανονική Μορφή (2NF) 44
3.5.3 Τρίτη Κανονική Μορφή (3NF) 45
3.6 Διάγραμμα Βάσης Δεδομένων 46
3.7 Κώδικας SQL για την δημιουργία της βάσης δεδομένων 47

Κεφάλαιο 4: Εγχειρίδιο χρήσης και υλοποίηση 54
4.1 Διεπαφή της εφαρμογής 54
4.2 Αρχική σελίδα 55
4.3 Ασθενείς 55
4.3.1 Δημιουργία Ασθενή (Patients->Create new) 56
4.3.2 Εύρεση Ασθενή (Patients->Find) 58
4.3.3 Προβολή όλων των Ασθενών (Patients->View All) 60
4.3.4 Επεξεργασία Ασθενή 61
4.3.5 Έναρξη Νοσηλείας Ασθενή σε Κλινική 61
4.3.6 Λήξη Νοσηλείας Ασθενή σε Κλινική (Check out) 62
4.3.7 Προβολή Ιστορικού Νοσηλείας Ασθενή 63
4.3.8 Προβολή Ιστορικού Ασθενείας ενός Ασθενή 63
4.3.9 Προβολή Ιστορικού Εξετάσεων ενός Ασθενή 64
4.3.10 Διαγραφή Ασθενή 65
4.4 Γιατροί 66
4.4.1 Δημιουργία Γιατρού (Doctors->Create new) 67
4.4.2 Εύρεση Γιατρού (Doctors->Find) 69
4.4.3 Προβολή όλων των Γιατρών 70
4.4.4 Επεξεργασία Γιατρού 71
4.4.5 Διαγραφή Γιατρού 72
4.5 Κλινικές 73
4.5.1 Δημιουργία Κλινικής (Clinics->Create new) 73
4.5.2 Εύρεση Κλινικής(Clinics->Find) 74
4.5.3 Προβολή όλων των Κλινικών (Clinics->View All) 76
4.5.4 Επεξεργασία Κλινικής & Ορισμός Manager 76
4.5.5 Διαγραφή Κλινικής 77

4.6 Εξετάσεις 78
4.6.1 Δημιουργία Εξέτασης 79
4.6.2 Εύρεση Εξέτασης 80
4.6.3 Προβολή όλων των εξετάσεων 81
4.6.4 Επεξεργασία Εξέτασης 82
4.6.5 Διαγραφή Εξέτασης 83
4.6.6 Προγραμματισμός εξέτασης σε ασθενή 84
4.6.7 Προβολή όλων των προγραμματισμένων εξετάσεων 85
4.6.8 Ακύρωση προγραμματισμένης εξέτασης 85
4.6.9 Επεξεργασία Προγραμματισμένης Εξέτασης 86
4.6.10 Προβολή ιστορικού εξετάσεων του νοσοκομείου 87
4.7 Ασθένειες 88
4.7.1 Δημιουργία Ασθένειας 88
4.7.2 Εύρεση Ασθένειας 89
4.7.3 Προβολή όλων των Ασθενειών 90
4.7.4 Επεξεργασία Ασθένειας 91
4.7.5 Διαγραφή Ασθένειας 91
4.8 Αντιδραστήρια 92
4.8.1 Δημιουργία Αντιδραστήριου (Reagents->Create new) 93
4.8.2 Εύρεση Αντιδραστήριου(Reagent->Find) 94
4.8.3 Προβολή όλων των Αντιδραστήριων (Reagents->View All) 95
4.8.4 Επεξεργασία Αντιδραστήριου 96
4.8.5 Διαγραφή Αντιδραστηριού 97
Κεφάλαιο 5: Αποτελέσματα 99
5.1 Συμπεράσματα 99
5.2 Μελλοντική Εργασία και Επεκτάσεις 100
Βιβλιογραφία 101

Παράρτημα..... 103

1. Βασικό template 103
2. Φόρμα δημιουργίας νέου ασθενή (front-end)..... 105
3. Υλοποίηση μηχανισμού για Real-Time Validation (AJAX/JavaScript)..... 106
4. PHP αρχείο realTimeVal.php 107
5. Δημιουργία νέου ασθενή (back-end)..... 107
6. Φόρμα επεξεργασίας ασθενή (front-end) 108
7. Φόρμα προβολής ιστορικού του ασθενή 110
8. Φόρμα προβολής ιστορικού εξετάσεων του ασθενή 112
9. Επεξεργασία ασθενή (back-end) 112
10. Φόρμα εύρεσης ασθενή (front-end)..... 115
11. Εύρεση/Προβολή ασθενών..... 116
12. Φόρμα δημιουργίας νέου γιατρού(front-end)..... 118
13. Δημιουργία νέου γιατρού(back-end) 120
14. Φόρμα επεξεργασίας γιατρού(front-end) 121
15. Επεξεργασίας γιατρού(back-end)..... 124
16. Φόρμα εύρεσης γιατρού (front-end)..... 127
17. Εύρεση/Προβολή γιατρών..... 128
18. Φόρμα δημιουργίας νέας κλινικής(front-end)..... 129
19. Δημιουργία νέας κλινικής(back-end) 129
20. Φόρμα επεξεργασίας κλινικής(front-end) 130
21. Επεξεργασίας κλινικής(back-end)..... 132
22. Φόρμα εύρεσης κλινικής (front-end)..... 133
23. Εύρεση/Προβολή κλινικών 133
24. Φόρμα δημιουργίας νέας ασθένειας(front-end)..... 134
25. Δημιουργία νέας ασθένεια(back-end) 135
26. Φόρμα επεξεργασίας ασθένειας(front-end)..... 135

27. Επεξεργασία ασθένειας(back-end) 136
28. Φόρμα εύρεσης ασθένειας (front-end) 137
29. Εύρεση/Προβολή ασθενειών 138
30. Φόρμα δημιουργίας νέου αντιδραστήριου(front-end) 138
31. Δημιουργίας νέου αντιδραστήριου(back-end) 139
32. Φόρμα επεξεργασίας αντιδραστήριου(front-end) 140
33. Επεξεργασία αντιδραστήριου (back-end) 141
34. Φόρμα εύρεσης αντιδραστήριου (front-end) 142
35. Εύρεση/Προβολή αντιδραστήριων 143

Πίνακας Εικόνων

Εικόνα 1: Μοντέλο E-R του συστήματος. 29
Εικόνα 2: Μοντέλο E-R του συστήματος (χωρίς γνωρίσματα). 30
Εικόνα 3: Η διεπαφή χρήστη του συστήματος. 54
Εικόνα 4: Η αρχική σελίδα του συστήματος. 55
Εικόνα 5: Patient menu. 56
Εικόνα 6: Φόρμα για την δημιουργία ασθενή. 56
Εικόνα 7: HTML5 date input type. 57
Εικόνα 8: Επικύρωση εισόδου με την χρήση pattern. 57
Εικόνα 9: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή ασθενή στο σύστημα. 58
Εικόνα 10: Φόρμα εύρεσης ασθενή. 58
Εικόνα 11: Προσαρμογή πεδίου αναζήτησης με JavaScript. 59
Εικόνα 12: Εμφάνιση αποτελεσμάτων κατά την εύρεση ασθενή. 59
Εικόνα 13: Σελίδα προβολής όλων των ασθενών. 60
Εικόνα 14: Φόρμα επεξεργασίας ασθενή. 61
Εικόνα 15: Πεδία για εισαγωγή ασθενή σε κλινική. 62
Εικόνα 16: Επιλογή checkout για ασθενείς που νοσηλεύονται. 62
Εικόνα 17: Ιστορικό νοσηλείας ασθενή. 63
Εικόνα 18: Ιστορικό ασθενείας ενός ασθενή 63
Εικόνα 18: Ιστορικό ασθενείας ενός ασθενή 64
Εικόνα 19: Μήνυμα επιβεβαίωσης πριν την διαγραφή ασθενή. 65
Εικόνα 20: Μήνυμα ενημέρωσης για επιτυχή διαγραφή ασθενή. 66
Εικόνα 21: Doctors menu. 66
Εικόνα 22: Φόρμα για την δημιουργία γιατρού. 67
Εικόνα 23: Χρήση JavaScript για δυναμική αλλαγή πεδίου. 68
Εικόνα 24: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή γιατρού στο σύστημα. 68
Εικόνα 25: Φόρμα εύρεσης γιατρού. 69
Εικόνα 26: Εμφάνιση αποτελεσμάτων κατά την εύρεση γιατρού. 70
Εικόνα 27: Σελίδα προβολής όλων των γιατρών. 70
Εικόνα 28: Φόρμα επεξεργασίας γιατρού. 71
Εικόνα 29: Μήνυμα επιβεβαίωσης πριν την διαγραφή γιατρού. 72
Εικόνα 30: Μήνυμα ενημέρωσης για επιτυχή διαγραφή γιατρού. 73
Εικόνα 31: Clinics menu. 73

Εικόνα 32: Φόρμα για την δημιουργία κλινικής. 74
Εικόνα 33: HTML5 number input type. 74
Εικόνα 34: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή κλινικής στο σύστημα. 74
Εικόνα 35: Φόρμα εύρεσης κλινικής. 74
Εικόνα 36: Εμφάνιση αποτελεσμάτων κατά την εύρεση κλινικής. 75
Εικόνα 37: Σελίδα προβολής όλων των κλινικών. 76
Εικόνα 38: Φόρμα επεξεργασίας κλινικής. 76
Εικόνα 39: Μήνυμα επιβεβαίωσης πριν την διαγραφή κλινικής. 77
Εικόνα 40: Μήνυμα ενημέρωσης για επιτυχή διαγραφή κλινικής. 78
Εικόνα 41: Exams menu. 78
Εικόνα 42: Φόρμα για την δημιουργία εξέτασης. 79
Εικόνα 43: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή εξέτασης στο σύστημα. 79
Εικόνα 44: Φόρμα εύρεσης ασθένειας. 80
Εικόνα 45: Εμφάνιση αποτελεσμάτων κατά την εύρεση εξέτασης. 81
Εικόνα 46: Σελίδα προβολής όλων των εξετάσεων. 81
Εικόνα 47: Φόρμα επεξεργασίας εξέτασης. 82
Εικόνα 48: Μήνυμα επιβεβαίωσης πριν την διαγραφή εξέτασης. 83
Εικόνα 49: Μήνυμα ενημέρωσης για επιτυχή διαγραφή εξέτασης. 83
Εικόνα 50: Προγραμματισμός εξέτασης σε ασθενή. 84
Εικόνα 51: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή εξέτασης ασθενή. 84
Εικόνα 52: Σελίδα προβολής όλων των προγραμματισμένων εξετάσεων 85
Εικόνα 53: Μήνυμα ενημέρωσης για επιτυχή ακύρωση εξέτασης. 85
Εικόνα 54:Φόρμα επεξεργασίας προγραμματισμένης εξέτασης. 86
Εικόνα 55: Μήνυμα ενημέρωσης για τον καθορισμό αποτελέσματος μιας εξέτασης. 87
Εικόνα 56: Σελίδα προβολής προηγούμενων εξετάσεων 87
Εικόνα 57: Diseases menu. 88
Εικόνα 58: Φόρμα για την δημιουργία ασθένειας. 88
Εικόνα 59: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή γιατρού στο σύστημα. 89
Εικόνα 60: Φόρμα εύρεσης ασθένειας. 89
Εικόνα 61: Εμφάνιση αποτελεσμάτων κατά την εύρεση ασθένειας. 90
Εικόνα 62: Σελίδα προβολής όλων των ασθενειών 90
Εικόνα 63: Φόρμα επεξεργασίας ασθένειας. 91
Εικόνα 64: Μήνυμα επιβεβαίωσης πριν την διαγραφή ασθένειας. 91

Εικόνα 65: Μήνυμα ενημέρωσης για επιτυχή διαγραφή ασθένειας. 92
Εικόνα 66: Reagents menu. 92
Εικόνα 67: Φόρμα για την δημιουργία αντιδραστήριου. 93
Εικόνα 68: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή αντιδραστήριου στο σύστημ93
Εικόνα 69: Φόρμα εύρεσης αντιδραστήριου. 94
Εικόνα 70: Εμφάνιση αποτελεσμάτων κατά την εύρεση αντιδρασήριου. 95
Εικόνα 71: Σελίδα προβολής όλων των αντιδραστήριων. 95
Εικόνα 72: Φόρμα επεξεργασίας αντιδραστήριου. 96
Εικόνα 73: Μήνυμα επιβεβαίωσης πριν την διαγραφή αντιδραστήριου 97
Εικόνα 74: Μήνυμα ενημέρωσης για επιτυχή διαγραφή αντιδραστήριου. 97

Κεφάλαιο 1: Εισαγωγή

Οι βάσεις δεδομένων αποτελούν το βασικότερο εργαλείο για την διαχείριση πληροφορίας και την πραγματοποίηση διάφορων συναλλαγών. Είναι ένα τόσο σημαντικό κομμάτι της σημερινής ζωής που συχνά αγνοούμε ότι βρίσκεται πίσω από κάποια διαδικασία. Μια βάση δεδομένων είναι μια οργανωμένη συλλογή των δεδομένων. Τα δεδομένα οργανώνονται συνήθως ώστε να μοντελοποιήσουν διάφορες πτυχές της καθημερινότητας (για παράδειγμα, την διαθεσιμότητα των δωματίων σε ένα ξενοδοχείο), με τρόπο που να υποστηρίζονται οι διαδικασίες που απαιτούν την διαθέσιμη πληροφορία (για παράδειγμα, η εύρεση ενός ξενοδοχείου με τις κενές θέσεις).

Ο όρος «βάση δεδομένων» αναφέρεται στα δεδομένα αυτά κάθε αυτά καθώς και στις διάφορες δομές που υλοποιούνται για την αποθήκευσή τους. Αντίθετα, ο όρος «σύστημα διαχείρισης βάσης δεδομένων» - (ΣΔΒΔ) αναφέρεται στο λογισμικό που διαχειρίζεται αυτή την βάση δεδομένων[1]. Η βάση δεδομένων σε συνδυασμό με το ΣΔΒΔ ονομάζεται «σύστημα βάσης δεδομένων» και προϋποθέτει την ύπαρξη μίας μεθόδου διαχείρισης των δεδομένων που διέπεται από *ποιότητα* (ακρίβεια, διαθεσιμότητα, χρηστικότητα και ελαστικότητα). Το ΣΔΒΔ είναι συνήθως ένα σύνθετο λογισμικό που καλύπτει διάφορες απαιτήσεις χρήσης, έτσι ώστε να διατηρεί με τον καταλληλότερο τρόπο τις μεγάλες και σύνθετες δομές δεδομένων.

Η χρήση βάσεων δεδομένων είναι τόσο διαδεδομένη σήμερα που δεν θα ήταν υπερβολή να πούμε ότι κάθε τεχνολογία και προϊόν βασίζεται σε βάσεις δεδομένων και ΣΔΒΔ τόσο για την ανάπτυξη όσο και για την διάθεσή του στην αγορά. Πολλά συστήματα έχουν ακόμα και ενσωματωμένες βάσεις δεδομένων, ενώ κάθε είδους επιχείρηση και εταιρία βασίζονται αποκλειστικά σε κάποια βάση δεδομένων για την ομαλή λειτουργία τους.

1.1 Κίνητρα και στόχοι της εργασίας

Στην παρούσα εργασία γίνεται μία πλήρης μελέτη για την δημιουργία ενός on-line συστήματος για την διαχείριση ενός υποθετικού νοσοκομείου. Παράλληλα γίνεται ο σχεδιασμός και η υλοποίηση της ίδιας της βάσης δεδομένων. Οι στόχοι κατά την διάρκεια της εκπόνησης αυτής της εργασίας ήταν:

1. Να σχεδιαστεί μία βάση δεδομένων η οποία θα καλύψει όσο το δυνατών καλύτερα τις ιδιαίτερες πληροφοριακές ανάγκες ενός νοσοκομειακού περιβάλλοντος.

2. Να δημιουργηθεί ένα φιλικό προς το χρήστη σύστημα, μέσω του οποίου οι εργαζόμενοι (στη γραμματεία) του νοσοκομείου θα μπορούν εύκολα και γρήγορα να φέρουν εις πέρας συγκεκριμένα καθήκοντα.
3. Να υπάρχει δυνατότητα εύκολης και χωρίς προβλήματα επέκτασης τόσο της βάσης όσο και του συστήματος διαχείρισης αυτής.
4. Να χρησιμοποιηθεί η καλύτερη τρέχουσα τεχνολογία τόσο στην δημιουργία της βάσης, όσο και στο σύστημα διαχείρισης της.

Τα παραπάνω είναι σημαντικά στοιχεία που θα οδηγήσουν σε ένα άρτιο και σύμφωνο με τα ψηλότερα διαθέσιμα πρότυπα σύστημα βάσης δεδομένων. Είναι γεγονός ότι παρόλη την εκτεταμένη χρήση των βάσεων δεδομένων ανά τον κόσμο, τα ελληνικά (δημόσια) νοσοκομεία δεν έχουν ακόμα εξοπλιστεί πλήρως με τέτοιου είδους λογισμικό. Η μελέτη προς αυτή την κατεύθυνση θα μπορούσε να προσφέρει σημαντική επίγνωση του πως θα μπορούσε να γίνει με τον ευκολότερο και γρηγορότερο τρόπο η πλήρης ψηφιοποίηση του Ελληνικού συστήματος υγείας.

1.2 Δομή της εργασίας

Η εργασία αυτή έχει την ακόλουθη δομή. Στο Κεφάλαιο 2, γίνεται μία σύντομη επισκόπηση των μεθόδων και εργαλείων που έχουν χρησιμοποιηθεί για την δημιουργία του συστήματος διαχείρισης ενός νοσοκομείου. Επιπλέον παρουσιάζονται ορισμένες σημαντικές αρχές για την σωστή σχεδίαση και ανάπτυξη τόσο του ΣΔΒΔ όσο και την βάσης δεδομένων.

Στο Κεφάλαιο 3 παρουσιάζεται το μοντέλο οντοτήτων σχέσεων, καθώς και η μετατροπή του στο σχεσιακό μοντέλο. Αρχικά γίνεται μία εκτεταμένη αναφορά στα πιο σημαντικά τμήματα της αντίστοιχης θεωρίας. Στη συνέχεια παρουσιάζεται το E-R μοντέλο του συστήματος και αναλύεται η αναπαράσταση των απαιτήσεων μέσω αυτού του μοντέλου. Επιπλέον, περιγράφονται οι κανόνες μετατροπής ενός μοντέλου E-R στο σχεσιακό, και εφαρμόζονται στο μοντέλο που δημιουργήθηκε. Προσφέρεται μία διεξοδική ανάλυση των πινάκων της βάσης, των πεδίων του κάθε πίνακα, καθώς και των γνωρισμάτων, ιδιοτήτων του κάθε ορίσματος. Το σύστημα μετατρέπεται σε κανονική μορφή και επίσης παρουσιάζεται ο κώδικας SQL που χρησιμοποιήθηκε για να δημιουργηθεί η βάση.

Στο κεφάλαιο 4 δίνεται το εγχειρίδιο χρήσης του συστήματος. Για κάθε μία από τις ισχυρές οντότητες του συστήματος, παρουσιάζονται οι λειτουργίες που προσφέρονται, ο

τρόπος που αυτές οι λειτουργίες μπορούν να χρησιμοποιηθούν, ενώ παρέχονται στιγμιότυπα από την εκτέλεση και ο αντίστοιχος HTML/PHP κώδικας.

Τέλος, στο κεφάλαιο 5, το οποίο είναι και το τελευταίο κεφάλαιο της εργασίας παρατίθενται τα συμπεράσματα από την συγκεκριμένη υλοποίηση. Επιπλέον αναφέρονται κάποια σημαντικά θέματα σχετικά με την μελλοντική εργασία και επέκταση αυτού το συστήματος.

Κεφάλαιο 2: Βιβλιογραφική επισκόπηση

2.1 Η δημιουργία των βάσεων δεδομένων – Ιστορική αναδρομή

Η ιδέα για την δημιουργία βάσεων δεδομένων άρχισε να αναπτύσσεται την δεκαετία του 60 με στόχο να ξεπεραστούν οι αυξανόμενες δυσκολίες στον σχεδιασμό, ανάπτυξη και διατήρηση των σύνθετων πληροφοριακών συστημάτων. Φυσικά, η ιδέα αναπτύχθηκε σε συνδυασμό με την δημιουργία των πρώτων ΣΔΒΔ τα οποία έκαναν δυνατή την διαχείριση των βάσεων δεδομένων. Παρόλο που οι όροι βάση δεδομένων και ΣΔΒΔ αναφέρονται σε διαφορετικές οντότητες είναι σημαντικό να τονίσουμε ότι είναι και τα δύο αναπόσπαστα τμήματα ενός συστήματος βάσης δεδομένων και δεν μπορεί να υπάρξει το ένα χωρίς το άλλο.

Πρώτη αναφορά στον όρο βάση δεδομένων γίνεται από μία τεχνική αναφορά του 1962 [2]. Έκτοτε, με την τρομερή πρόοδο στους επεξεργαστές, τις μνήμες, τις συσκευές αποθήκευσης και τα δίκτυα, παρατηρήθηκε μία τεράστια αύξηση στο μέγεθος, τις δυνατότητες και την επίδοση των βάσεων δεδομένων. Εδώ και δεκαετίες είναι απίθανο να γίνει η ανάπτυξη ενός σύνθετου πληροφοριακού συστήματος που δεν περιλαμβάνει μία κατάλληλη βάση δεδομένων και το αντίστοιχο ΣΔΒΔ.

Δεν υπάρχει ένας ευρέως αποδεκτός και ακριβής ορισμός όσον αφορά τα ΣΔΒΔ. Ωστόσο, ένα σύστημα πρέπει να παρέχει σημαντική λειτουργικότητα για να χαρακτηριστεί ως ένα ΣΔΒΔ. Ως εκ τούτου η υποστηριζόμενη συλλογή δεδομένων, θα πρέπει να πληροί τις αντίστοιχες απαιτήσεις ευχρηστίας (σε μεγάλο βαθμό καθορίζεται από τις απαιτήσεις χρήσης) ώστε να μπορεί να χαρακτηρισθεί ως μια βάση δεδομένων.

Στα πρώτα συστήματα βάσεων δεδομένων, η βασική ανάγκη ήταν η καλή τους απόδοση. Ωστόσο, είχε ήδη αναγνωριστεί ότι υπάρχουν και άλλα σημαντικά ζητούμενα. Από τις αρχές ήταν βασικός στόχος να γίνουν τα δεδομένα ανεξάρτητα από την λογική του εκάστοτε προγράμματος, ώστε τα ίδια δεδομένα να είναι διαθέσιμα για διαφορετικές λειτουργίες.

2.2 Μοντέλα συστημάτων βάσεων δεδομένων

2.2.1 Μοντέλο πλοήγησης

Η πρώτη γενιά ΣΒΔ ήταν συστήματα *πλοήγησης*. Οι εφαρμογές είχαν πρόσβαση στα δεδομένα ακλουθώντας δείκτες από μία εγγραφή σε κάποια άλλη. Τα δύο βασικά συστήματα αυτού του τύπου ήταν τα *ιεραρχικά μοντέλα,* με βασικό παράδειγμα το σύστημα IMS της

IBM, και το μοντέλο *Codasyl – δικτυακό μοντέλο,* το οποίο υλοποιήθηκε σε πολλά προϊόντα, όπως για παράδειγμα το IDMS.

2.2.2 Σχεσιακό μοντέλο

Το *σχεσιακό μοντέλο*, προτάθηκε πρώτα το 1970 από τον E.F. Codd [3][4], ο οποίος κατάλαβε ότι οι εφαρμογές θα έπρεπε να αναζητούν δεδομένα με βάση το περιεχόμενο και όχι ακολουθώντας συνδέσεις μεταξύ των εγγραφών. Αυτό θεωρήθηκε απαραίτητο και ώστε να επιτραπεί η αλλαγή των δεδομένων της βάσης χωρίς να απαιτείται η συνεχής ανανέωση της εκάστοτε εφαρμογής. Καθώς όμως τα σχεσιακά συστήματα έχουν σημαντικές απαιτήσεις σε επεξεργασία πληροφορίας, μέχρι τα μέσα της δεκαετίας του 80, δεν υπήρχαν αρκετή υπολογιστική ισχύς ώστε να αρχίσουν να χρησιμοποιούνται ευρέως. Μέσα σε μία δεκαετία τα σχεσιακά μοντέλα έγιναν κυρίαρχα σε κάθε είδους επεξεργασία δεδομένων μεγάλης κλίμακας και μέχρι σήμερα παραμένουν βασικό εργαλείο. Η κυρίαρχη γλώσσα προγραμματισμού βάσεων δεδομένων για το σχεσιακό μοντέλο είναι η SQL. Στο σχεσιακό μοντέλο, όλα τα δεδομένα αντιπροσωπεύονται από πλειάδες και ομαδοποιούνται σε σχέσεις.

Η βασική υπόθεση του σχεσιακού μοντέλου είναι ότι όλα τα δεδομένα μπορούν να αναπαρασταθούν με την μορφή ν-αδικών σχέσεων. Μία ν-αδική σχέση είναι ένα υποσύνολο του Καρτεσιανού γινομένου ν συντελεστών. Στο μαθηματικό μοντέλο, ο υπολογισμός της τιμής τέτοιων δεδομένων γίνεται με δυαδική κατηγορηματική λογική. Επομένως, για κάθε πρόταση υπάρχουν δύο πιθανές τιμές, αληθής ή ψευδής.

Το σχεσιακό μοντέλο επιτρέπει στον σχεδιαστή μίας βάσης δεδομένων να δημιουργήσει μία λογική και συνεπή αναπαράσταση της πληροφορίας. Η συνέπεια επιτυγχάνεται με την περίληψη δηλωμένων περιορισμών κατά την σχεδίαση της βάση, οι οποίοι συχνά ονομάζονται λογικό σχήμα. Η θεωρία περιλαμβάνει μία διαδικασία κανονικοποιήσης της βάσης όπου μία αρχιτεκτονική με συγκεκριμένες επιθυμητές ιδιότητες μπορεί να επιλεχθεί από ένα σύνολο λογικά ισοδύναμων εναλλακτικών. Τα σχέδια πρόσβασης και άλλες λεπτομέρειες της υλοποίησης και λειτουργίας διαχειρίζονται από το ΣΔΒΔ, και δεν εμφανίζονται στο λογικό μοντέλο. Αυτό έρχεται σε αντίθεση με την κοινή πρακτική για SQL ΣΔΒΔ, όπου η ρύθμιση απόδοσης απαιτεί συχνά αλλαγές στο λογικό μοντέλο. Το βασικό δομικό στοιχείο της σχεσιακής λογικής είναι ο τύπος δεδομένων πεδίου ή απλά τύπος. Μια πλειάδα είναι μια διατεταγμένη σειρά από τιμές χαρακτηριστικών. Ένα χαρακτηριστικό είναι ένα διατεταγμένο ζεύγος από το όνομα του χαρακτηριστικού και από το όνομα του τύπου. Η τιμή του χαρακτηριστικού είναι μια συγκεκριμένη τιμή που ισχύει για

τον συγκεκριμένο τύπο του χαρακτηριστικού. Η τιμή μπορεί να είναι βαθμωτή ή από έναν πιο σύνθετο τύπο. Μια σχέση αποτελείται από την ***επικεφαλίδα*** και το ***σώμα***. Η επικεφαλίδα είναι ένα σύνολο χαρακτηριστικών ενώ το σώμα μίας ν-αδικής σχέσης είναι ένα σύνολο από ν-πλειάδων. Ο τίτλος της σχέσης είναι επίσης ο τίτλος της κάθε πλειάδας. Μία σχέση ορίζεται ως ένα σύνολο από ν-πλειάδες. Τόσο στο μαθηματικό μοντέλο όσο και στο σχεσιακό μοντέλο βάσεων δεδομένων, σύνολο ονομάζεται μία μη διατεταγμένη συλλογή από μοναδικά αντικείμενα (χωρίς επαναλήψεις).

2.2.3 Το μοντέλο οντοτήτων σχέσεων

Το σχεσιακό μοντέλο δίνει έμφαση στην αναζήτηση δεδομένων και όχι στην πλοήγηση ανάμεσά τους. Δεν δημιουργεί σχέσεις μεταξύ διαφορετικών οντοτήτων ορατές με την μορφή των δεικτών αλλά αναπαριστά τις σχέσεις με την χρήση των πρωτευόντων και ξένων κλειδιών. Αυτή η ιδιότητα αποτελεί μία πολύ καλή βάση για μία γλώσσα επερωτήσεων (όπως είναι για παράδειγμα η SQL) αλλά όχι για μία γλώσσα μοντελοποίησης. Για αυτό τον λόγο το μοντέλο οντοτήτων-σχέσεων, που προέκυψε λίγο καιρό μετά την διάδοση του σχεσιακού μοντέλου [5], έγινε γρήγορα ιδιαίτερα δημοφιλές για τον σχεδιασμό βάσεων δεδομένων.

2.3 Στατικές vs Δυναμικές Ιστοσελίδες

Ένα σύστημα για να ονομάζεται on-line σύστημα προϋποθέτει την ύπαρξη μιας ιστοσελίδας. Όπως ήδη αναφέρθηκε, μέρος της εργασίας απαιτεί την δημιουργία ενός web interface για την διαχείριση της βάσης δεδομένων. Σε αυτή την ενότητα παρουσιάζονται οι 2 μεγάλες κατηγορίες ιστοσελίδων, οι στατικές και οι δυναμικές.

2.3.1 Στατικές Ιστοσελίδες

Οι στατικές ιστοσελίδες είναι ουσιαστικά απλά ηλεκτρονικά έγγραφα τα οποία περιέχουν στατικό περιεχόμενο όπως κείμενα, συνδέσμους, εικόνες κτλ. Με τον όρο στατικό εννοούμε ότι το περιεχόμενο της κάθε σελίδας είναι σταθερό και συγκεκριμένο. Αυτή η κατηγορία ιστοσελίδων χρησιμοποιείται συνήθως για την δημιουργία μόνιμων/στατικών παρουσιάσεων όπου δεν υπάρχει συχνά η ανάγκη να τροποποιείται το περιεχόμενο τους [6].

Τα πλεονεκτήματα των στατικών ιστοσελίδων είναι πως δημιουργούνται πολύ εύκολα, γρήγορα και με χαμηλό κόστος. Τα μειονεκτήματα τους είναι πως απαιτείται η παρέμβαση προγραμματιστή για την ενημέρωση τους και ότι υστερούνε τεχνολογικά.

2.3.2 Δυναμικές Ιστοσελίδες

Μια δυναμική ιστοσελίδα παρέχει περισσότερες δυνατότητες αφού συνθέτουν μια εφαρμογή και όχι ένα απλό ηλεκτρονικό έγγραφο. Αν και η ανάπτυξη τους είναι πιο χρονοβόρα και στοιχίζουν πιο πολλά από τις στατικές ιστοσελίδες, τα πλεονεκτήματα τους είναι πολλά:

- Πολύ πιο λειτουργική ιστοσελίδα
- Εύκολη ενημέρωση
- Φιλικές προς τις μηχανές αναζήτησης
- Η διαχείριση τους δεν απαιτεί την βοήθεια ειδικού προγραμματιστή.

Συνήθως οι δυναμικές ιστοσελίδες χρησιμοποιούν κάποια βάση δεδομένων, όπου αποθηκεύουν πληροφορίες και από την οποία αντλούν το περιεχόμενο τους, αντιδρώντας σε διάφορα events από την αλληλεπίδραση με τον χρήστη. Η χρήση των βάσεων δεδομένων, είναι αυτή που επιτρέπει την εύκολη προσθαφαίρεση περιεχομένου στις δυναμικές ιστοσελίδες, καθώς δεν απαιτείται να επεξεργάζεται κανείς κάθε φορά την ίδια την ιστοσελίδα, αλλά απλά να διαχειρίζεται έμμεσα το περιεχόμενο στην βάση δεδομένων και οι υπόλοιπες διαδικασίες γίνονται αυτοματοποιημένα από τον "μηχανισμό" της ιστοσελίδας [7].

2.4 Εργαλεία Ανάπτυξης

Σε αυτή την ενότητα παρουσιάζονται τα εργαλεία που χρησιμοποιήθηκαν στον σχεδιασμό και την υλοποίηση της εφαρμογής.

2.4.1 SmartDraw

Το SmartDraw είναι ένα λογισμικό το οποίο χρησιμοποιείται για την δημιουργία επαγγελματικών γραφικών όπως διαγράμματα ροής (flowcharts), οργανογράμματα, διαγράμματα UML, χρονοδιαγράμματα και άλλα πολλά. Το εργαλείο αυτό χρησιμοποιήθηκε στην δημιουργία του E-R μοντέλου του συστήματος.

2.4.2 XAMPP

Το XAMPP είναι ένα ελεύθερο πακέτο λογισμικού το οποίο περιέχει τον εξυπηρετητή HTTP Apache, την βάση δεδομένων MySQL, το εργαλείο PhpMyAdmin και επίσης ένα διερμηνέα για προγράμματα γραμμένα σε Php και Perl.

Apache HTTP server

Πρόκειται για έναν εξυπηρετητή (server) του παγκόσμιου Ιστού (Web). Ο ρόλος του Apache είναι να αναμένει αιτήσεις από διάφορα προγράμματα – χρήστες (clients) όπως είναι ο browser ενός χρήστη και στη συνέχεια να εξυπηρετεί αυτές τις αιτήσεις "σερβίροντας" τις σελίδες που ζητούν είτε απευθείας μέσω μιας ηλεκτρονικής διεύθυνσης (URL), είτε μέσω ενός συνδέσμου (link). Ο τρόπος με τον οποίο ο Apache εξυπηρετεί αυτές τις αιτήσεις, είναι σύμφωνος με τα πρότυπα που ορίζει το πρωτόκολλο HTTP Hypertext Transfer Protocol.

PhpMyAdmin

Το PhpMyAdmin είναι ένα εργαλείο ανοιχτού κώδικα, γραμμένο σε PHP που χρησιμοποιείται για την διαχείριση βάσεων δεδομένων σε MySQL. Το εργαλείο αυτό υποστηρίζει ένα ευρύ φάσμα εργασιών όπως [8]:

- δημιουργία βάσεων δεδομένων από SQL αρχείο
- εξαγωγή μιας βάσης δεδομένων σε SQL αρχείο
- εκτέλεση εντολών SQL
- διαχείριση μιας βάσης δεδομένων (δημιουργία και τροποποίηση πινάκων, πεδίων, γραμμών κτλ)
- διαχείριση χρηστών και δικαιωμάτων.
- διαχείριση πολλαπλών server και άλλα.

Το PhpMyAdmin χρησιμοποιήθηκε στην δημιουργία και διαχείριση της βάσης δεδομένων του συστήματος.

2.4.3 MySQL workbench

Το MySQL Workbench είναι ένα οπτικό εργαλείο σχεδιασμού της βάσης δεδομένων SQL που ενσωματώνει την ανάπτυξη, διοίκηση, σχεδιασμό βάσεων δεδομένων, τη δημιουργία και τη συντήρηση σε ένα ενιαίο ολοκληρωμένο περιβάλλον ανάπτυξης για το σύστημα της βάσης δεδομένων MySQL [9]. Το εργαλείο αυτό χρησιμοποιήθηκε στον σχεδιασμό και την γραφική αναπαράσταση της βάσης δεδομένων της εφαρμογής, και την παραγωγή των εντολών SQL για την δημιουργία της βάσης στο PhpMyAdmin.

2.4.4 Adobe Dreamweaver CS6

Το Dreamweaver είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης που χρησιμοποιείται κυρίως για το σχεδιασμό και την ανάπτυξη ιστοσελίδων σε HTML και CSS. Επίσης υποστηρίζει ένα ευρύ φάσμα γλωσσών προγραμματισμού από πλευράς server όπως PHP, JSP, ASP. Το Dreamweaver είναι ένα WYSIWYG (What You See Is What You Get) περιβάλλον που επιτρέπει στους προγραμματιστές να γράφουν κώδικα και να παράλληλα να βλέπουν τις ιστοσελίδες που δημιουργήθηκαν σε πραγματικό χρόνο ή να δημιουργήσουν μία ιστοσελίδα σε ένα γραφικό περιβάλλον κάνοντας drag and drop από μία παλέτα HTML στοιχείων. Το CS6 είναι απλά η τελευταία έκδοση του Dreamweaver και είναι το εργαλείο που χρησιμοποιήθηκε για την δημιουργία του web interface της εφαρμογής.

2.5 Τεχνολογίες

Σε αυτήν την ενότητα περιγράφονται οι τεχνολογίες και γλώσσες προγραμματισμού που χρησιμοποιήθηκαν στην υλοποίηση της εφαρμογής.

2.5.1 HTML5

Η HTML5 είναι μια γλώσσα σήμανσης για τη δόμηση και την παρουσίαση περιεχόμενου για το World Wide Web και μια βασική τεχνολογία του Διαδικτύου. Είναι η πέμπτη αναθεώρηση του προτύπου HTML το οποίο είναι ακόμη υπό ανάπτυξη [10]. Μερικά από τα πιο ενδιαφέροντα νέα χαρακτηριστικά της HTML5 είναι τα εξής:

- νέα form controls όπως calendar, date, time, email, url, search. Μέχρι τώρα ήταν κοινή τακτική η χρησιμοποίηση Javascript/AJAX για τον έλεγχο της ημερομηνίας και ώρας (date picker fields). Πλέον με την HTML5 δεν υπάρχει η ανάγκη για κάτι τέτοιο.
- καλύτερη διαχείριση λαθών
- τα στοιχεία <video> και <audio> για αναπαραγωγή πολυμέσων. Με αυτά τα νέα στοιχεία περιορίζεται η ανάγκη χρησιμοποιήσεις εξωτερικών τεχνολογιών όπως Flash.
- Το στοιχείο <canvas> για 2D drawing
- Νέο ειδικά στοιχεία περιεχομένου, όπως <Article>, <footer>, <header>, <nav>, <section>
- νέα INPUT types και attibutes για επικύρωση της εισόδου από τον χρήστη (form validation) εξαλείφοντας και πάλι την ανάγκη χρήσης JavaScript.

Ο μόνος λόγος για την μην χρησιμοποίηση της HTML5 είναι πως ακόμη δεν υποστηρίζεται πλήρως από τα πρόγραμμα περιήγησης στο διαδίκτυο (web browsers). Αυτό όμως είναι θέμα χρόνου να ξεπεραστεί αφού όλοι οι μεγάλοι browsers (Safari, Chrome, Firefox, Opera, Internet Explorer) συνεχίζουν να προσθέτουν νέα χαρακτηριστικά HTML5 στις τελευταίες εκδόσεις τους [11].

Συνιστάται η χρήση του Google Chrome για την εφαρμογή αυτή και για όλες όσες χρησιμοποιούν HTML5 αφού από διάφορες δοκιμές φάνηκε πως μέχρι τώρα, υποστηρίζει την HTML5 καλύτερα από οποιοδήποτε άλλο browser.

2.5.2 PHP

Η PHP είναι ένα μία server-side scripting γλώσσα που αρχικά σχεδιάστηκε για την παραγωγή δυναμικών ιστοσελίδων. Είναι μία από τις πρώτες server-side scripting γλώσσες που ενσωματώθηκαν σε ένα έγγραφο HTML έτσι ώστε να μην απαιτείται η κλήση ενός εξωτερικού αρχείου για την επεξεργασία των δεδομένων. Η PHP υποστηρίζει ένα ευρύ φάσμα βάσεων δεδομένων όπως Oracle, PostgreSQL, ODBC, MySQL (η βάση δεδομένων που χρησιμοποιήθηκε στην εφαρμογή) και άλλες.

Στην εφαρμογή της εργασίας, η PHP χρησιμοποιήθηκε για την επεξεργασία των δεδομένων που εισάγει ο χρήστης, την εισαγωγή τους στην βάση δεδομένων, την εκτέλεση επερωτήσεων SQL, την ανάκτηση δεδομένων και την παρουσίαση τους στις φόρμες του συστήματος.

2.5.3 CSS3

Η CSS3 είναι το τελευταίο πρότυπο της CSS (Cascading Style Sheets), μίας γλώσσας που χρησιμοποιείται για τον καθορισμό της εμφάνισης και την μορφοποίηση ενός εγγράφου που είναι γραμμένο σε μια γλώσσα σήμανσης. Η πιο κοινή εφαρμογή της είναι σε ιστοσελίδες γραμμένες σε HTML και XHTML, αλλά μπορεί επίσης να εφαρμοστεί σε οποιοδήποτε είδος εγγράφου XML, συμπεριλαμβανομένου του απλού XML, SVG και XUL.

Η CSS έχει σχεδιαστεί κυρίως για να επιτρέπει το διαχωρισμό του περιεχομένου ενός εγγράφου (γραμμένο σε HTML ή άλλη γλώσσα σήμανσης) από την παρουσίαση του. Με την χρήση της CSS μπορούμε να καθορίσουμε την διάταξη, τα χρώματα, τις γραμματοσειρές και άλλα στοιχεία παρουσίασης μίας ιστοσελίδας. Αυτός ο διαχωρισμός παρέχει μεγάλη ευελιξία αφού πολλές σελίδες μπορούν να μοιραστούν την μορφοποίηση του ίδιου CSS αρχείου. Η

CSS μπορεί επίσης να χρησιμοποιηθεί την εμφανίσει μίας ιστοσελίδας με διαφορετικό τρόπο ανάλογα με το μέγεθος της οθόνης ή συσκευής, από την οποία ανοίχτηκε [12].

2.5.4 JavaScript

Η JavaScript είναι μια γλώσσα προγραμματισμού με την οποία οι ιστοσελίδες μπορούν να αποκτήσουν μια πιο εκλεπτυσμένη λειτουργικότητα. Η πιο ευρεία χρήση της JavaScript είναι η δημιουργία μεθόδων που ενσωματώνονται ή εισάγονται από κάποια βιβλιοθήκη σε σελίδες HTML και που αλληλεπιδρούν με το Document Object Model (DOM) της σελίδας. Μερικά απλά παραδείγματα της χρήσης της ακολουθούν [13]:

- Τοποθέτηση νέου περιεχομένου στην σελίδας ή την υποβολή δεδομένων στο διακομιστή μέσω AJAX χωρίς να χρειάζεται η επαναφόρτωση της σελίδας
- Δημιουργία διαδραστικού περιεχομένου σε μία ιστοσελίδα
- Επικύρωση των τιμών εισόδου από μια ηλεκτρονική φόρμα για να επιβεβαιωθεί ότι είναι αποδεκτές πριν υποβληθούν στο διακομιστή.
- Απόκρυψη ή εμφάνιση διαφόρων στοιχείων (HTML elements) της σελίδας.
- Αλλαγή του τύπου των HTML elements και άλλα.

2.5.5 AJAX (Asynchronous JavaScript and XML)

Το AJAX είναι μια τεχνική για την γρήγορη δημιουργία δυναμικών ιστοσελίδων. Επιτρέπει στις ιστοσελίδες να ενημερώνεται ασύγχρονα με την ανταλλαγή μικρών ποσοτήτων δεδομένων με το διακομιστή πίσω από τις σκηνές. Αυτό σημαίνει ότι είναι δυνατόν να ενημερώνονται μέρη της ιστοσελίδας, χωρίς την επαναφόρτωση ολόκληρης τη σελίδα. Το AJAX βασίζεται σε πρότυπα διαδικτύου, και χρησιμοποιεί ένα συνδυασμό των πιο κάτω [14]:

- αντικείμενα XMLHttpRequest (για την ασύγχρονη ανταλλαγή δεδομένων με ένα διακομιστή)
- JavaScript / DOM (για την εμφάνιση / αλληλεπίδραση με τις πληροφορίες)
- CSS (στο στυλ των δεδομένων)
- XML (χρησιμοποιείται συχνά ως το σχήμα για τη μεταφορά δεδομένων)

Η τεχνολογία αυτή χρησιμοποιήθηκε για το real-time validation της εισόδου που δίνει ο χρήστης με την βάση, σε πεδία των οποίων η τιμή τους στην βάση πρέπει να μοναδική.

Κεφάλαιο 3: Μοντέλο οντοτήτων συσχετίσεων - μετατροπή σε σχεσιακό μοντέλο

3.1 Μοντέλο Οντοτήτων - Συσχετίσεων

3.1.1 Θεωρία

Το μοντέλο οντοτήτων συσχετίσεων ή αλλιώς E-R μοντέλο, είναι ένα εννοιολογικό μοντέλο το οποίο χρησιμοποιείται για να καταγράψει τις απαιτήσεις ενός νέου πληροφοριακού συστήματος με γραφικό τρόπο. Μία βάση δεδομένων μπορεί να μοντελοποιηθεί εννοιολογικά σαν:

1. Ένα σύνολο **οντοτήτων** που υποθέτουμε πώς υπάρχουν στο κόσμο ενδιαφέροντος μας.
2. Ένα σύνολο **συσχετίσεων** μεταξύ των οντοτήτων αυτών
3. Ένα σύνολο **περιορισμών** σχετικά με τον τρόπο με τον οποίο οι οντότητες συμμετέχουν σε σχέσεις.

Στα επόμενα υποκεφάλαια δίνεται μία συνοπτική περιγραφή όλων των στοιχείων από τα οποία αποτελείται ένα μοντέλο οντοτήτων σχέσεων.

Οντότητες

Μία οντότητα (ή ένα σύνολο οντοτήτων) είναι ένα υπαρκτό αντικείμενο, που είναι διακριτό από τα άλλα αντικείμενα της βάσης. Οι οντότητες χαρακτηρίζονται από τις εξής ιδιότητες:

1. Μπορούν να αντιστοιχούν σε αντικείμενα με φυσική ή αφηρημένη υπόσταση.
2. Μπορούν να έχουν πολλά στιγμιότυπα.

Κάθε οντότητα σε ένα E-R μοντέλο ανήκει σε μία από τις πιο κάτω κατηγορίες:

1. **Ισχυρές Οντότητες:** θεωρείται ως ο κανονικός τύπος οντοτήτων που έχουν ένα γνώρισμα που αναγνωρίζει που αναγνωρίζει μοναδικά κάθε στιγμιότυπο της οντότητας αυτής. (κλειδί)
2. **Ασθενείς Οντότητες:** μία οντότητα E1 λέγεται ασθενής αν η ύπαρξη των στιγμιότυπων της εξαρτάται από μια άλλη οντότητα E2 μέσω μίας σχέσης R. Η E2 λέγεται ισχυρή οντότητα. Γενικά για μία ασθενή οντότητα ισχύουν τα πιο κάτω:

- Η ύπαρξη της εξαρτάται από μία άλλη οντότητα
- Δεν υπάρχει η ύπαρξη της χωρίς την οντότητα από την οποία εξαρτάται, η οποία ονομάζεται ορίζουσα οντότητα.
- Χρειάζεται γνώρισμα από την ορίζουσα (ισχυρή) οντότητα για την πλήρη ταυτοποίηση της, πέρα από το δικό της γνώρισμα που όμως την προσδιορίζει μερικώς,
- Μπορεί να είναι κάτοχος άλλων ασθενών οντοτήτων
- Μπορεί να σχετίζεται με περισσότερες από μία ισχυρές οντότητες μέσω διαφορετικών σχέσεων
- Στο E-R μοντέλο απεικονίζεται με ορθογώνιο με διπλό περίγραμμα.

Γνώρισμα

Ένα γνώρισμα (attribute) είναι μία περιγραφή μιας ιδιότητας η οποία έχει τιμή και αποδίδεται σε μία οντότητα. Τα στιγμιότυπα μιας οντότητας έχουν ένα κοινό σύνολο γνωρισμάτων. Ένα υποσύνολο των γνωρισμάτων μιας οντότητας χρησιμοποιείται ως αναγνωριστικό, των οποίων ο συνδυασμός τιμών είναι μοναδικός για κάθε στιγμιότυπο της οντότητας.

Τα γνωρίσματα ανήκουν σε μία από τις εξής κατηγορίες:

1. **Απλά:** δέχονται απλές τιμές από κάποιο πεδίο τιμών
2. **Σύνθετα:** γνωρίσματα τα οποία αποτελούνται από ένα αριθμό γνωρισμάτων τα οποία σαν σύνολο περιγράφουν μια ιδιότητα.

Επίσης τα γνωρίσματα διακρίνονται σε **μονότιμα** και **πλειότιμα.**

Συσχετίσεις

Μία συσχέτιση ορίζεται ως μία εννοιολογική σύνδεση κάποιων οντοτήτων. Τυπικά δεδομένου ενός συνόλου οντοτήτων E1, E2…En, μία συσχέτιση R ορίζει την αντιστοίχηση μεταξύ των στιγμιότυπων των οντοτήτων αυτών.

Ο αριθμός των οντοτήτων που συμμετέχουν σε μία συσχέτιση ορίζει και τον βαθμό n της συσχέτιση αυτής. Πχ μία συσχέτιση με βαθμό n=2 ονομάζεται δυαδική, με n=3 τριαδική κλπ. Επίσης, μία συσχέτιση είναι δυνατόν να έχει τα δικά της γνωρίσματα.

Πληθυκότητες

Κάθε οντότητα συμμετέχει σε μία συσχέτιση με μία δεδομένη ελάχιστη (min) και μέγιστη (max) πληθικότητα. Οι πληθικότητες των συσχετίσεων ορίζονται κατά τον

σχεδιασμό μίας βάσης δεδομένων και ο ρόλος τους είναι να περιορίζουν τους τρόπους με τους οποίους στιγμιότυπα των οντοτήτων συμμετέχουν σε στιγμιότυπα συσχετίσεων.

Έστω ότι έχουμε δύο οντότητες E, F που συμμετέχουν σε μία συσχέτιση R. Ανάλογα με τον τρόπο που οι οντότητες συμμετέχουν στην συσχέτιση αυτή, έχουμε τους πιο κάτω κανόνες:

- Αν η μέγιστη πληθυκότητα της οντότητας E στην R είναι 1, τότε λέμε πως η E έχει **μονότιμη συμμετοχή** στην R.
- Αν η μέγιστη πληθυκότητα της οντότητας E στην R είναι N, τότε λέμε πως η E έχει **πλειότιμη συμμετοχή** στην R.
- Αν η ελάχιστη συμμετοχή της οντότητας E στην R είναι 1, τότε λέμε πως η E έχει **υποχρεωτική συμμετοχή** στην R.
- Αν η ελάχιστη συμμετοχή της οντότητας E στην R είναι 0, τότε λέμε πως η E έχει **προαιρετική συμμετοχή** στην R.
- Μία δυαδική συσχέτιση R μεταξύ των οντοτήτων E,F ονομάζεται **συσχέτιση «πολλά προς πολλά» (N-N)** αν και οι δύο οντότητες έχουν πλειότιμη συμμετοχή στην R.
- Μία δυαδική συσχέτιση R μεταξύ των οντοτήτων E,F ονομάζεται **συσχέτιση «ένα προς ένα» (1-1)** αν και οι δύο οντότητες έχουν μονότιμη συμμετοχή στην R.
- Μία δυαδική συσχέτιση R μεταξύ των οντοτήτων E,F ονομάζεται **συσχέτιση «ένα προς πολλά» (1-N)** αν η E έχει μονότιμη και η F πλειότιμη συμμετοχή στην συσχέτιση αυτή.

  Οι πληθυκότητες συμβολίζονται σαν ζεύγη τιμών πάνω στις γραμμές οι οποίες ενώνουν τις οντότητες με τις συσχετίσεις.

Εξειδίκευση

Εξειδίκευση ονομάζεται η διαδικασία προσδιορισμού υπό-ομάδων μέσα σε σύνολα οντοτήτων. Μία οντότητα μπορεί να περιλαμβάνει υπό-ομάδες οντοτήτων οι οποίες διακρίνονται από άλλες οντότητες στην ίδια ομάδα, και χαρακτηρίζονται από γνωρίσματα τα οποία δεν χαρακτηρίζουν όλες τις οντότητες στο σύνολο. Η χρήση της εξειδίκευσης σε ένα μοντέλο οντοτήτων συσχετίσεων δημιουργεί ιεραρχίες εξειδίκευσης με την χρήση της **συσχέτισης «είναι υπό-ομάδα» (IsA).** Η συσχέτιση **IsA** ορίζει μία σχέση υπέρκλασης-υποκλάσης

Γενίκευση

Η γενίκευση είναι μία απλή αναστροφή της εξειδίκευσης. Η εξειδίκευση οντοτήτων σε υπό-ομάδες αντιστοιχεί σε μία top-down διαδικασία σχεδιασμού ενός εννοιολογικού μοντέλου. Ο σχεδιασμός αυτός μπορεί να γίνει και bottom-up όπου οι οντότητες χρησιμοποιούνται για να συνθέσουν άλλες οντότητες σε υψηλότερα επίπεδα, βάσει των κοινών γνωρισμάτων των οντοτήτων. Η διαδικασία αυτή ονομάζεται γενίκευση.

Κληρονομικότητα Γνωρισμάτων

Όταν οντότητες οργανώνονται σε ιεραρχίες εξειδίκευσης/γενίκευσης, τα γνωρίσματα των οντοτήτων που βρίσκονται στα υψηλότερα επίπεδα κληρονομούνται από τις οντότητες που βρίσκονται σε χαμηλότερα επίπεδα. Επίσης κληρονομούνται οι συμμετοχές σε συσχετίσεις με τους ίδιους περιορισμούς.

Οι συσχετίσεις εξειδίκευσης/γενίκευσης υπόκεινται στους πιο κάτω περιορισμούς που αφορούν τα στιγμιότυπα των οντοτήτων που συμμετέχουν σε αυτές:

1. **Αποκλειστικότητα:** Ένα στιγμιότυπο δεν μπορεί να ανήκει σε περισσότερες από μία οντότητες στο ίδιο επίπεδο μιας ιεραρχίας IsA
2. **Επικάλυψη**: το ίδιο στιγμιότυπο μπορεί να ανήκει σε περισσότερες από μία οντότητες σε μία ιεραρχία.
3. **Πληρότητα:** καθορίζει αν ένα στιγμιότυπο μιας οντότητας πρέπει να ανήκει σε τουλάχιστο μία οντότητα σε χαμηλότερο επίπεδο.

Οι περιορισμοί πληρότητας μπορεί να είναι:

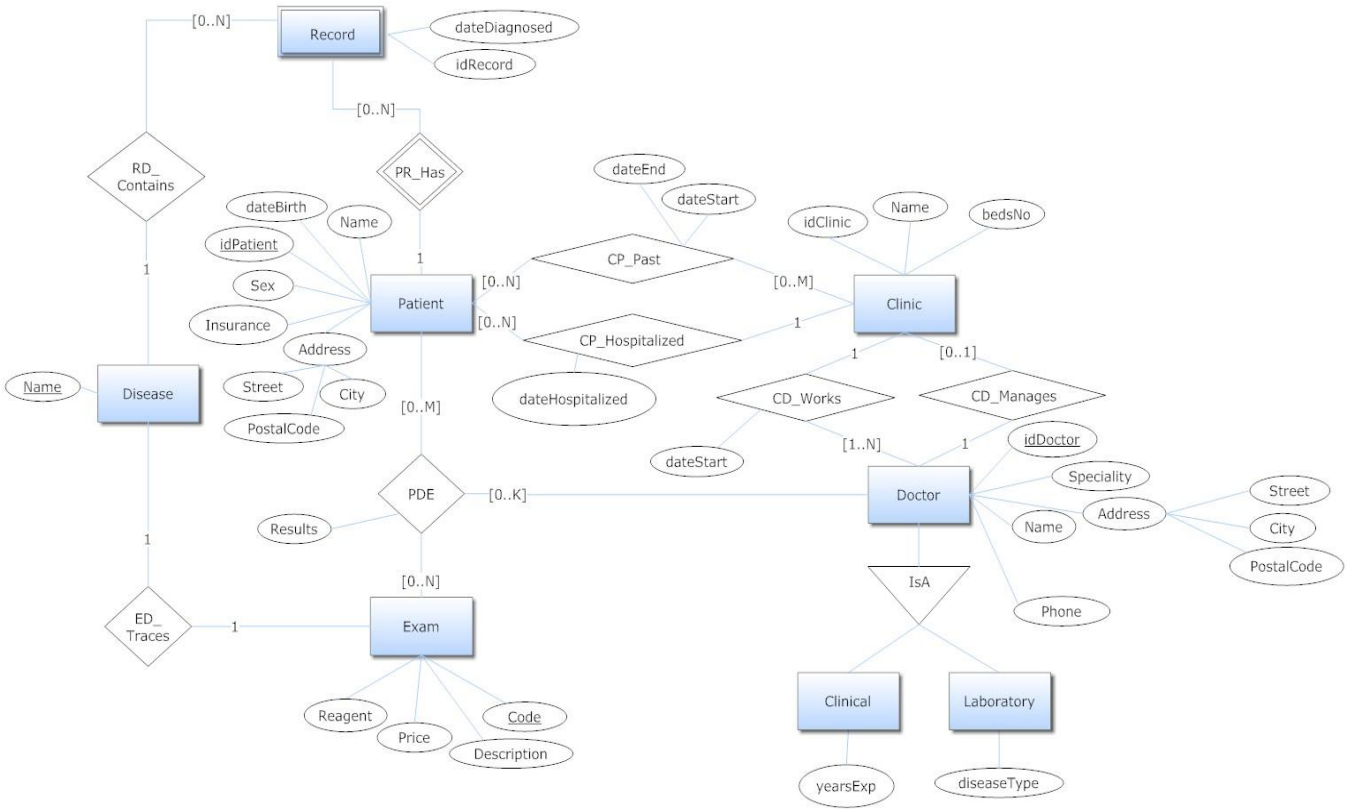
- **Ολικοί**: κάθε στιγμιότυπο πρέπει να ανήκει σε μία οντότητα σε χαμηλότερο επίπεδο. Σε αυτή την περίπτωση, όποτε εισάγεται ένα στιγμιότυπο ως μέλος μιας οντότητας, πρέπει να εισαχθεί και ως μέλος μιας οντότητας σε χαμηλότερο επίπεδο. Επίσης, η διαγραφή ενός στιγμιότυπου πρέπει να συνοδεύεται από την διαγραφή του από όλες τις οντότητες χαμηλότερου επιπέδου στις οποίες ανήκει.
- **Μερικοί**: κάποια στιγμιότυπα μπορούν να μην ανήκουν σε κάποια από τις οντότητες σε χαμηλότερα επίπεδα.

3.1.2 Εφαρμογή του μοντέλου οντοτήτων-συσχετίσεων

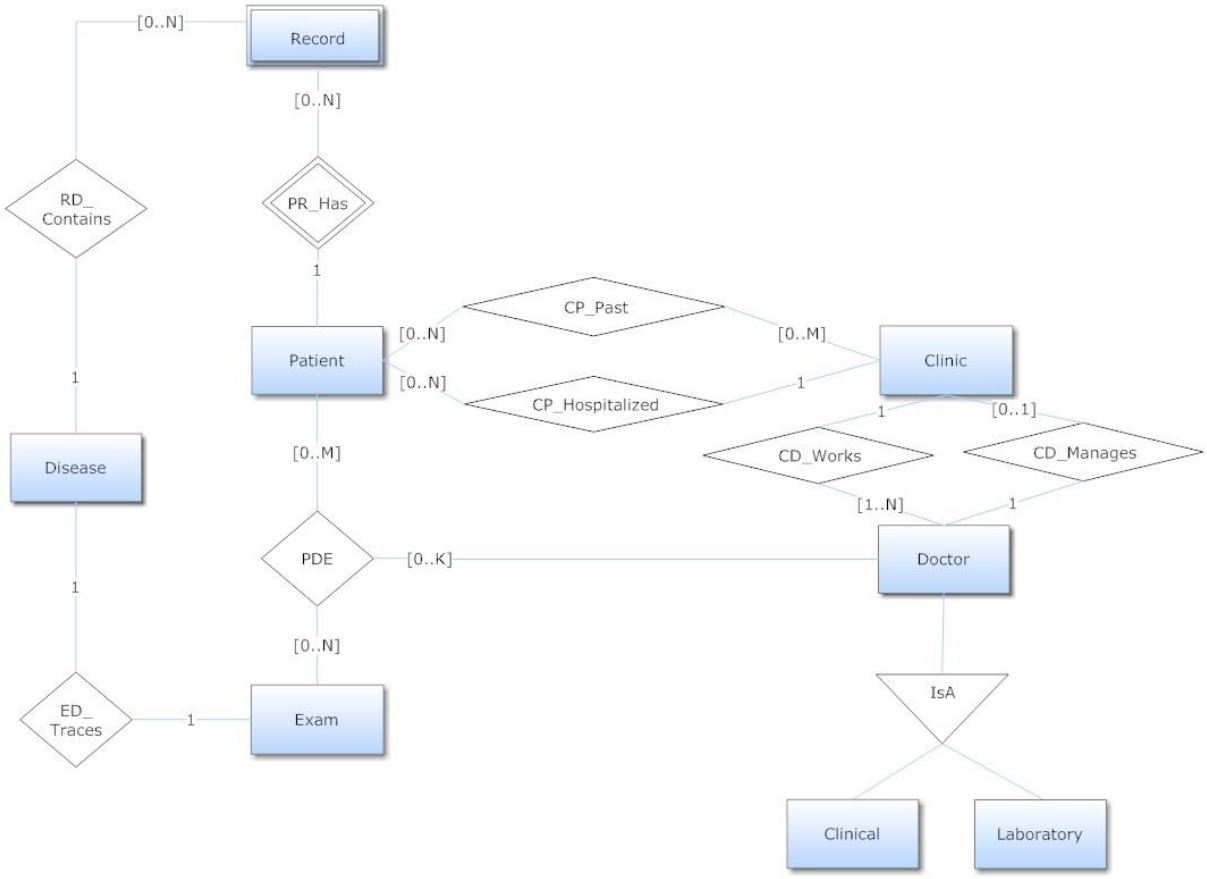
Η ενότητα αυτή παρουσιάζει το τελικό E-R μοντέλο του συστήματος. Στη συνέχεια γίνεται μία ανάλυση για το πώς αναπαραστάθηκαν και ικανοποιήθηκαν οι απαιτήσεις του συστήματος στο E-R μοντέλο ακολουθώντας την θεωρία και τους κανονισμούς που περιγράψαμε πιο πάνω.

Σχεδιάγραμμα E-R μοντέλου

Η Εικόνα 1 δείχνει το τελικό E-R μοντέλο του συστήματος όπως αυτό προέκυψε από την ανάλυση των απαιτήσεων και την μοντελοποίηση τους. Η Εικόνα 2 απεικονίζει το ίδιο E-R μοντέλο χωρίς τα γνωρίσματα για καλύτερη κατανόηση των σχέσεων μεταξύ των οντοτήτων.



Εικόνα 1: Μοντέλο E-R του συστήματος.



Εικόνα 2: Μοντέλο E-R του συστήματος (χωρίς γνωρίσματα).

Ανάλυση των απαιτήσεων του συστήματος.

Απαίτηση #1

«Για κάθε ασθενή θέλουμε να αποθηκεύομε το όνομα του, τον αριθμό ταυτότητας που είναι μοναδικός, την διεύθυνση του, την ημερομηνία γέννησης, το φύλο, την ασφάλεια που έχει. Επίσης θέλουμε να αποθηκεύουμε πληροφορία για το ιατρικό ιστορικό του ασθενή, που περιλαμβάνει την ασθένεια και την ημερομηνία διάγνωσης.»

Ανάλυση

Δημιουργήθηκε η οντότητα **Patient** με τα αντίστοιχα γνωρίσματα για κάθε απαιτούμενη πληροφορία που πρέπει να αποθηκεύεται. Ως αναγνωριστικό της οντότητας **Patient** ορίστηκε το γνώρισμα **patiendID** που αντιστοιχεί στον αριθμό ταυτότητας του ασθενή. Η διεύθυνση του ασθενή αντιμετωπίστηκε ως σύνθετο γνώρισμα, και διασπάστηκε στα γνωρίσματα:

1. Street

2. Postal Code
3. City

Για το ιατρικό του κάθε ασθενή δημιουργήθηκε η ασθενής οντότητα **Record** καθώς η ύπαρξη της εξαρτάται από την ύπαρξη του αντίστοιχου ασθενή. Επιλέχθηκε να αναπαρασταθεί ως ασθενής οντότητα καθώς έχει δικά της γνωρίσματα για την αποθήκευση της ημερομηνίας διάγνωσης και την ασθένεια. Ως μερικό αναγνωριστικό της οντότητας ορίστηκε ο συνδυασμός των 2 γνωρισμάτων, **dateHospitalized** και **Disease.**

Απαίτηση #2

«Για κάθε κλινική θέλομε να αποθηκεύομε πληροφορία για το όνομα της κλινικής, τον αριθμό κρεβατιών που υπάρχουν.»

Ανάλυση

Δημιουργήθηκε η οντότητα **Clinic** με τα αντίστοιχα γνωρίσματα για κάθε απαιτούμενη πληροφορία που πρέπει να αποθηκεύεται. Ως αναγνωριστικό της οντότητας **Clinic** ορίστηκε το γνώρισμα **idClinic,** το οποίο εισάχθηκε έτσι ώστε να αναγνωρίζει μοναδικά μία κλινική.

Απαίτηση #3

«Για κάθε γιατρό θέλομε να αποθηκεύομε το όνομα του, τον αριθμό ταυτότητας του, την ειδικότητα του, το τηλέφωνο του, την διεύθυνση του. Οι γιατροί χωρίζονται σε κλινικούς και εργαστηριακούς. Οι κλινικοί έχουν ως επιπλέον όρισμα τα έτη που έχουν εργαστεί μέσα σε νοσοκομειακούς χώρους ενώ οι εργαστηριακοί έχουν ως επιπλέον γνώρισμα αν ασχολούνται με ιογενής ή κληρονομικές ασθένειες.»

Ανάλυση

Δημιουργήθηκε η οντότητα **Doctor** με τα αντίστοιχα γνωρίσματα για κάθε απαιτούμενη πληροφορία που πρέπει να αποθηκεύεται. Ως αναγνωριστικό της οντότητας **Doctor** ορίστηκε το γνώρισμα **doctorID** που αντιστοιχεί στον αριθμό ταυτότητας του ασθενή. Όπως και στην οντότητα **Patient** το πεδίο **«Διεύθυνση»** αντιμετωπίστηκε ως σύνθετο γνώρισμα και διασπάστηκε. Η διάκριση μεταξύ εργαστηριακών και κλινικών ιατρών επιτυγχάνεται με την χρήση εξειδίκευσης (συσχέτιση **IsA**) όπως φαίνεται στην Εικόνα 1. Δημιουργήθηκαν δύο νέες υποκλάσεις, **Clinical** και **Laboratory** για την αναπαράσταση των

κλινικών και των εργαστηριακών ιατρών οι οποίες κληρονομούν όλα τα γνωρίσματα από την υπερκλάση **Doctor**. Σε κάθε μία από τις υποκλάσεις προστέθηκε το επιπλέον γνώρισμα που απαιτείται.

Απαίτηση #4

«Για κάθε εξέταση αποθηκεύομε τον κωδικό της εξέτασης και μια περιγραφή του είδους της, την ασθένεια που ανιχνεύει και το αντιδραστήριο το οποίο χρησιμοποιεί. Επίσης αποθηκεύομε την τιμή της εξέτασης. Το αντιδραστήριο καθορίζει μοναδικά την ασθένεια και την τιμή. Ο κωδικός καθορίζει μοναδικά το αντιδραστήριο. Η ασθένεια καθορίζει την περιγραφή.»

Ανάλυση

Σε πρώτη φάση δημιουργήθηκε η οντότητα **Exam** με όλα τα απαιτούμενα γνωρίσματα **Code, Price, Reagent, Disease, Description.** Ως αναγνωριστικό ορίστηκε το πεδίο **Code** (κωδικός εξέτασης).

Απαίτηση #5

«Κάθε γιατρός ανήκει σε μια και μόνο κλινική. Θέλομε να αποθηκεύομε την ημέρα που ένας γιατρός/νοσηλευτής άρχισε την εργασία στην κλινική. Κάθε κλινική έχει πολλούς γιατρούς (τουλάχιστον 1)»

Ανάλυση

Η **1-N** συσχέτιση **CD_Works** και η υποχρεωτική συμμετοχή των οντοτήτων **Clinic** και **Doctor** σε αυτή την συσχέτιση διασφαλίζει πως:

1. κάθε γιατρός εργάζεται σε μία μόνο κλινική, ενώ κάθε κλινική έχει πολλούς γιατρούς.
2. Κάθε κλινική πρέπει να έχει τουλάχιστον ένα γιατρό.

Επιπλέον, στην συσχέτιση **CD_Works** προστέθηκε το γνώρισμα **dateStart** για την αποθήκευση της ημερομηνίας στην οποία ο γιατρός ξεκίνησε την εργασία στην συγκεκριμένη κλινική.

Απαίτηση #6

«Κάθε κλινική έχει ένα και μόνο ένα γιατρό ως διευθυντή.»

Ανάλυση

Η **1-1** συσχέτιση **CD_Manages** μεταξύ των οντοτήτων **Clinic** και **Doctor** διασφαλίζει πως μια κλινική μπορεί να έχει ένα και μόνο ένα γιατρό ως διευθυντή. Η οντότητα **Doctor** έχει προαιρετική συμμετοχή στην συσχέτιση αυτή και επομένως, ενώ κάθε κλινική θα έχει οπωσδήποτε ένα διευθυντή δεν είναι απαραίτητο ο κάθε γιατρός να είναι διευθυντής σε μία κλινική.

Απαίτηση #7

«Οι ασθενείς νοσηλεύονται σε μια κλινική κάθε φορά. Θέλομε να αποθηκεύομε την έναρξη και λήξη της νοσηλείας. Ένας ασθενής μπορεί να έχει νοσηλευτεί σε πολλές κλινικές και κάθε κλινική έχει πολλούς ασθενείς.»

Ανάλυση

Η **1-N** συσχέτιση **CP_Hospitalized** μεταξύ των οντοτήτων **Clinic** και **Patient** διασφαλίζει πως κάθε ασθενής μπορεί να νοσηλεύεται σε μία και μόνο μία κλινική κάθε φορά, και πως κάθε κλινική μπορεί να νοσηλεύει πολλούς ασθενείς. Η οντότητες **Patient** και **Clinic** είναι προαιρετικές στην συσχέτιση αυτή αφού υπάρχει περίπτωση ένας ασθενής που είναι στο σύστημα να μην νοσηλεύεται σε καμιά κλινική κάποια δεδομένη στιγμή και αντίστοιχα, μία κλινική να μην νοσηλεύει κάποιο ασθενή. Στην συσχέτιση **CP_Hospitalized** προστέθηκε το γνώρισμα **dateHospitalized** για την αποθήκευση της έναρξης νοσηλείας του ασθενή.

Η απαίτηση πως ένας ασθενής μπορεί να έχει νοσηλευτεί σε πολλές κλινικές αναπαριστάται από την **N-M** συσχέτιση **CP_Past** μεταξύ των οντοτήτων **Clinic** και **Patient**. Γνωρίσματα σε αυτή την συσχέτιση είναι το **dateStart** και **dateEnd**, στα οποία αποθηκεύεται η ημερομηνία έναρξης και λήξης της κάθε νοσηλείας.

Απαίτηση #8

«Οι ασθενής μπορεί να κάνει μία ή περισσότερες εξετάσεις. Την εξέταση ενός ασθενή την διατάζει ένας γιατρός. Θέλομε να αποθηκεύομε τα αποτελέσματα τις κάθε εξέτασης. Η ίδια εξέταση μπορεί να γίνει σε πολλούς ασθενείς.»

Ανάλυση

Για τον προγραμματισμό μίας εξέτασης απαιτείται η συμμετοχή 3 διαφορετικών οντοτήτων, του **Patient**, του **Doctors** και του **Exam**. Με την χρήση της τριαδικής συσχέτισης **PDE** μπορούμε να αποθηκεύσουμε πληροφορία για το ποιος γιατρός διατάζει ποιά εξέταση και για ποιόν ασθενή. Τα αποτελέσματα της εξέτασης αποτελούν γνώρισμα στη συσχέτιση αυτή. Οι πληθυκότητες (**0-N**) είναι κοινές για όλες τις πλευρές και έτσι σημειώνονται μία μόνο φορά στο διάγραμμα οντοτήτων σχέσεων.

3.2 Κανόνες Μετατροπής στο σχεσιακό μοντέλο.

Ένα E-R μοντέλο μπορεί να μετατραπεί σε ένα σύνολο σχέσεων (σχεσιακό μοντέλο) σύμφωνα με τους πιο κάτω κανόνες:

1. Κάθε οντότητα σε ένα διάγραμμα E-R απεικονίζεται σε μια σχέση με το ίδιο όνομα
    a. Όλα τα μονότιμα γνωρίσματα της οντότητας γίνονται γνωρίσματα της αντίστοιχης σχέσης.
    b. Τα αναγνωριστικά γνωρίσματα της σχέσης σχηματίζουν ένα υποψήφιο κλειδί της σχέσης.
    c. Τα στιγμιότυπα της οντότητας γίνονται πλειάδες της σχέσης
2. Κάθε ασθενής οντότητα απεικονίζεται σε μια σχέση με το ίδιο όνομα
    a. Όλα τα γνωρίσματα της ασθενούς οντότητας γίνονται γνωρίσματα της αντίστοιχης σχέσης.
    b. Τα αναγνωριστικά γνωρίσματα της σχέσης σχηματίζουν ένα υποψήφιο κλειδί της σχέσης.
    c. Το αναγνωριστικό γνώρισμα της αντίστοιχης ισχυρής σχέσης μεταφέρεται σε αυτή την σχέση σαν γνώρισμα, και μαζί με το υποψήφιο κλειδί σχηματίζεται το πρωτεύον κλειδί της σχέσης.
3. Μια οντότητα **E** με αναγνωριστικό **p** και ένα γνώρισμα πολλαπλών τιμών **a** απεικονίζεται σε μια σχέση με όνομα το όνομα του γνωρίσματος πολλαπλών τιμών.
    a. Το σχήμα της σχέσης περιλαμβάνει τα γνωρίσματα **p** και **a** και οι πλειάδες της είναι ζεύγη των τιμών των **p** και **a**
    b. Το πρωτεύων κλειδί της σχέσης είναι το σύνολο που περιλαμβάνει τα γνωρίσματα **p** και **a**
4. Μια **N-N** σχέση **r** μεταξύ οντοτήτων **E** και **F** απεικονίζεται σε μια σχέση **R** το σχήμα της οποίας περιέχει όλα τα γνωρίσματα που ανήκουν στα πρωτεύοντα κλειδιά των σχέσεων που αντιστοιχούν στις οντότητες **E** και **F**. Ο συνδυασμός αυτός σχηματίζει το πρωτεύων

κλειδί της **R**. Επίσης, το σχήμα της **R** περιέχει όλα τα γνωρίσματα της σχέσης **r**. Οι πλειάδες της **R** αντιστοιχούν στα στιγμιότυπα της **r**.

5. Αν οι οντότητες **E** και **F** συμμετέχουν σε μια 1-1 σχέση r, τότε αν η συμμετοχή των οντοτήτων είναι προαιρετική, δημιουργούμε σχέσεις για τις **E** και **F** και προσθέτουμε στην μία από αυτές ένα γνώρισμα για το πρωτεύον κλειδί της άλλης. Αν η συμμετοχή τους είναι υποχρεωτική, τότε οι 2 σχέσεις μπορούν να συνδυαστούν σε μία.
6. Για μια **N-1** σχέση **r** μεταξύ των οντοτήτων **E** και **F** δεν δημιουργούμε νέα σχέση για την αναπαράσταση της. Αν max_cardinality(F,r) = 1, τότε η σχέση της **F** πρέπει να περιέχει γνωρίσματα που αντιστοιχούν στο πρωτεύον κλειδί της **E** ως ξένο κλειδί. Αν η **F** έχει υποχρεωτική συμμετοχή στην **r,** τότε το ξένο κλειδί δεν δέχεται κενές τιμές.
7. Για κάθε n-αδικό τύπο συσχέτισης **R**, με **n > 2**,
    a. Δημιουργούμε μία νέα σχέση **S,** που θα αναπαριστά τον **R**
    b. Περιλαμβάνουμε ως ξένα κλειδιά στην **S** τα πρωτεύοντα κλειδιά των σχέσεων που αναπαριστούν τους τύπους οντοτήτων που συμμετέχουν στην **R**.
    c. Προσθέτουμε στην νέα σχέση τυχόν απλά γνωρίσματα του n-αδικού τύπου συσχέτισης ως γνωρίσματα στην **S**

3.2.1 Εφαρμογή 1ου κανόνα (Ισχυρές οντότητες)

Εφαρμόζοντας τον 1ο κανόνα μετατροπής E-R μοντέλου σε σχεσιακό μοντέλο παίρνουμε τις πιο κάτω σχέσεις (πίνακες). Τα υποψήφια πρωτεύονται κλειδιά είναι υπογραμμισμένα:

- Patient

patientID	Name	Sex	dateBirth	Insurance	City	Street	PostalCode

- Clinic

idClinic	Name	bedsNo

- Exam

idExam	Description	Reagent	Price

- Doctor

doctorID	Name	Specialty	City	Street	PostalCode

Οι πιο κάτω σχέσεις Clinical και Laboratory προκύπτουν από την ISA συσχέτιση μεταξύ των οντοτήτων Clinical – Doctor και Laboratory – Doctor

→Clinical

doctorID	YearsExp

→Laboratory

doctorID	diseaseType

3.2.2 Εφαρμογή 2ου κανόνα (Ασθενείς οντότητες)

Από την ασθενή οντότητα **Record** προκύπτει η πιο κάτω σχέση.

- Record

patientID	Disease	dateDiagnosed

- Patient

patientID	Name	Sex	dateBirth	Insurance	City	Street	PostalCode

3.2.3 Εφαρμογή 3ου κανόνα (Πλειότιμα γνωρίσματα)

Δεν υπάρχουν multivalued (πλειότιμα) γνωρίσματα στο E-R μοντέλο

3.2.4 Εφαρμογή 4ου κανόνα (Συσχετίσεις M-N)

Από την συσχέτιση CP_Past

- Clinic

idClinic	Name	bedsNo

- Patient

idPatient	Name	Sex	dateBirth	Insurance	City	Street	PostalCode

- Νέος πίνακας: CP_Past

idPatient	idClinic	dateStart	dateEnd

Από την συσχέτιση PDE

- Doctor

idDoctor	Name	Specialty	City	Street	PostalCode

- Patient

idPatient	Name	Sex	dateBirth	Insurance	City	Street	PostalCode

- Exam

idExam	Description	Reagent	Price

- Νέος πίνακας: PDE

idPatient	idDoctor	idExam	Results

3.2.5 Εφαρμογή 5ου κανόνα (Συσχετίσεις 1-1)

Από την συσχέτιση CD_Manages

- Doctor:

idDoctor	Name	Speciality	City	Street	PostalCode

- Clinic: νέο γνώρισμα idManager, dateStart

idClinic	Name	#Beds	idManager	dateStart

3.2.6 Εφαρμογή 6ου κανόνα (Συσχετίσεις 1-N)

Από την συσχέτιση CD_Works

- Clinic

idClinic	Name	#Beds	idManager	dateStart

- Doctor: νέο γνώρισμα idClinic, dateEmployment

idDoctor	Name	Specialty	City	Street	PostalCode	idClinic	dateEmployment

Από την συσχέτιση CP_Hospitalized

- Clinic

idClinic	Name	#Beds	idManager	dateStart

- Patient: νέα γνωρίσματα: dateHS (=dateHospitalizationStart), idClinic

idPatient	Name	Sex	dateBirth	Insurance	City	Street	PostalCode	dateHS	idClinic

Από την συσχέτιση Contains

- Disease

idDisease	Name

- Record: νέο γνώρισμα idDisease

idRecord	idPatient	dateDiagnosed	idDisease

3.2.7 Συνολικά

Patient

idPatient	Name	Sex	dateBirth	Insurance	City	Street	PostalCode	dateHS	idClinic

Record

idRecord	idPatient	idDisease	dateDiagnosed

Clinic

idClinic	name	#Beds	idManager	dateStart

Doctor

idDoctor	Name	Specialty	City	Street	PostalCode	idClinic	dateEmployment

Phones

idDoctor	Phone

Clinical

idDoctor	YearsExp

Laboratory

idDoctor	diseaseType

Disease

idDisease	Name

Exam

idExam	Description	Reagent	Price

PDE

idPatient	idDoctor	idExam	Results

CP_Past

idPatient	idClinic	dateStart	dateEnd

3.3 Περιορισμοί Ακεραιότητας

Αναφέρονται σύντομα οι περιορισμοί ακεραιότητας που μπορεί να ισχύουν σε κάποιες από τις σχέσεις. Η εφαρμογή τους γίνεται στην επόμενη ενότητα.

- **Περιορισμός Πεδίου Ορισμού:** Η τιμή κάθε γνωρίσματος A πρέπει να είναι μία ατομική τιμή από το πεδίο ορισμού αυτού του γνωρίσματος dom(A)
- **Περιορισμός Κλειδιού**: όλες οι πλειάδες σε μία σχέση πρέπει να είναι διαφορετικές
- **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
- **Περιορισμός Αναφορικής Ακεραιότητας**: Όταν μια πλειάδα μιας σχέσης s, αναφέρεται σε μια άλλη , τότε αυτή η άλλη πρέπει να υπάρχει.
- **Περιορισμός Σημασιολογικής Ακεραιότητας** : Λογικοί περιορισμοί που ισχύουν στον πραγματικό κόσμο.

3.4 Συναρτησιακές Εξαρτήσεις

Patients (idPatient, Sex, Name, DateBirth, Insurance, City, Street, PostalCode, idClinic dateHS)

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Πεδίου Ορισμού:** το κάθε γνώρισμα Sex μπορεί να λάβει τιμές από το πεδίο ορισμού dom(Sex) = {Male, Female}.

2. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
3. **Περιορισμός Αναφορικής Ακεραιότητας**: Το πεδίο idClinic αναφέρεται στην σχέση Clinic και πρέπει να υπάρχει η συγκεκριμένη πλειάδα.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idPatient→ Name, DateBirth, Insurance, City, Street, PostalCode, idClinic, dateHS **(περιορισμός πρωτεύοντος κλειδιού)**

καθώς με μία συγκεκριμένη τιμή idPatient μπορώ να βρω μόνο έναν ασθενή με όνομα Name, ασφάλεια Insurance και λοιπά. Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "Patients".

Πρωτεύον κλειδί: Το πεδίο idPatient αποτελεί το **πρωτεύον κλειδί** της σχέσης.

Doctors {idDoctor, Name, Specialty, City, Street, PostalCode, idClinic, dateStart}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Το πεδίο idClinic αναφέρεται στην σχέση Clinic και πρέπει να υπάρχει η συγκεκριμένη πλειάδα.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idDoctor → Name, Specialty, City, Street, PostalCode, idClinic, dateStart **(περιορισμός πρωτεύοντος κλειδιού)**

καθώς με μία συγκεκριμένη τιμή idDoctor μπορώ να βρώ μόνο έναν γιατρό με όνομα Name, ειδικότητα Specialty και λοιπά. Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "Doctors".

Πρωτεύον κλειδί: Το πεδίο idDoctor αποτελεί το **πρωτεύον κλειδί** της σχέσης.

Clinics {idClinic, Name, #Beds, idManager, dStart}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Το πεδίο idManager αναφέρεται στην σχέση Doctor και πρέπει να υπάρχει η συγκεκριμένη πλειάδα.
3. **Περιορισμός Σημασιολογικής Ακεραιότητας** : Ο αριθμός των κρεβατιών σε μία κλινική δεν μπορεί να είναι αρνητικός αριθμός.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idClinic → Name, #Beds, idManager, dStart **(περιορισμός πρωτεύοντος κλειδιού)**

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "Clinics".

Πρωτεύον κλειδί: Το πεδίο idClinic αποτελεί το **πρωτεύον κλειδί** της σχέσης.

Exams {idExam, Reagent, Price, Description, idDisease}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Το πεδίο idDisease αναφέρεται στην σχέση Disease και πρέπει να υπάρχει η συγκεκριμένη πλειάδα.
3. **Περιορισμός Σημασιολογικής Ακεραιότητας** : Η τιμή της εξέτασης δεν μπορεί να είναι αρνητικός αριθμός.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idExam→Reagent, Price, Description, idDisease **(περιορισμός πρωτεύοντος κλειδιού)**

Reagent→idDisease, Price (καθώς το αντιδραστήριο καθορίζει μοναδικά την ασθένεια και την τιμή)

idExam→Reagent (καθώς η εξέταση καθορίζει μοναδικά το αντιδραστήριο)

idDisease→Description (καθώς η ασθένεια καθορίζει μοναδικά την περιγραφή)

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "Exams".

Πρωτεύον κλειδί: Το πεδίο idExam αποτελεί το **πρωτεύον κλειδί** της σχέσης.

Diseases {idDisease, Name}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idDisease → Name **(περιορισμός πρωτεύοντος κλειδιού)**

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "Clinics".

Πρωτεύον κλειδί: Το πεδίο idDisease αποτελεί το **πρωτεύον κλειδί** της σχέσης.

CP_Past {idPatient, idClinic, dateStart, dateEnd}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Τα πεδία idPatient και idClinic αναφέρονται στις σχέσεις Patients και Clinics και πρέπει να υπάρχουν οι συγκεκριμένες πλειάδες.

Συναρτησιακές Εξαρτήσεις:

idPatient, dateStart → idClinic , dateEnd **(περιορισμός πρωτεύοντος κλειδιού)**

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "CP_Past".

Πρωτεύον κλειδί: Τα πεδία idPatient, dateStart αποτελούν το **πρωτεύον κλειδί** της σχέσης.

PDE {idPatient, idDoctor, idExam, Result}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δεν μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Τα πεδία idPatient, idDoctor, idExam αναφέρονται στις σχέσεις Patients, Doctors και Exams αντίστοιχα και πρέπει να υπάρχουν οι συγκεκριμένες πλειάδες.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idPatient, idDoctor, idExam→ Result **(περιορισμός πρωτεύοντος κλειδιού)**

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "CP_Past".

Πρωτεύον κλειδί: Τα πεδία {idPatient, idDoctor, idExam} αποτελούν το **πρωτεύον κλειδί** της σχέσης.

Records {idRecord, idPatient, idDisease, dateDiagnosis}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Τα πεδία idPatient, idDisease αναφέρονται στις σχέσεις Patients και Diseases αντίστοιχα και πρέπει να υπάρχουν οι συγκεκριμένες πλειάδες.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idRecord, idPatient → idDisease, dateDiagnosis **(περιορισμός πρωτεύοντος κλειδιού)**

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης "CP_Past".

Πρωτεύον κλειδί: Τα πεδία {idRecord, idPatient} αποτελούν το **πρωτεύον κλειδί** της σχέσης.

ClinicalDoctor {idDoctor, yearsExp}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Το πεδίο idDoctor αναφέρεται στη σχέση Doctor και πρέπει να υπάρχει η συγκεκριμένη πλειάδα.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idDoctor → yearsExp **(περιορισμός πρωτεύοντος κλειδιού)**

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης “ClinicalDoctor ”.

Πρωτεύον κλειδί: Το πεδίο {idDoctor} αποτελεί το **πρωτεύον κλειδί** της σχέσης.

LaboratoryDoctor {idDoctor, diseaseType}

Περιορισμοί Ακεραιότητας:

1. **Περιορισμός Ακεραιότητας Οντοτήτων**: Δε μπορεί η τιμή του πρωτεύοντος κλειδιού να είναι null
2. **Περιορισμός Αναφορικής Ακεραιότητας**: Το πεδίο idDoctor αναφέρεται στη σχέση Doctor και πρέπει να υπάρχει η συγκεκριμένη πλειάδα.

Συναρτησιακές Εξαρτήσεις: Σε αυτή την σχέση ισχύει ότι:

idDoctor → diseaseType **(περιορισμός πρωτεύοντος κλειδιού)**

Δεν υπάρχουν άλλες συναρτησιακές εξαρτήσεις καθώς κανένα από τα υπόλοιπα πεδία δεν καθορίζει μοναδικά κάποιο υποσύνολο της σχέσης “LaboratoryDoctor ”.

Πρωτεύον κλειδί: Το πεδίο {idDoctor} αποτελεί το **πρωτεύον κλειδί** της σχέσης.

3.5 Κανονικές Μορφές

3.5.1 Πρώτη Κανονική Μορφή (1NF)

Ορισμός: Μια σχέση R είναι σε πρώτη κανονική μορφή αν δεν περιέχει κανένα πλειότιμο και κανένα σύνθετο γνώρισμα.

Εφαρμογή: Η εφαρμογή του καταλήγει στην διάσπαση όλων των σύνθετων πεδίων στα πεδία που τα αποτελούν (παράδειγμα το πεδίο Address διασπάται στα πεδία Street, City και PostalCode) αλλά και την δημιουργία νέων πινάκων για όλα τα γνωρίσματα πολλαπλών τιμών. Οι σχέσεις που προέκυψαν από την μετατροπή του E-R μοντέλου σε σχεσιακό μοντέλο είναι αυτομάτως σε $1^{η}$ κανονική μορφή.

3.5.2 Δεύτερη Κανονική Μορφή (2NF)

Ορισμός: Μία σχέση R είναι σε δεύτερη κανονική μορφή αν **1)** είναι σε πρώτη κανονική μορφή και **2)** αν κάθε συναρτησιακή εξάρτηση X→Y που υπάρχει στην R, είναι full functional dependency. Μια συναρτησιακή εξάρτηση X→Y είναι full functional dependency αν η συναρτησιακή εξάρτηση παύει να ισχύει αν αφαιρέσουμε οποιοδήποτε πεδίο από το X.

Εφαρμογή: Από τις συναρτησιακές εξαρτήσεις που αναφέρθηκαν νωρίτερα οι παρακάτω έχουν παραπάνω από ένα πεδία στο X και εξηγείται γιατί είναι full functional dependency:

- idPatient, dateStart → idClinic , dateEnd
  1) idPatient → idClinic, dateEnd: Δεν ισχύει γιατί ο ίδιος ασθενής μπορεί να έχει νοσηλευτεί στην ίδια κλινική πολλές φορές
  2) dateStart → idClinic, dateEnd: Δεν ισχύει γιατί πολλοί ασθενείς μπορεί να εισαχθούν στην ίδια κλινική τις ίδιες ημερομηνίες.
- idPatient, idDoctor, idExam→ Result
  1) idPatient, idDoctor → Result: Δεν ισχύει αφού ένας γιατρός μπορεί να δώσει στον ίδιο ασθενή περισσότερες από μία εξετάσεις.
  2) idPatient, idExam → Result: Δεν ισχύει αφού ένας ασθενής μπορεί να κάνει την ίδια εξέταση μετά από παραγγελία από διαφορετικούς ιατρούς.
  3) idDoctor, idExam→ Result: Δεν ισχύει αφού ένας γιατρός μπορεί να δώσει την ίδια εξέταση σε πολλούς ασθενείς.

- idRecord, idPatient → idDisease, dateDiagnosis
  1) idPatient → idDisease, dateDiagnosis: Δεν ισχύει αφού ένας ασθενής μπορεί να διαγνωσθεί με μία ασθένεια περισσότερες από μία φορές.
  2) idRecord → idDisease, dateDiagnosis: Δεν ισχύει αφού πολλοί ασθενείς μπορεί να έχουν το ίδιο αναγνωριστικό διάγνωσης.

3.5.3 Τρίτη Κανονική Μορφή (3NF)

Ορισμός: Μια σχέση R είναι σε Τρίτη κανονική μορφή αν **1)** είναι σε δεύτερη κανονική και **2)** αν δεν υπάρχουν μεταβατικές εξαρτήσεις. Μια συναρτησιακή εξάρτηση X→Y είναι μεταβατική αν ισχύει X→Z και Z→Y για κάποιο σύνολο από πεδία.

Εφαρμογή: Για την σχέση Exam ισχύει ότι:

idExam→idDisease (X→Y)

idExam→Reagent (X→Z)

Reagent→idDisease (Z→Y)

Εάν θελήσουμε να διαγράψουμε όλες τις ασθένειες, θα διαγραφούν και όλα τα αντιδραστήρια που χρησιμοποιούνται.

Άρα ο πίνακας Exam θα πρέπει να διασπαστεί σε δύο πίνακες ως εξής:

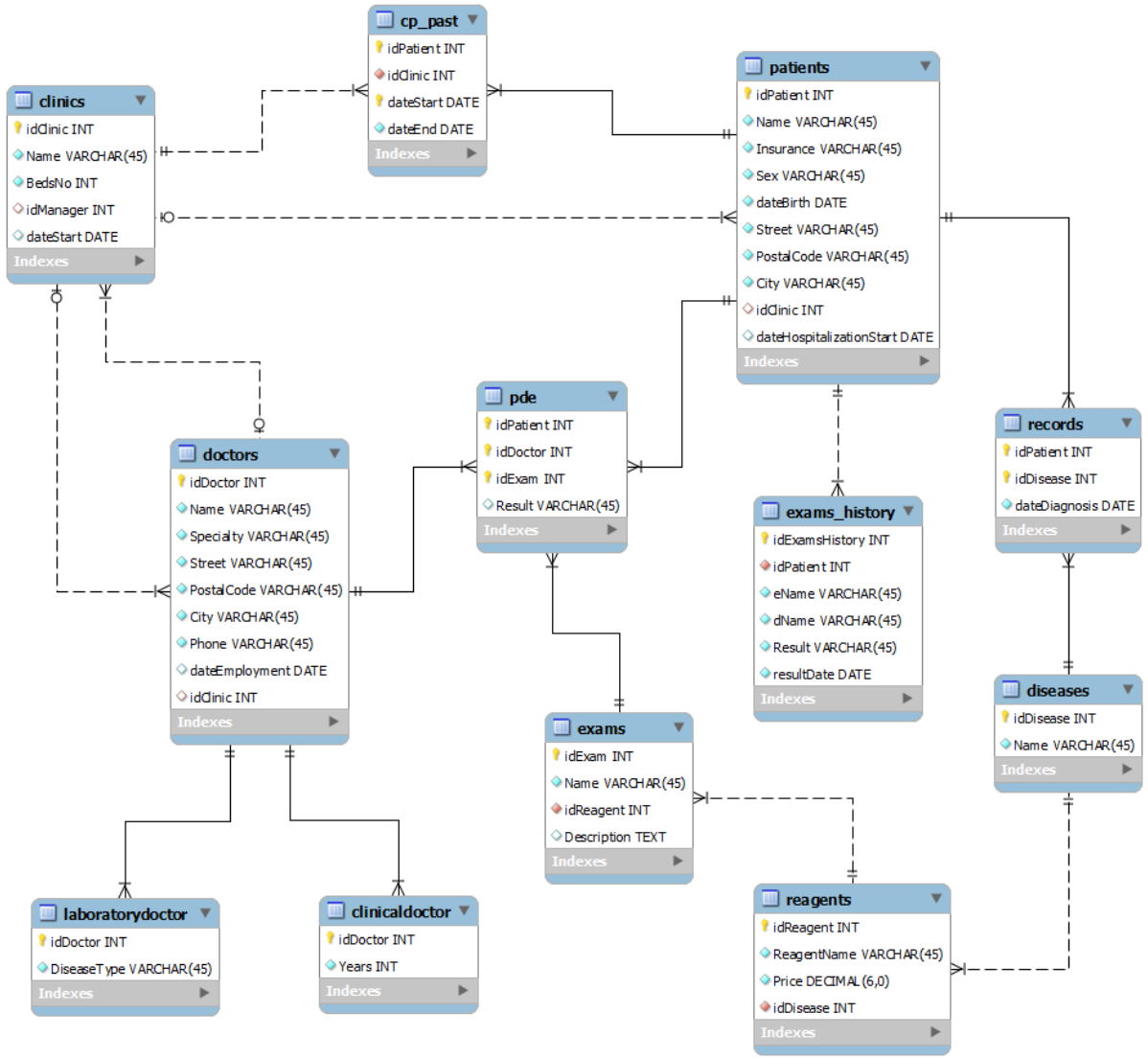
1. Πίνακας Exam

idExam	Description	idReagent

2. Πίνακας Reagent

idReagent	ReagentName	idDisease	Price

3.6 Διάγραμμα Βάσης Δεδομένων



3.7 Κώδικας SQL για την δημιουργία της βάσης δεδομένων

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';

DROP SCHEMA IF EXISTS `hospital` ;
CREATE SCHEMA IF NOT EXISTS `hospital` DEFAULT CHARACTER SET
latin1 ;
USE `hospital` ;

-- -----------------------------------------------------
-- Table `hospital`.`clinics`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`clinics` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`clinics` (
`idClinic` INT NOT NULL AUTO_INCREMENT ,
`Name` VARCHAR(45) NOT NULL ,
`BedsNo` INT UNSIGNED NOT NULL ,
`idManager` INT NULL DEFAULT NULL ,
`dateStart` DATE NULL DEFAULT NULL ,
PRIMARY KEY (`idClinic`) ,
UNIQUE INDEX `Name` (`Name` ASC) ,
INDEX `FK_Clinics_Doctors_1` (`idManager` ASC) ,
CONSTRAINT `FK_Clinics_Doctors_1`
FOREIGN KEY (`idManager` )
REFERENCES `hospital`.`doctors` (`idDoctor` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
AUTO_INCREMENT = 0
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`doctors`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`doctors` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`doctors` (
`idDoctor` INT NOT NULL ,
`Name` VARCHAR(45) NOT NULL ,
```

```
`Specialty` VARCHAR(45) NOT NULL ,
`Street` VARCHAR(45) NOT NULL ,
`PostalCode` VARCHAR(45) NOT NULL ,
`City` VARCHAR(45) NOT NULL ,
`Phone` VARCHAR(45) NOT NULL ,
`dateEmployment` DATE NULL DEFAULT NULL ,
`idClinic` INT NULL DEFAULT NULL ,
PRIMARY KEY (`idDoctor`) ,
UNIQUE INDEX `Phone` (`Phone` ASC) ,
INDEX `FK_Doctors_Clinics_1` (`idClinic` ASC) ,
UNIQUE INDEX `doctorID_UNIQUE` (`idDoctor` ASC) ,
CONSTRAINT `FK_Doctors_Clinics_1`
FOREIGN KEY (`idClinic` )
REFERENCES `hospital`.`clinics` (`idClinic` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
AUTO_INCREMENT = 0
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`clinicaldoctor`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`clinicaldoctor` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`clinicaldoctor` (
`idDoctor` INT NOT NULL ,
`Years` INT NOT NULL ,
PRIMARY KEY (`idDoctor`) ,
INDEX `FK_Clinical_Doctors_1` (`idDoctor` ASC) ,
CONSTRAINT `FK_Clinical_Doctors_1`
FOREIGN KEY (`idDoctor` )
REFERENCES `hospital`.`doctors` (`idDoctor` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`patients`
-- -----------------------------------------------------
```

```
DROP TABLE IF EXISTS `hospital`.`patients` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`patients` (
`idPatient` INT NOT NULL ,
`Name` VARCHAR(45) NOT NULL ,
`Insurance` VARCHAR(45) NOT NULL ,
`Sex` VARCHAR(45) NOT NULL ,
`dateBirth` DATE NOT NULL ,
`Street` VARCHAR(45) NOT NULL ,
`PostalCode` VARCHAR(45) NOT NULL ,
`City` VARCHAR(45) NOT NULL ,
`idClinic` INT NULL DEFAULT NULL ,
`dateHospitalizationStart` DATE NULL DEFAULT NULL ,
PRIMARY KEY (`idPatient`) ,
INDEX `FK_Patients_Clinics_1` (`idClinic` ASC) ,
UNIQUE INDEX `idPatient_UNIQUE` (`idPatient` ASC) ,
CONSTRAINT `FK_Patients_Clinics_1`
FOREIGN KEY (`idClinic` )
REFERENCES `hospital`.`clinics` (`idClinic` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
AUTO_INCREMENT = 0
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`cp_past`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`cp_past` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`cp_past` (
`idPatient` INT NOT NULL ,
`idClinic` INT NOT NULL ,
`dateStart` DATE NOT NULL ,
`dateEnd` DATE NOT NULL ,
PRIMARY KEY (`idPatient`, `dateStart`) ,
INDEX `FK_Past_Clinics_1` (`idClinic` ASC) ,
INDEX `FK_Past_Patient_2` (`idPatient` ASC) ,
CONSTRAINT `FK_Past_Clinics_1`
FOREIGN KEY (`idClinic` )
REFERENCES `hospital`.`clinics` (`idClinic` )
ON DELETE NO ACTION
```

```
ON UPDATE NO ACTION,
CONSTRAINT `FK_Past_Patient_2`
FOREIGN KEY (`idPatient` )
REFERENCES `hospital`.`patients` (`idPatient` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`diseases`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`diseases` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`diseases` (
`idDisease` INT NOT NULL AUTO_INCREMENT ,
`Name` VARCHAR(45) NOT NULL ,
PRIMARY KEY (`idDisease`) )
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`reagents`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`reagents` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`reagents` (
`idReagent` INT NOT NULL AUTO_INCREMENT ,
`ReagentName` VARCHAR(45) NOT NULL ,
`Price` DECIMAL(6,0) NOT NULL ,
`idDisease` INT NOT NULL ,
PRIMARY KEY (`idReagent`) ,
UNIQUE INDEX `ReagentName` (`ReagentName` ASC) ,
INDEX `FK_Diseases_Reagents_1` (`idDisease` ASC) ,
CONSTRAINT `FK_Diseases_Reagents_1`
FOREIGN KEY (`idDisease` )
REFERENCES `hospital`.`diseases` (`idDisease` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;
```

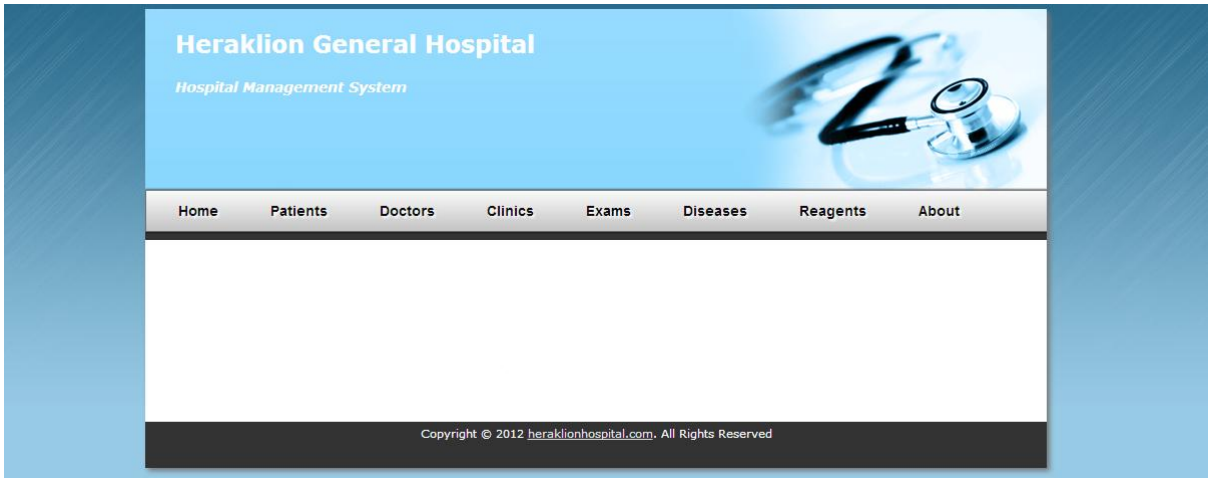
```
-- -----------------------------------------------------
-- Table `hospital`.`exams`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`exams` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`exams` (
`idExam` INT NOT NULL AUTO_INCREMENT ,
`Name` VARCHAR(45) NOT NULL ,
`idReagent` INT NOT NULL ,
`Description` TEXT NULL ,
PRIMARY KEY (`idExam`) ,
UNIQUE INDEX `Name` (`Name` ASC) ,
INDEX `FK_Exams_Reagents_1` (`idReagent` ASC) ,
CONSTRAINT `FK_Exams_Reagents_1`
FOREIGN KEY (`idReagent` )
REFERENCES `hospital`.`reagents` (`idReagent` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`exams_history`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`exams_history` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`exams_history` (
`idExamsHistory` INT NOT NULL AUTO_INCREMENT ,
`idPatient` INT NOT NULL ,
`eName` VARCHAR(45) NOT NULL ,
`dName` VARCHAR(45) NOT NULL ,
`Result` VARCHAR(45) NOT NULL ,
`resultDate` DATE NOT NULL ,
PRIMARY KEY (`idExamsHistory`) ,
INDEX `FK_Patient_ExamsH1` (`idPatient` ASC) ,
CONSTRAINT `FK_Patient_ExamsH1`
FOREIGN KEY (`idPatient` )
REFERENCES `hospital`.`patients` (`idPatient` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
```

```
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`laboratorydoctor`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`laboratorydoctor` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`laboratorydoctor` (
`idDoctor` INT NOT NULL ,
`DiseaseType` VARCHAR(45) NOT NULL ,
PRIMARY KEY (`idDoctor`) ,
INDEX `FK_Lab_Doctors_1` (`idDoctor` ASC) ,
CONSTRAINT `FK_Lab_Doctors_1`
FOREIGN KEY (`idDoctor` )
REFERENCES `hospital`.`doctors` (`idDoctor` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`pde`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`pde` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`pde` (
`idPatient` INT NOT NULL ,
`idDoctor` INT NOT NULL ,
`idExam` INT NOT NULL ,
`Result` VARCHAR(45) NULL DEFAULT NULL ,
PRIMARY KEY (`idPatient`, `idDoctor`, `idExam`) ,
INDEX `FK_PDE_Patients_1` (`idPatient` ASC) ,
INDEX `FK_PDE_Doctors_2` (`idDoctor` ASC) ,
INDEX `FK_PDE_Exams_3` (`idExam` ASC) ,
CONSTRAINT `FK_PDE_Doctors_2`
FOREIGN KEY (`idDoctor` )
REFERENCES `hospital`.`doctors` (`idDoctor` )
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `FK_PDE_Exams_3`
```

```
FOREIGN KEY (`idExam` )
REFERENCES `hospital`.`exams` (`idExam` )
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `FK_PDE_Patients_1`
FOREIGN KEY (`idPatient` )
REFERENCES `hospital`.`patients` (`idPatient` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;


-- -----------------------------------------------------
-- Table `hospital`.`records`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `hospital`.`records` ;

CREATE  TABLE IF NOT EXISTS `hospital`.`records` (
`idPatient` INT NOT NULL ,
`idDisease` INT NOT NULL ,
`dateDiagnosis` DATE NOT NULL ,
PRIMARY KEY (`idPatient`, `idDisease`) ,
INDEX `FK_Records_Patients_1` (`idPatient` ASC) ,
INDEX `FK_Records_Diseases_2` (`idDisease` ASC) ,
CONSTRAINT `FK_Records_Diseases_2`
FOREIGN KEY (`idDisease` )
REFERENCES `hospital`.`diseases` (`idDisease` )
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `FK_Records_Patients_1`
FOREIGN KEY (`idPatient` )
REFERENCES `hospital`.`patients` (`idPatient` )
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;



SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Κεφάλαιο 4: Εγχειρίδιο χρήσης και υλοποίηση

4.1 Διεπαφή της εφαρμογής

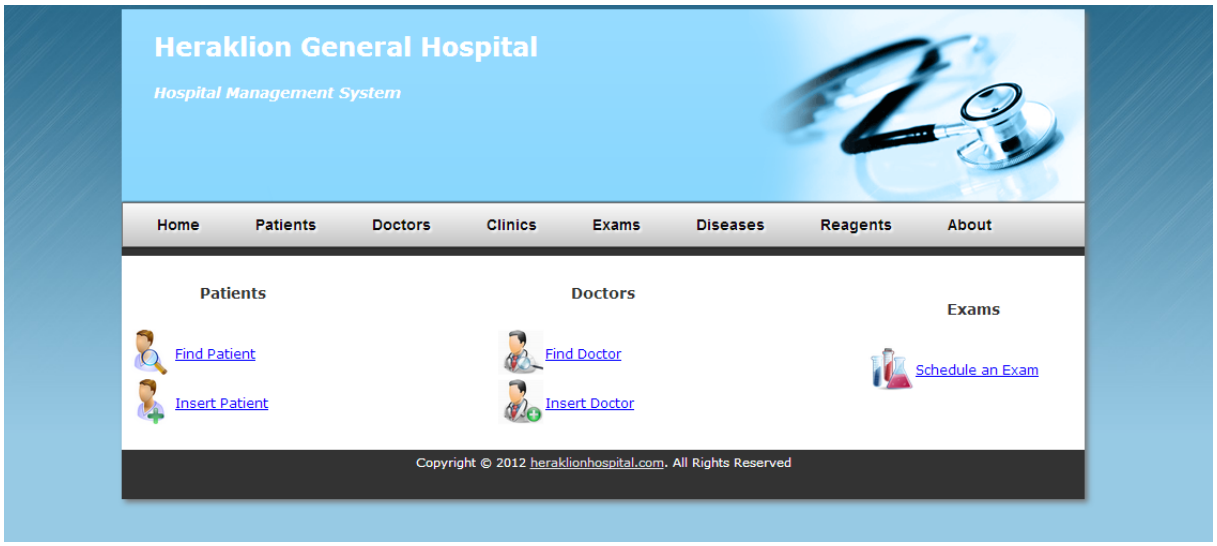


Εικόνα 3: Η διεπαφή χρήστη του συστήματος.

Η Εικόνα 3 δείχνει την διεπαφή χρήστη του συστήματος. Ο σχεδιασμός και η υλοποίηση της διεπαφής χρήστη έγινε με την δημιουργία ένος template στο Dreamweaver, βάση του οποίου δημιουργήθηκαν όλες οι σελίδες του συστήματος. Η χρήση template επιτρέπει την κατασκευή ενός site το οποίο θα είναι ευέλικτο και που θα μπορεί εύκολα να ενημερωθεί, εξασφαλίζοντας μια συνέπεια μεταξύ των σελίδων HTML. Επίσης τα templates ενισχύουν την παραγωγικότητα αφού για την αλλαγή της διάταξης ενός site, χρειάζεται να αλλάξει μόνο ένα αρχείο, το template . Όλες οι σελίδες που δημιουργήθηκαν βάση αυτού του template ενημερώνονται αυτόματα επιτρέποντας τροποποιήσεις σε όλο το site μέσα σε λίγα λεπτά.

Μέσα στο template που δημιουργήθηκε αποκλειστικά για αυτή την εφαρμογή, καθορίστηκαν κάποια σημεία τα οποία μπορούν να τροποποιηθούν μέσα από τις σελίδες που συνδέονται με το template αυτό. Τα σημεία αυτά φαίνονται πιο κάτω.

4.2 Αρχική σελίδα



Εικόνα 4: Η αρχική σελίδα του συστήματος.

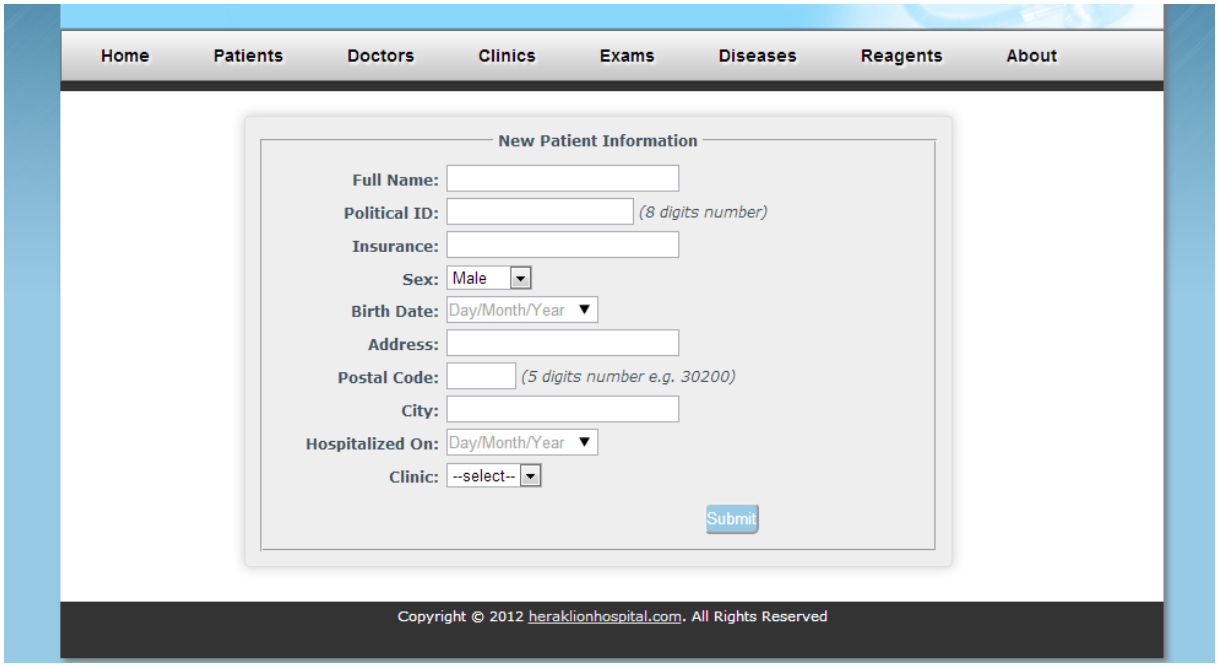
Η Εικόνα 4 δείχνει την αρχική σελίδα του συστήματος. Αυτή η σελίδα εμφανίζεται όταν από οποιοδήποτε μέρος στην εφαρμογή, επιλέξουμε τον σύνδεσμο **Home** από το top bar menu. Όπως φαίνεται στην αρχική σελίδα και στον χώρο content υπάρχουν κάποιες συντομεύσεις σε βασικές λειτουργίες του συστήματος όπως **Find Patient**, **Insert Patient**, **Find Doctor, Instert Doctor, Schedule an Exam** με σκοπό την γρήγορη πλοήγηση στις σελίδες όπου υλοποιούνται οι διαδικασίες αυτές. Αξίζει να σημειωθεί πώς σύνδεσμοι στις λειτουργίες αυτές υπάρχουν και στο αποκλειστικό menu της κάθε οντότητας.

4.3 Ασθενείς

Εικόνα 5: Patient menu.

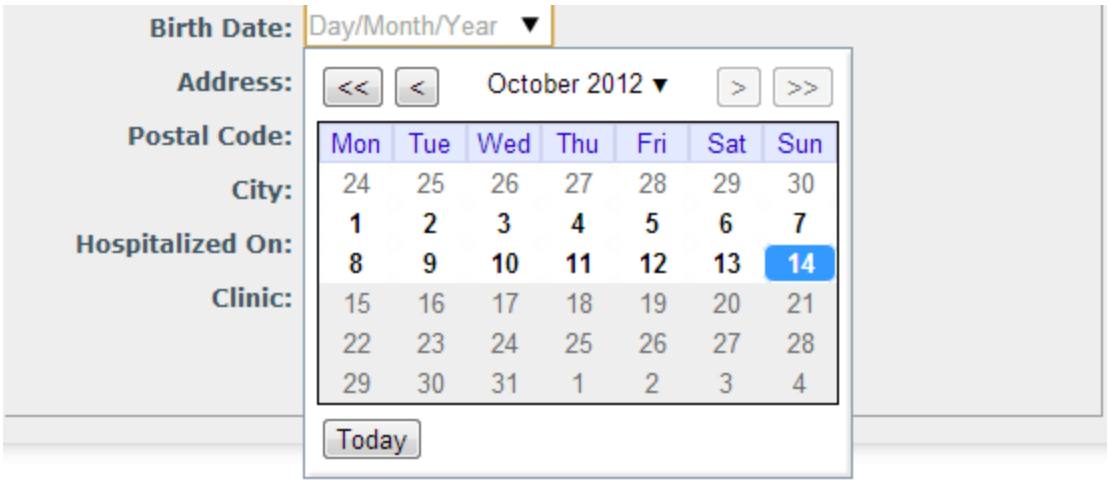
Η Εικόνα 5 δείχνει όλες τις διαδικασίες που μπορούμε να εκτελέσουμε από το μενού του **Patients**. Θα ακολουθήσει λεπτομερής περιγραφή των διαδικασιών αυτών στα πιο κάτω υποκεφάλαια.

4.3.1 Δημιουργία Ασθενή (Patients->Create new)



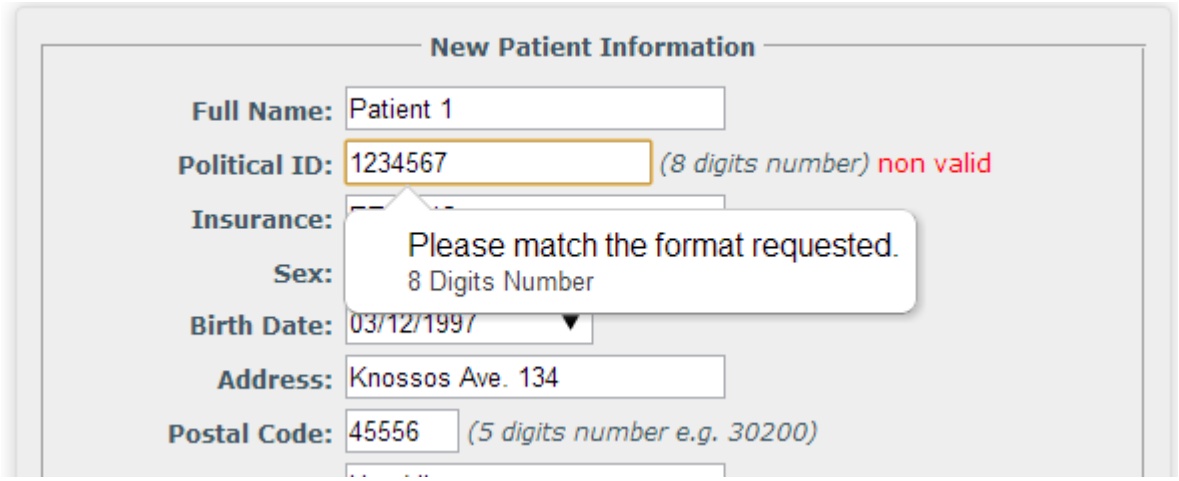
Εικόνα 6: Φόρμα για την δημιουργία ασθενή.

Η Εικόνα 6 δείχνει την φόρμα για την εισαγωγή ενός ασθενή στο σύστημα. Όπως βλέπετε σε κάποια πεδία όπως το **Political ID** και **Postal Code** υπάρχουν επεξηγηματικές σημειώσεις για την μορφή και τον τύπο των τιμών που δέχονται. Στην φόρμα αυτή αλλά και σε όλα τα πεδία του συστήματος όπου απαιτείται ημερομηνία γίνεται χρήση ενός νέου input type το οποίο προσφέρεται από την HTML5 και ονομάζεται **date.** Με αυτό τον τρόπο επιτυγχάνουμε την εμφάνιση ημερολογίου (Εικόνα 7) όταν ο χρήστης προσπαθήσει να εισάγει κάποια ημερομηνία χωρίς να απαιτείται η χρήση JavaScript. Επίσης αυτό βοηθάει τον χρήστη στον να εισάγει την ημερομηνία στο σωστό format το οποίο είναι αποδεκτό από το σύστημα.



Εικόνα 7: HTML5 date input type.

Επιπλέον, το validation των τιμών που εισάγει ο χρήστης επιτυγχάνεται με την χρήση patterns το οποίο είναι επίσης ένα νέο feature που προσφέρεται από την HTML5. Πχ για το πεδίο **Political ID** απαιτείται ένας 8-ψήφιος αριθμός. Σε περίπτωση εισαγωγής μη επιτρεπτής τιμής, το πιο κάτω μήνυμα θα εμφανιστεί όταν προσπαθήσουμε να κάνουμε submit.

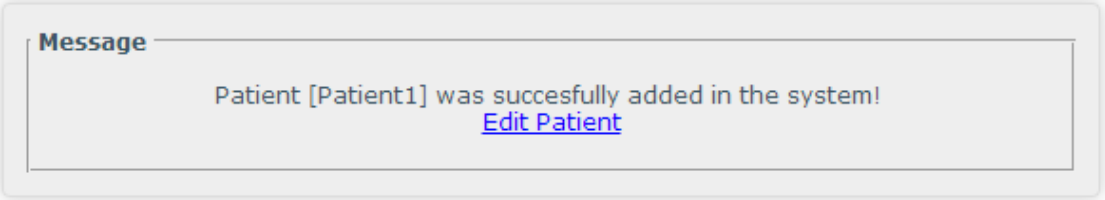


Εικόνα 8: Επικύρωση εισόδου με την χρήση pattern.

Επίσης στην Εικόνα 8 είναι ορατός και ο μηχανισμός για real-time validation του πεδίου **Political ID** ο οποίος δουλεύει ως εξής:

- Καθώς ο χρήστης γράφει σε αυτό το πεδίο, με κάθε νέο χαρακτήρα η τιμή στέλνεται μέσω AJAX σε μια μέθοδο PHP, η οποία ελέγχει κατά πόσο η συγκεκριμένη είσοδος είναι έγκυρη (valid) ή άκυρη (non-valid). Με αυτό τον τρόπο οι χρήστες ενημερώνονται κατά πόσο ο αριθμός ταυτότητας που δώσανε είναι στην σωστή μορφή και δεν έχει καταχωρηθεί σε κάποιο άλλο ασθενή.

Όταν ο χρήστης συμπληρώσει σωστά όλα τα απαιτούμενα πεδία και πατήσει το κουμπί **Submit**, τότε το σύστημα θα προσπαθήσει να εισάγει τα στοιχεία αυτά στην βάση δημιουργώντας βασικά ένα νέο ασθενή. Σε περίπτωση επιτυχίας το σύστημα θα ενημερώσει τον χρήστη με το πιο κάτω μήνυμα και επίσης, ένα link που οδηγεί στον δημιουργημένο ασθενή.



Εικόνα 9: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή ασθενή στο σύστημα.

Σε περίπτωση αποτυχίας το σύστημα θα ενημερώσει με ανάλογο τρόπο το λόγο που απότυχε η δημιουργία του ασθενή.

4.3.2 Εύρεση Ασθενή (Patients->Find)

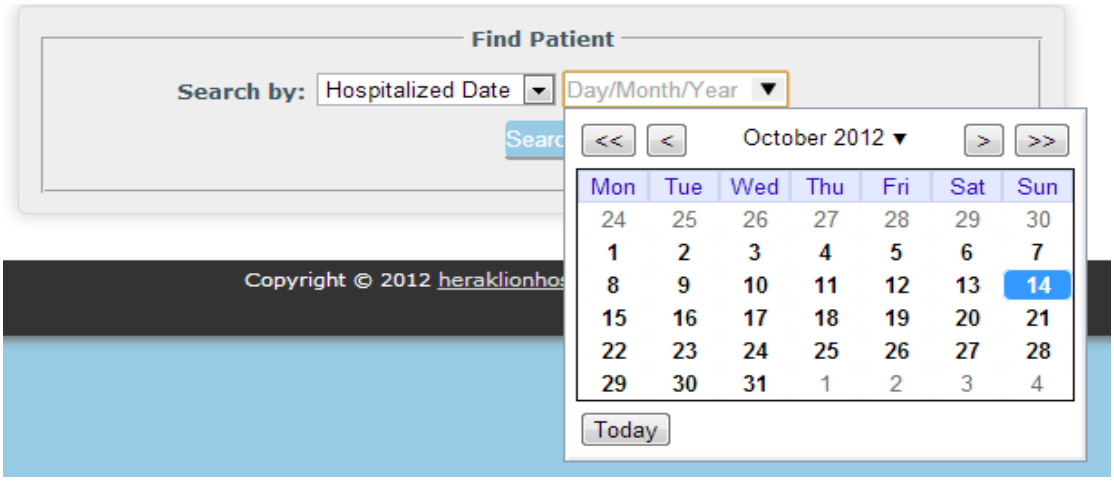
Εικόνα 10: Φόρμα εύρεσης ασθενή.

Η εφαρμογή επιτρέπει την εύρεση ενός καταχωρημένου ασθενή στο σύστημα με διάφορους τρόπους χρησιμοποιώντας ένα από τα διαθέσιμα κριτήρια:

- Name
- Political ID
- Birth Date
- Hospitalized Date

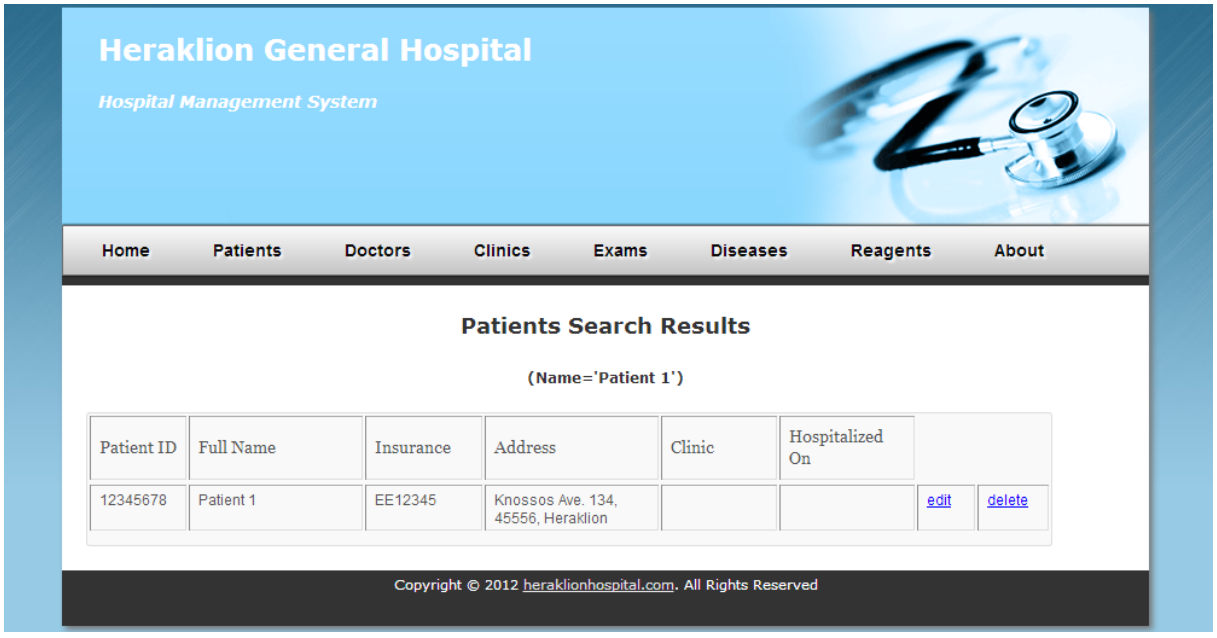
- Clinic

Με την βοήθεια της JavaScript, ανάλογα με τον τύπο του κριτηρίου που ο χρήστης επέλεξε τότε ο τύπος του πεδίου καταχώρησης αλλάζει αυτόματα και προσαρμόζεται στον τύπο του επιλεγμένου κριτηρίου. Για παράδειγμα, όταν ο χρήστης επιλέξει ένα πεδίο το οποίο απαιτεί ημερομηνία, τότε το πεδίο αναζήτησης θα αλλάξει σε date όπως δείχνει η πιο κάτω εικόνα:



Εικόνα 11: Προσαρμογή πεδίου αναζήτησης με JavaScript.

Όταν ο χρήστης πατήσει το κουμπί Search, τότε το σύστημα ελέγχει κατά πόσο υπάρχει ασθενής ή ασθενείς οι οποίοι πληρούν το συγκεκριμένο κριτήριο και εμφανίζει τα αποτελέσματα σε ένα πίνακα όπως φαίνεται στην πιο κάτω εικόνα.



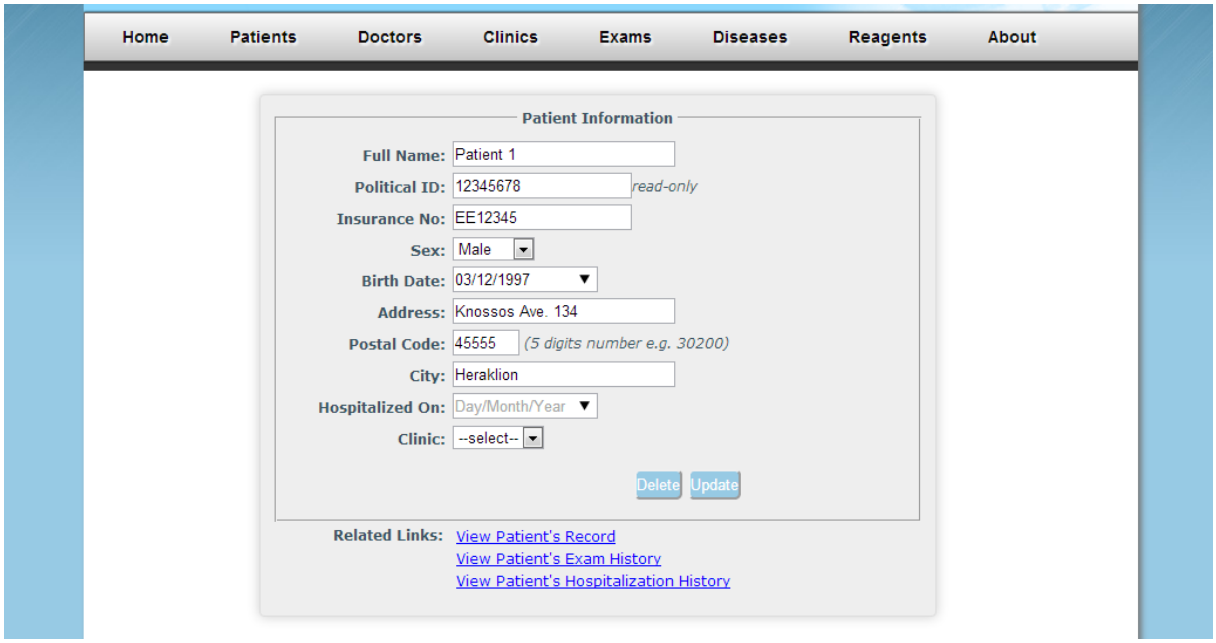
Εικόνα 12: Εμφάνιση αποτελεσμάτων κατά την εύρεση ασθενή.

4.3.3 Προβολή όλων των Ασθενών (Patients->View All)

Εικόνα 13: Σελίδα προβολής όλων των ασθενών.

Η πιο πάνω εικόνα δείχνει την σελίδα προβολής όλων των ασθενών που είναι καταχωρημένοι στο σύστημα. Όπως και στα αποτελέσματα εύρεσης ενός ασθενή, για κάθε ασθενή προβάλλονται κάποιες βασικές πληροφορίες και επίσης τα αντίστοιχα links για την επεξεργασία ή την διαγραφή του ασθενή από το σύστημα.

4.3.4 Επεξεργασία Ασθενή



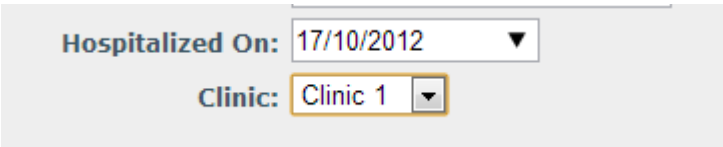
Εικόνα 14: Φόρμα επεξεργασίας ασθενή.

Στην Εικόνα 14 βλέπουμε την σελίδα στην οποία μπορούμε να επεξεργαστούμε υφισταμένους ασθενείς. Η αποθήκευση των αλλαγών στα στοιχεία του ασθενή επιτυγχάνεται με το πάτημα του κουμπιού **Update**. Πέρα από αυτό, η σελίδα αυτή επίσης προσφέρει στον χρήστη τις πιο κάτω επιλογές

- Διαγραφή ασθενή (**Delete**).
- Προβολή του ιστορικού ασθενείας του ασθενή (**View Patient's Record**).
- Προβολή του ιστορικού νοσηλείας του ασθενή (**View Patient's Hospitalization History**).

4.3.5 Έναρξη Νοσηλείας Ασθενή σε Κλινική

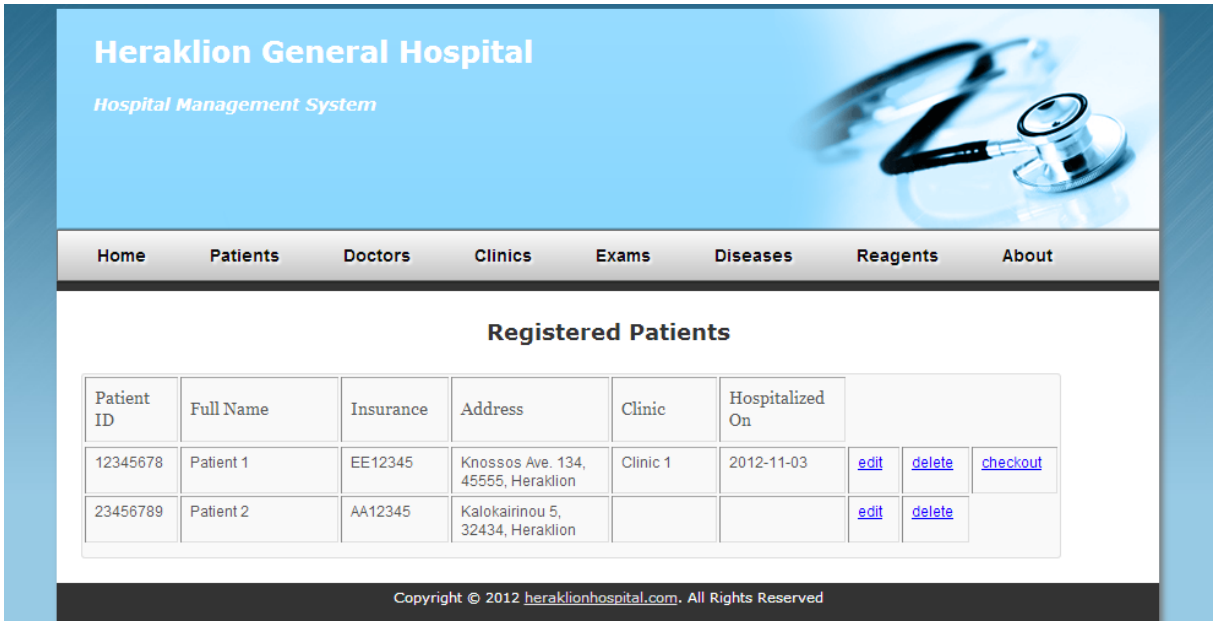
Η έναρξη νοσηλείας ενός ασθενή σε μία κλινική επιτυγχάνεται με τον καθορισμό των πεδίων **Clinic** (κλινική) και **Hospitalized On** (ημερομηνία εισαγωγής) είτε κατά την δημιουργία του ασθενή είτε κατά την επεξεργασία του. Όταν τα πεδία αυτά έχουν τιμή τότε αυτό σημαίνει πως ο συγκεκριμένος ασθενής νοσηλεύεται στην αναφερόμενη κλινική. Τα πεδία αυτά φαίνονται στην Εικόνα 20



Εικόνα 15: Πεδία για εισαγωγή ασθενή σε κλινική.

4.3.6 Λήξη Νοσηλείας Ασθενή σε Κλινική (Check out)

Σε περίπτωση που κάποιος ασθενής νοσηλεύται σε μία κλινική, τότε εμφανίζεται ο σύνδεσμος **Checkout** δίπλα από το record του ασθενή όπως φαίνεται στην πιο κάτω εικόνα.



Εικόνα 16: Επιλογή checkout για ασθενείς που νοσηλεύονται.

Πατώντας το **Checkout,** τότε αυτόματα ο ασθενής παίρνει εξιτήριο από την κλινική και όλες οι πληροφορίες σχετικά με την νοσηλεία του (έναρξη, λήξη, κλινική), καταγράφονται στον ιστορικό νοσηλείας του ασθενή.

4.3.7 Προβολή Ιστορικού Νοσηλείας Ασθενή

Εικόνα 17: Ιστορικό νοσηλείας ασθενή.

Η Εικόνα 17 δείχνει το ιστορικό νοσηλείας του ασθενή **Patient 1.** Η σελίδα αυτή θα εμφανιστεί πατώντας τον σύνδεσμο View Patient's Hospitalization History μέσα από την σελίδα επεξεργασίας του ασθενή. Επίσης, υπάρχει διαθέσιμο link για επιστροφή στις πληροφορίες του ασθενή.

4.3.8 Προβολή Ιστορικού Ασθενείας ενός Ασθενή

Εικόνα 18: Ιστορικό ασθενείας ενός ασθενή

Η Εικόνα 18 δείχνει το ιστορικό ασθενείας του ασθενή **Patient 1.** Η σελίδα αυτή θα εμφανιστεί πατώντας τον σύνδεσμο View Patient's Record μέσα από την σελίδα επεξεργασίας του ασθενή. Επίσης, υπάρχει διαθέσιμο link για επιστροφή στις πληροφορίες του ασθενή.

4.3.9 Προβολή Ιστορικού Εξετάσεων ενός Ασθενή

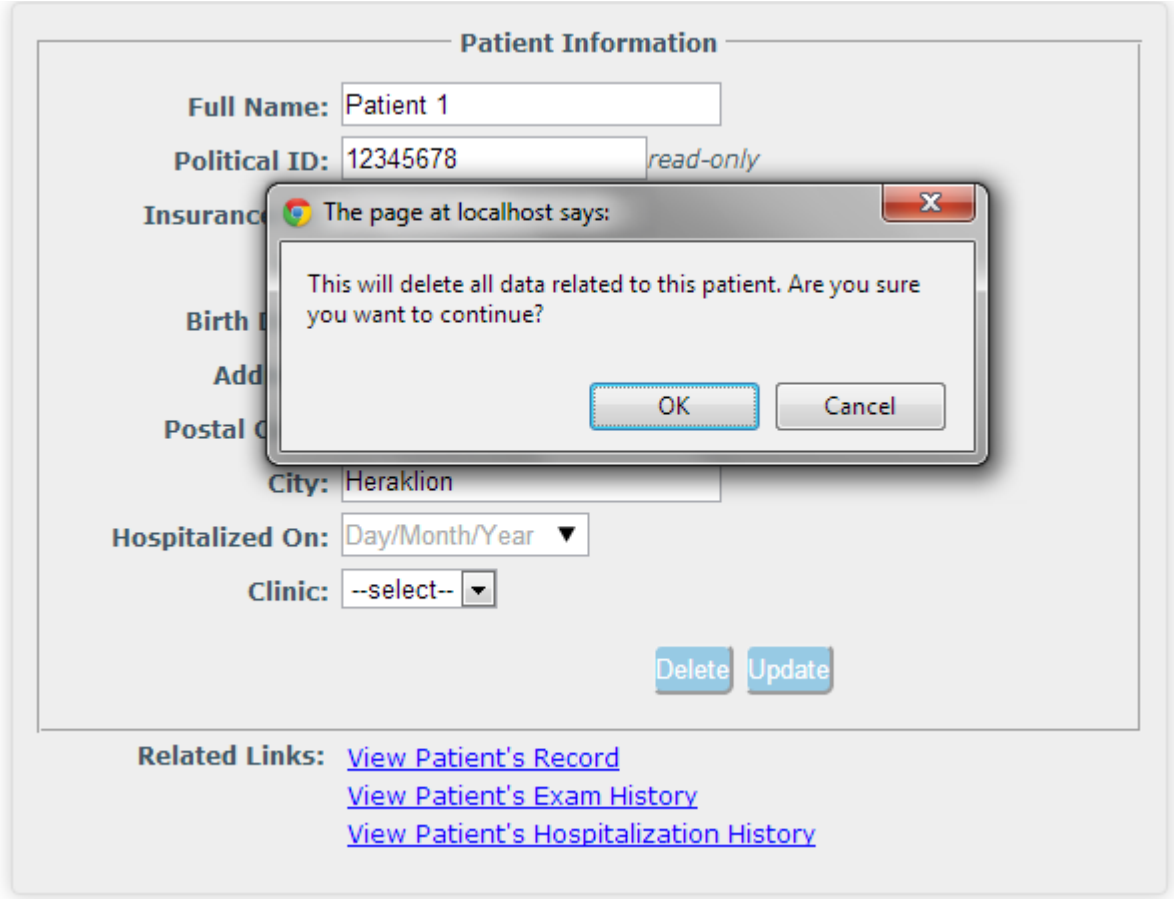
Εικόνα 19: Ιστορικό εξετάσεων ενός ασθενή

Η

Εικόνα 19 Εικόνα 18 δείχνει το ιστορικό εξετάσεων του ασθενή **Patient 1.** Η σελίδα αυτή θα εμφανιστεί πατώντας τον σύνδεσμο View Patient's Exam History μέσα από την σελίδα επεξεργασίας του ασθενή. Επίσης, υπάρχει διαθέσιμο link για επιστροφή στις πληροφορίες του ασθενή.

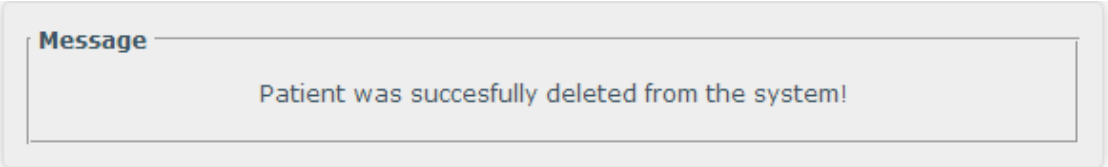
4.3.10 Διαγραφή Ασθενή

Σε περίπτωση που ο χρήστης επιλέξει την διαγραφεί ενός ασθενή, τότε το σύστημα θα εμφανίσει ένα παράθυρο επιβεβαίωσης ρωτώντας τον χρήστη αν όντως θέλει να προχωρήσει στην διαγραφή όπως φαίνεται πιο κάτω. Το παράθυρο επιβεβαίωσης εμφανίζεται για να αποφεύγεται η καταλάθος διαγραφή σε περιπτώσεις παρεξήγησης ή άγνοιας.



Εικόνα 20: Μήνυμα επιβεβαίωσης πριν την διαγραφή ασθενή.

Πατώντας OK τότε το σύστημα θα ψάξει και θα προσπαθήσει να διαγράψει από την βάση όλα τα δεδομένα τα οποία σχετίζονται με τον συγκεκριμένο ασθενή. Τα δεδομένα αυτά περιλαμβάνουν το ιστορικό ασθενείας, το ιστορικό νοσηλείας και επίσης το ιστορικό εξετάσεων του ασθενή. Σε περίπτωση που στην βάση δεδομένων υπάρχει οποιαδήποτε άλλη αναφορά στον ασθενή προς διαγραφή, τότε η διαδικασία θα αποτύχει και ο χρήστης θα ενημερωθεί για τον λόγο αποτυχίας. Σε περίπτωση που ο ασθενής διαγραφεί επιτυχώς από το σύστημα, τότε θα εμφανιστεί το πιο κάτω μήνυμα στον χρήστη.



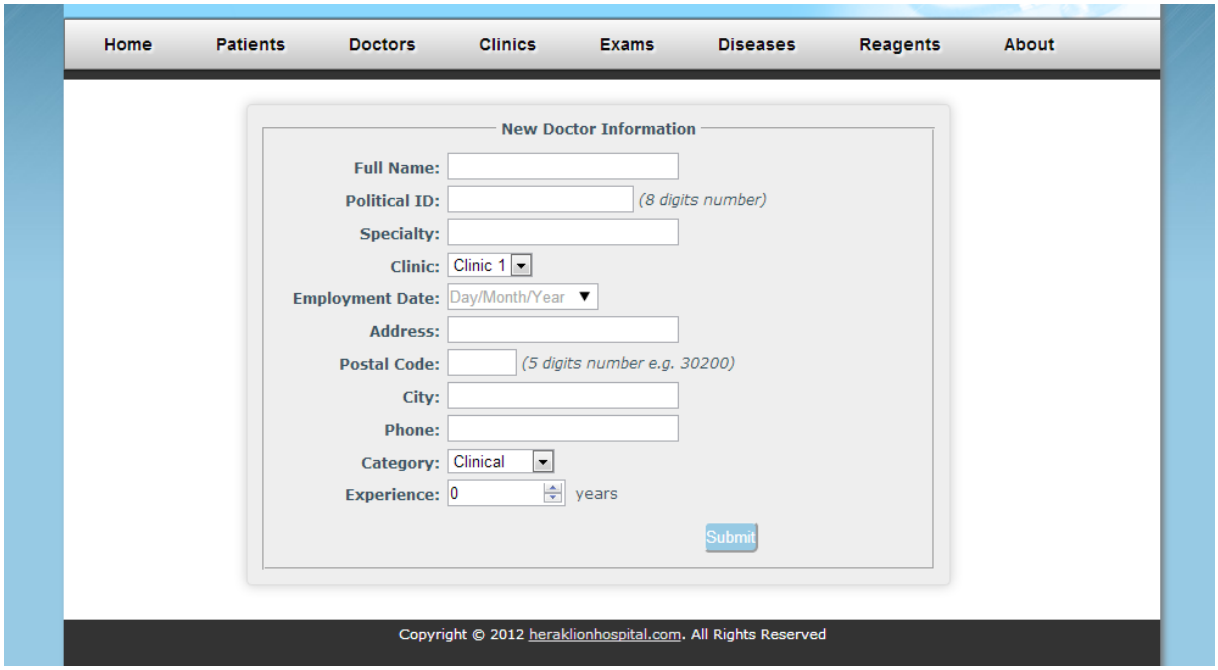
Εικόνα 21: Μήνυμα ενημέρωσης για επιτυχή διαγραφή ασθενή.

4.4 Γιατροί

Εικόνα 22: Doctors menu.

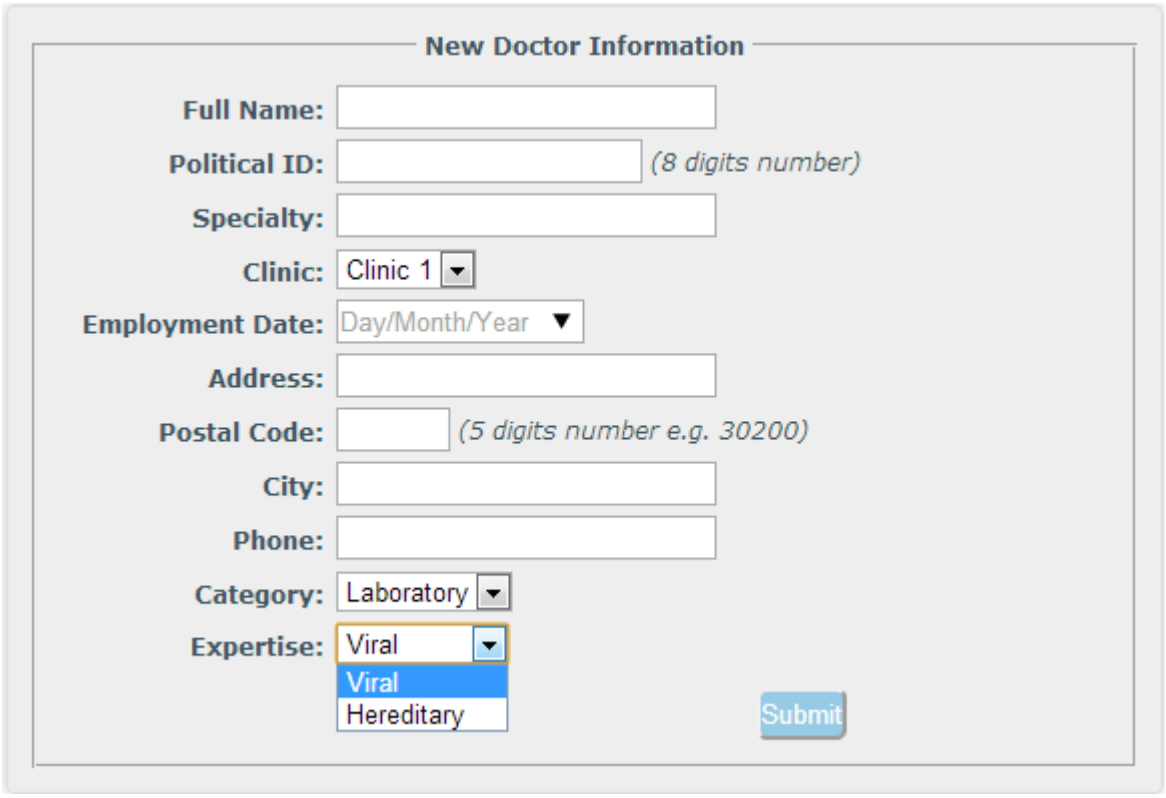
Η Εικόνα 22 δείχνει όλες τις διαδικασίες που μπορούμε να εκτελέσουμε από το μενού του **Doctors**. Θα ακολουθήσει λεπτομερής περιγραφή των διαδικασιών αυτών στα πιο κάτω υποκεφάλαια.

4.4.1 Δημιουργία Γιατρού (Doctors->Create new)



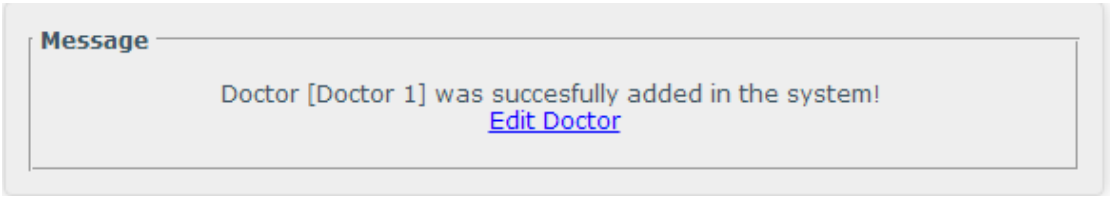
Εικόνα 23: Φόρμα για την δημιουργία γιατρού.

Η Εικόνα 23 δείχνει την φόρμα για την εισαγωγή ενός γιατρού στο σύστημα. Σύμφωνα με τις απαιτήσεις του συστήματος, κάθε γιατρός πρέπει υποχρεωτικά σε κάποια κλινική και για αυτό τον λόγο, στο πεδίο **Clinic** εμφανίζονται σε μία λίστα όλες οι καταχωρημένες κλινικές στο σύστημα. Επίσης, κάθε γιατρός πρέπει να ανήκει σε μία από τις διαθέσιμες κατηγορίες, **Clinical** ή **Laboratory**. Λόγω του ότι για κάθε κατηγορία χρειαζόμαστε διαφορετική πληροφορία (χρόνια εμπειρίας για τους κλινικούς και είδος ασθένειας για τους κλινικούς), με την χρήση JavaScript προσαρμόζουμε το τελευταίο πεδίο ανάλογα με την επιλογή που κάνει ο χρήστης στο πεδίο Category. Η λειτουργία αυτή φαίνεται στην πιο κάτω εικόνα όπου ο χρήστης επέλεξε την τιμή Laboratory ως Category.



Εικόνα 24: Χρήση JavaScript για δυναμική αλλαγή πεδίου.

Όταν ο χρήστης συμπληρώσει σωστά όλα τα απαιτούμενα πεδία και πατήσει το κουμπί **Submit**, τότε το σύστημα θα προσπαθήσει να εισάγει τα στοιχεία αυτά στην βάση δημιουργώντας βασικά ένα νέο γιατρό. Σε περίπτωση επιτυχίας το σύστημα θα ενημερώσει τον χρήστη με το πιο κάτω μήνυμα και επίσης, ένα link που οδηγεί στον δημιουργημένο γιατρό.



Εικόνα 25: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή γιατρού στο σύστημα.

Σε περίπτωση αποτυχίας το σύστημα θα ενημερώσει με ανάλογο τρόπο το λόγο που απότυχε η δημιουργία του ασθενή.

4.4.2 Εύρεση Γιατρού (Doctors->Find)

Εικόνα 26: Φόρμα εύρεσης γιατρού.

Η εφαρμογή επιτρέπει την εύρεση ενός καταχωρημένου γιατρού στο σύστημα με διάφορους τρόπους χρησιμοποιώντας ένα από τα διαθέσιμα κριτήρια:

- Name
- Specialty
- Clinic
- Employment Date

Όπως και στην φόρμα Εύρεση Ασθενή (Patients->Find), με την βοήθεια της JavaScript, ανάλογα με τον τύπο του κριτηρίου που ο χρήστης επέλεξε τότε ο τύπος του πεδίου αναζήτησης αλλάζει αυτόματα και προσαρμόζεται στον τύπο του επιλεγμένου κριτηρίου.

Όταν ο χρήστης πατήσει το κουμπί Search, τότε το σύστημα ελέγχει κατά πόσο υπάρχει γιατρός ή γιατροί οι οποίοι πληρούν το συγκεκριμένο κριτήριο και εμφανίζει τα αποτελέσματα σε ένα πίνακα όπως φαίνεται στην πιο κάτω εικόνα.

Εικόνα 27: Εμφάνιση αποτελεσμάτων κατά την εύρεση γιατρού.

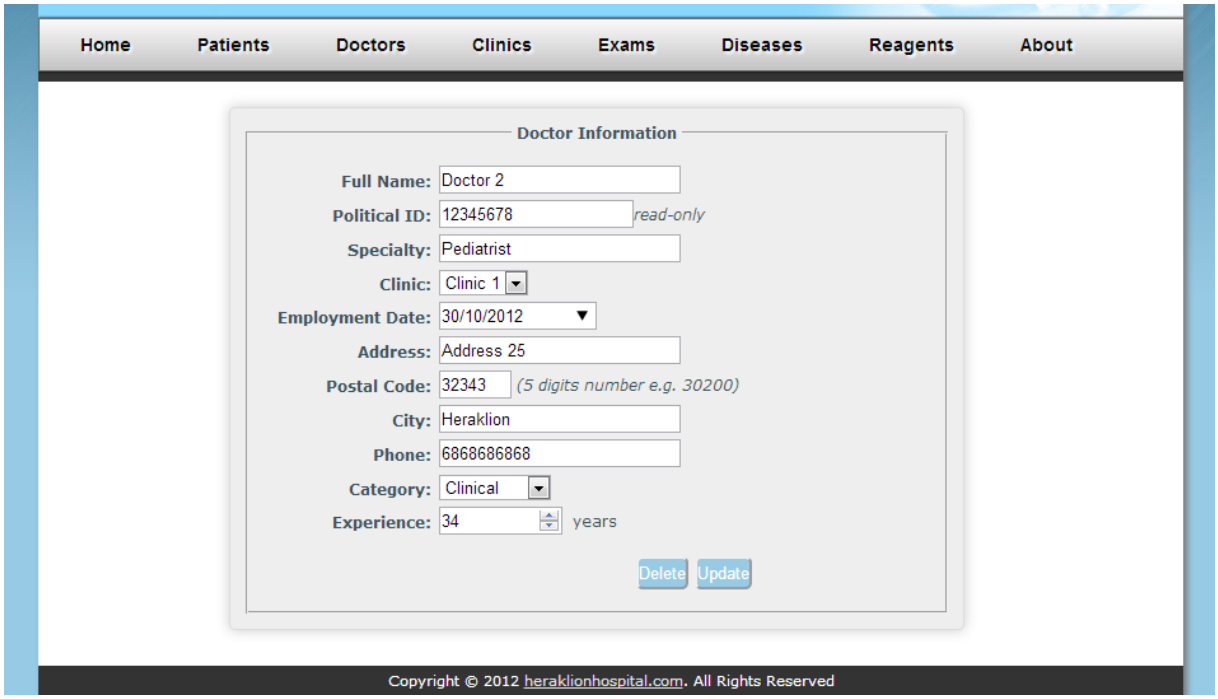
4.4.3 Προβολή όλων των Γιατρών

Εικόνα 28: Σελίδα προβολής όλων των γιατρών.

Η πιο πάνω εικόνα δείχνει την σελίδα προβολής όλων των γιατρών που είναι καταχωρημένοι στο σύστημα. Όπως και στα αποτελέσματα εύρεσης ενός γιατρού, για κάθε γιατρό προβάλλονται κάποιες βασικές πληροφορίες και επίσης τα αντίστοιχα links για την επεξεργασία ή την διαγραφή του γιατρού από το σύστημα. Το πεδίο Is Manager, δείχνει αν ο

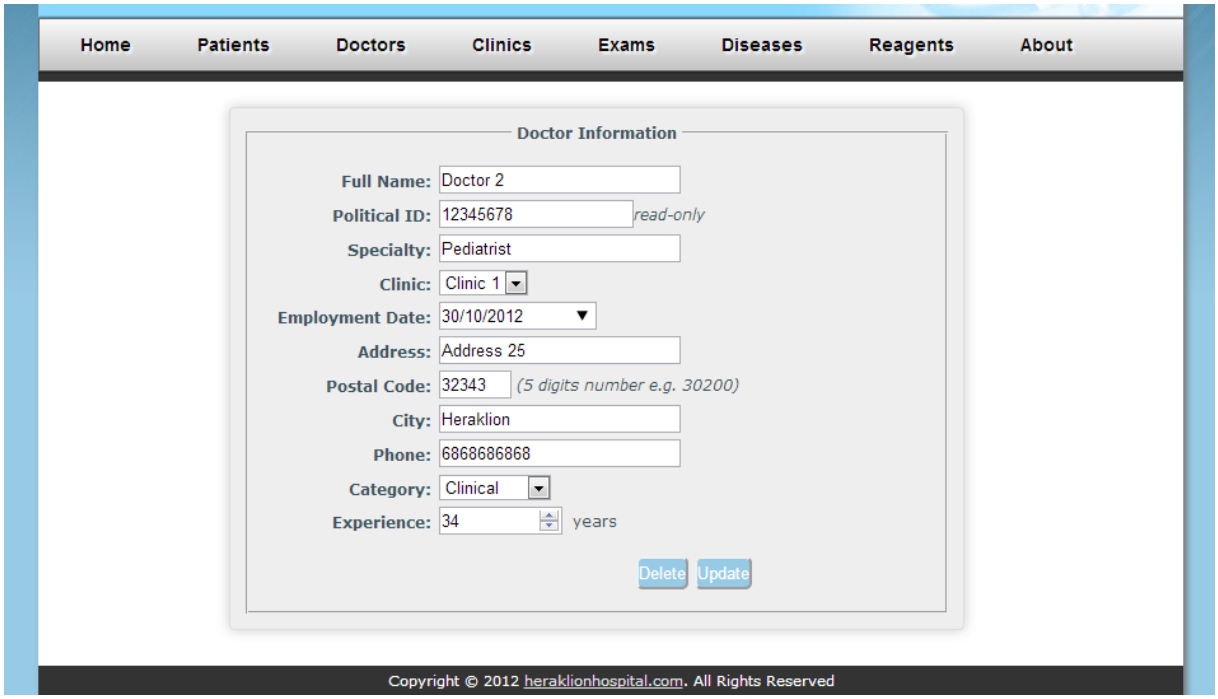
συγκεκριμένος γιατρός είναι μάνατζερ της κλινικής στην οποία ανήκει. Η διαδικασία εκχώρησης ενός γιατρού ως μάνατζερ μίας κλινικής περιγράφεται σε επόμενη ενότητα.

4.4.4 Επεξεργασία Γιατρού



Εικόνα 29: Φόρμα επεξεργασίας γιατρού.

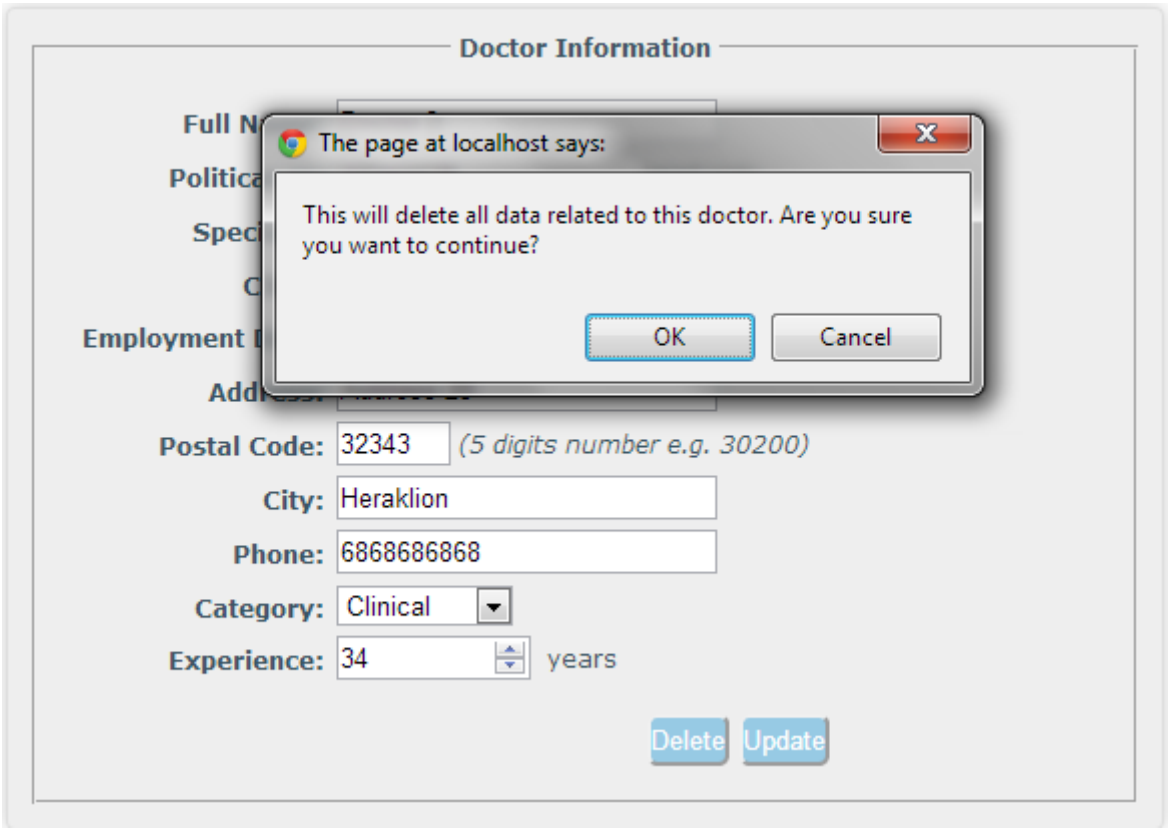
Στην



Εικόνα 29 βλέπουμε την σελίδα στην οποία μπορούμε να επεξεργαστούμε υφισταμένους γιατρούς. Η αποθήκευση των αλλαγών στα στοιχεία του γιατρού επιτυγχάνεται με το πάτημα του κουμπιού **Update**. Επίσης, υπάρχει διαθέσιμο το κουμπί **Delete** για την διαγραφή του συγκεκριμένου γιατρού.

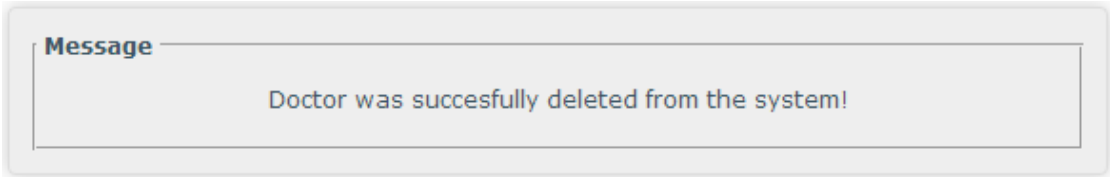
4.4.5 Διαγραφή Γιατρού

Σε περίπτωση που ο χρήστης επιλέξει την διαγραφή ενός γιατρού, τότε το σύστημα θα εμφανίσει ένα παράθυρο επιβεβαίωσης ρωτώντας τον χρήστη αν όντως θέλει να προχωρήσει στην διαγραφή όπως φαίνεται πιο κάτω. Το παράθυρο επιβεβαίωσης εμφανίζεται για να αποφεύγεται η καταλάθος διαγραφή σε περιπτώσεις παρεξήγησης ή άγνοιας.



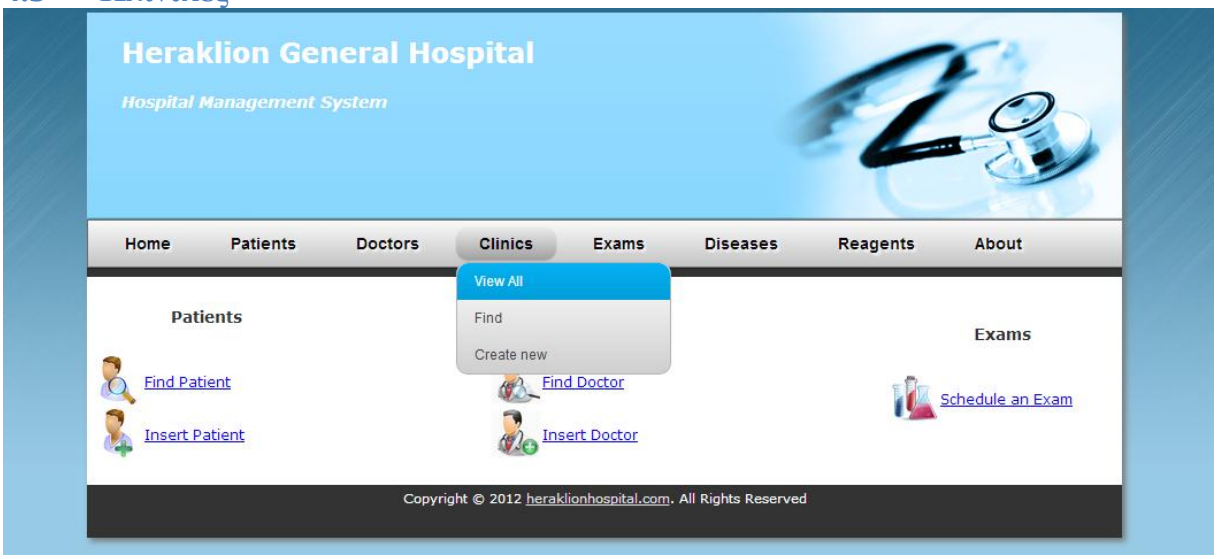
Εικόνα 30: Μήνυμα επιβεβαίωσης πριν την διαγραφή γιατρού.

Πατώντας OK τότε το σύστημα θα ψάξει και θα προσπαθήσει να διαγράψει από την βάση όλα τα δεδομένα τα οποία σχετίζονται με τον συγκεκριμένο γιατρό. Η διαδικασία αυτή θα αποτύχει σίγουρα σε περίπτωση που ο γιατρός προς διαγραφή είναι μάνατζερ της κλινικής στην οποία ανήκει. Σε περίπτωση που ο γιατρός διαγραφεί επιτυχώς από το σύστημα, τότε θα εμφανιστεί το πιο κάτω μήνυμα στον χρήστη.



Εικόνα 31: Μήνυμα ενημέρωσης για επιτυχή διαγραφή γιατρού.

4.5 Κλινικές



Εικόνα 32: Clinics menu.

Η Εικόνα 32 δείχνει όλες τις διαδικασίες που μπορούμε να εκτελέσουμε από το μενού του **Clinics**. Θα ακολουθήσει λεπτομερής περιγραφή των διαδικασιών αυτών στα πιο κάτω υποκεφάλαια.

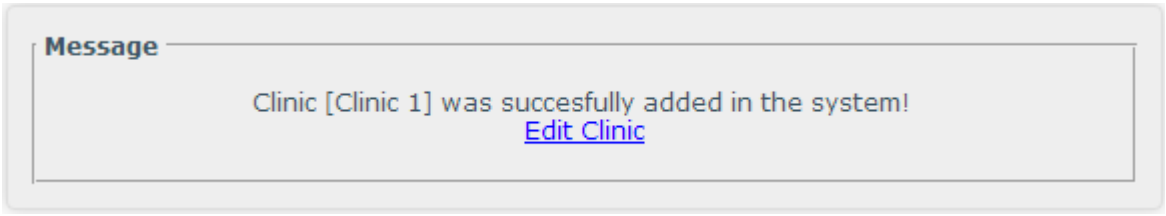
4.5.1 Δημιουργία Κλινικής (Clinics->Create new)

Εικόνα 33: Φόρμα για την δημιουργία κλινικής.

Η Εικόνα 23 δείχνει την φόρμα για την εισαγωγή μίας κλινικής στο σύστημα. Εδώ μπορούμε να δούμε την χρήση ενός νέου input type από την HTML5, του **number**.

Εικόνα 34: HTML5 number input type.

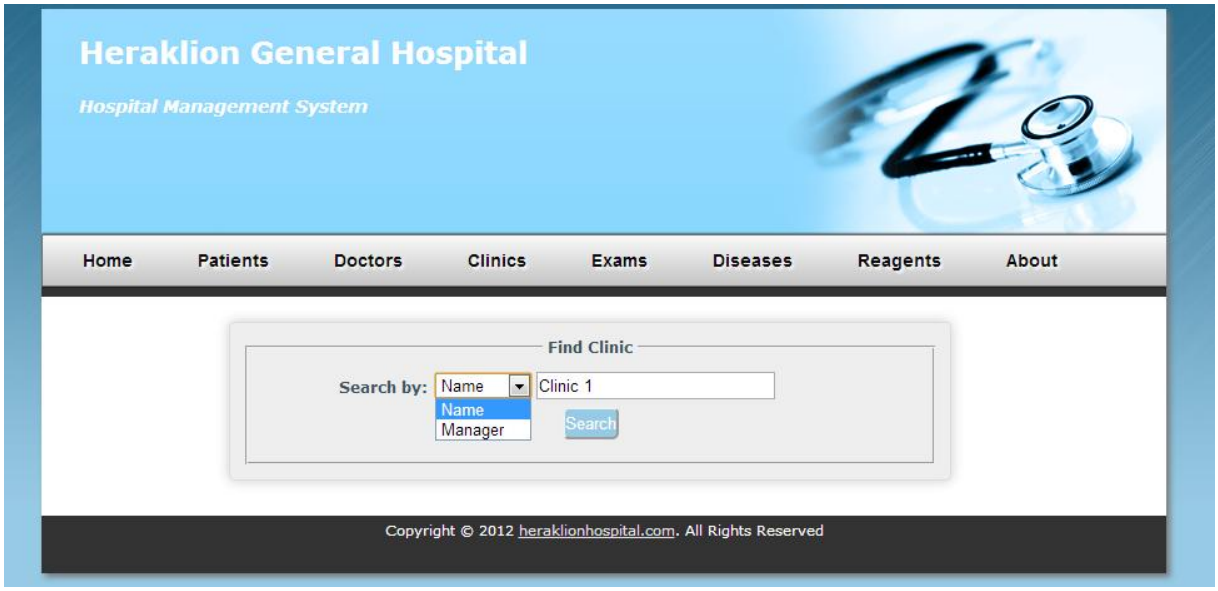
Όταν ο χρήστης συμπληρώσει σωστά όλα τα απαιτούμενα πεδία και πατήσει το κουμπί **Submit**, τότε το σύστημα θα προσπαθήσει να εισάγει τα στοιχεία αυτά στην βάση δημιουργώντας βασικά μία νέα κλινική. Σε περίπτωση επιτυχίας το σύστημα θα ενημερώσει τον χρήστη με το πιο κάτω μήνυμα και επίσης, ένα link που οδηγεί στην δημιουργημένη κλινική.



Εικόνα 35: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή κλινικής στο σύστημα.

Σε περίπτωση αποτυχίας το σύστημα θα ενημερώσει με ανάλογο τρόπο το λόγο που απότυχε η δημιουργία της κλινικής.

4.5.2 Εύρεση Κλινικής(Clinics->Find)



Εικόνα 36: Φόρμα εύρεσης κλινικής.

Η εφαρμογή επιτρέπει την εύρεση μίας καταχωρημένης κλινικής στο σύστημα με χρησιμοποιώντας ένα από τα διαθέσιμα κριτήρια:

- Name
- Manager

Όταν ο χρήστης πατήσει το κουμπί Search, τότε το σύστημα ελέγχει κατά πόσο υπάρχει κλινική ή κλινικές που πληρούν το συγκεκριμένο κριτήριο και εμφανίζει τα αποτελέσματα σε ένα πίνακα όπως φαίνεται στην πιο κάτω εικόνα.

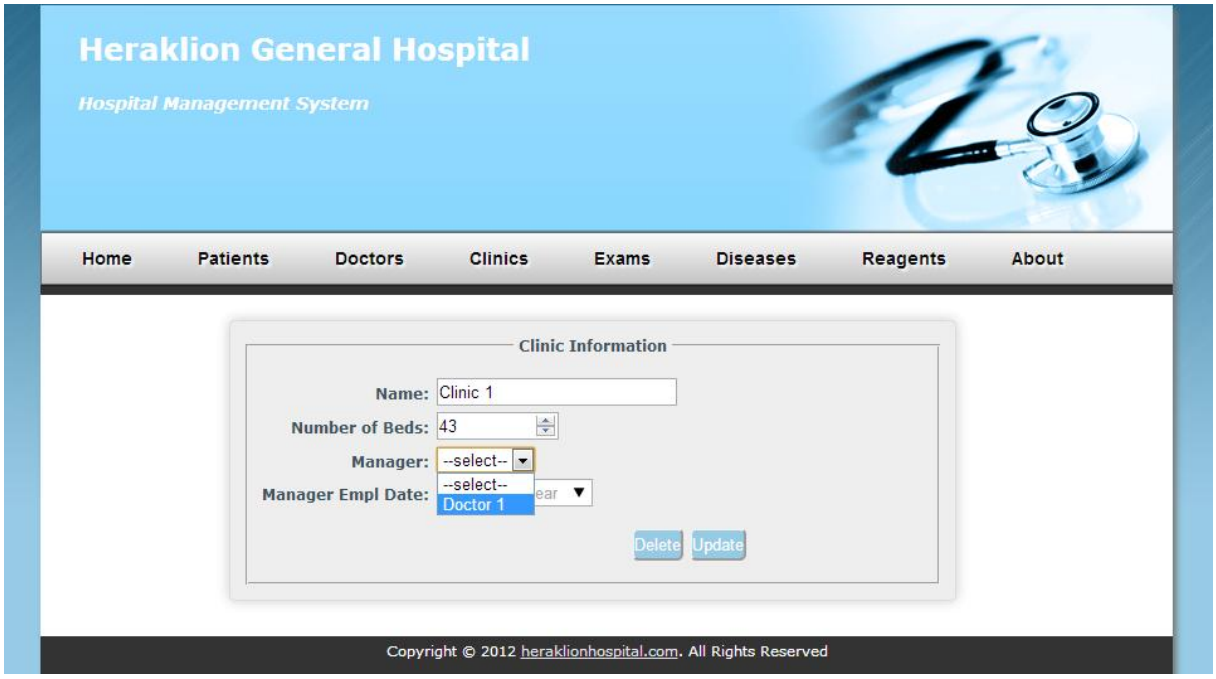
Εικόνα 37: Εμφάνιση αποτελεσμάτων κατά την εύρεση κλινικής.

4.5.3 Προβολή όλων των Κλινικών (Clinics->View All)

Εικόνα 38: Σελίδα προβολής όλων των κλινικών.

Η πιο πάνω εικόνα δείχνει την σελίδα προβολής όλων των γιατρών που είναι καταχωρημένοι στο σύστημα. Για κάθε κλινική προβάλλονται κάποιες βασικές πληροφορίες και επίσης τα αντίστοιχα links για την επεξεργασία ή την διαγραφή της κλινικής από το σύστημα. Το πεδίο Manager, δείχνει τον γιατρό ο οποίος είναι μάνατζερ της κάθε κλινικής.

4.5.4 Επεξεργασία Κλινικής & Ορισμός Manager

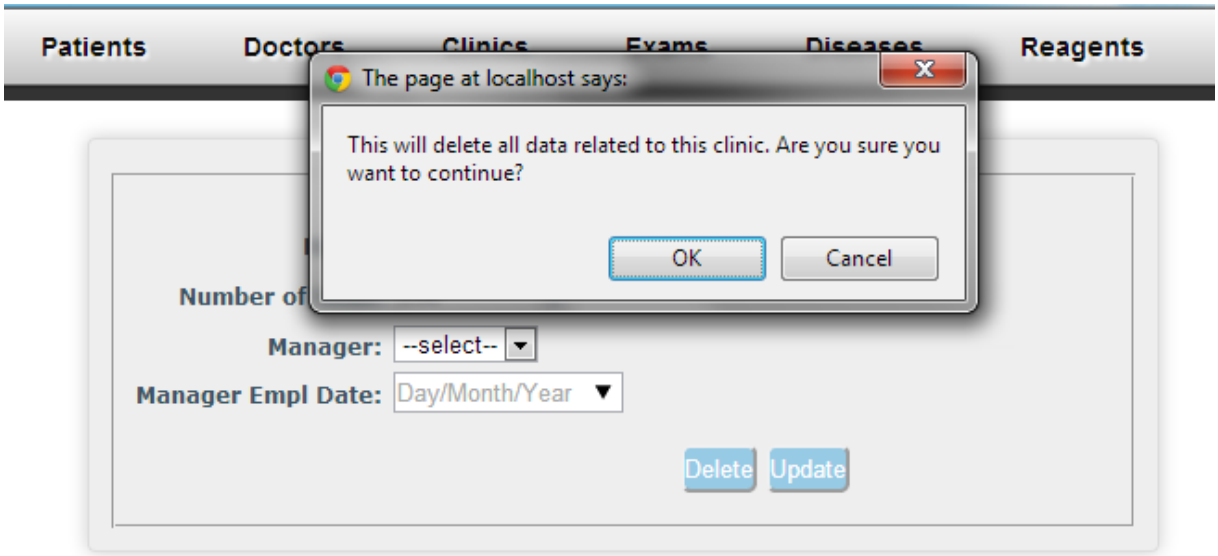


Εικόνα 39: Φόρμα επεξεργασίας κλινικής.

Στην Εικόνα 39 βλέπουμε την σελίδα στην οποία μπορούμε να επεξεργαστούμε μία καταχωρημένη κλινική. Όπως και στις υπόλοιπες φόρμες επεξεργασίας, έτσι και εδώ είναι διαθέσιμα τα κουμπιά **Update** και **Delete.** Όπως βλέπετε, στην επεξεργασία κλινικής υπάρχουν 2 επιπλέον πεδία, **Manager** και **Manager Empl. Date**, τα οποία δεν υπάρχουν κατά την δημιουργία της κλινικής και που με αυτά επιτυγχάνεται η διαδικασία εκχώρησης ενός γιατρού ως manager μίας κλινικής. Ο λόγος που τα πεδία αυτά εμφανίζονται μόνο εδώ είναι γιατί ως μάνατζερ μιας κλινικής μπορεί να οριστεί μόνο κάποιος γιατρός ο οποίος δουλεύει για την συγκεκριμένη κλινική.

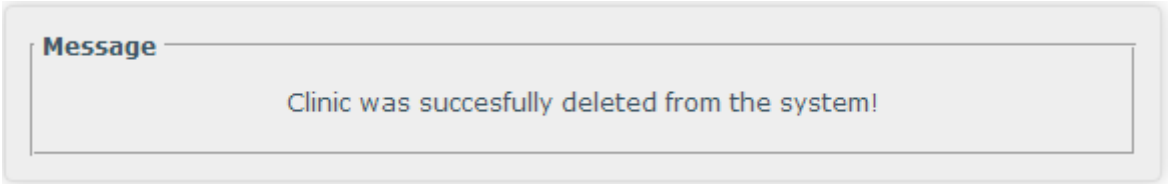
4.5.5 Διαγραφή Κλινικής

Σε περίπτωση που ο χρήστης επιλέξει την διαγραφή μίας κλινικής, τότε το σύστημα θα εμφανίσει ένα παράθυρο επιβεβαίωσης ρωτώντας τον χρήστη αν όντως θέλει να προχωρήσει στην διαγραφή όπως φαίνεται πιο κάτω. Το παράθυρο επιβεβαίωσης εμφανίζεται για να αποφεύγεται η καταλάθος διαγραφή σε περιπτώσεις παρεξήγησης ή άγνοιας.



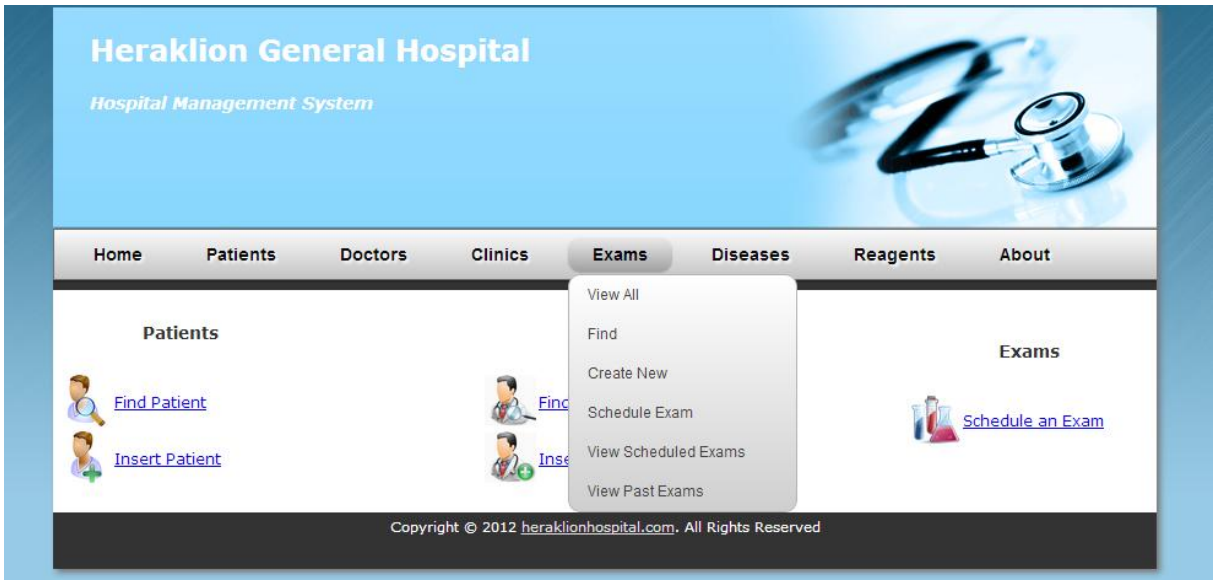
Εικόνα 40: Μήνυμα επιβεβαίωσης πριν την διαγραφή κλινικής.

Πατώντας OK τότε το σύστημα θα ψάξει και θα προσπαθήσει να διαγράψει από την βάση όλα τα δεδομένα τα οποία σχετίζονται με την συγκεκριμένο κλινική. Η διαδικασία αυτή θα αποτύχει σίγουρα σε περίπτωση που στην βάση δεδομένων υπάρχουν αναφορές στην συγκεκριμένη κλινική.



Εικόνα 41: Μήνυμα ενημέρωσης για επιτυχή διαγραφή κλινικής.

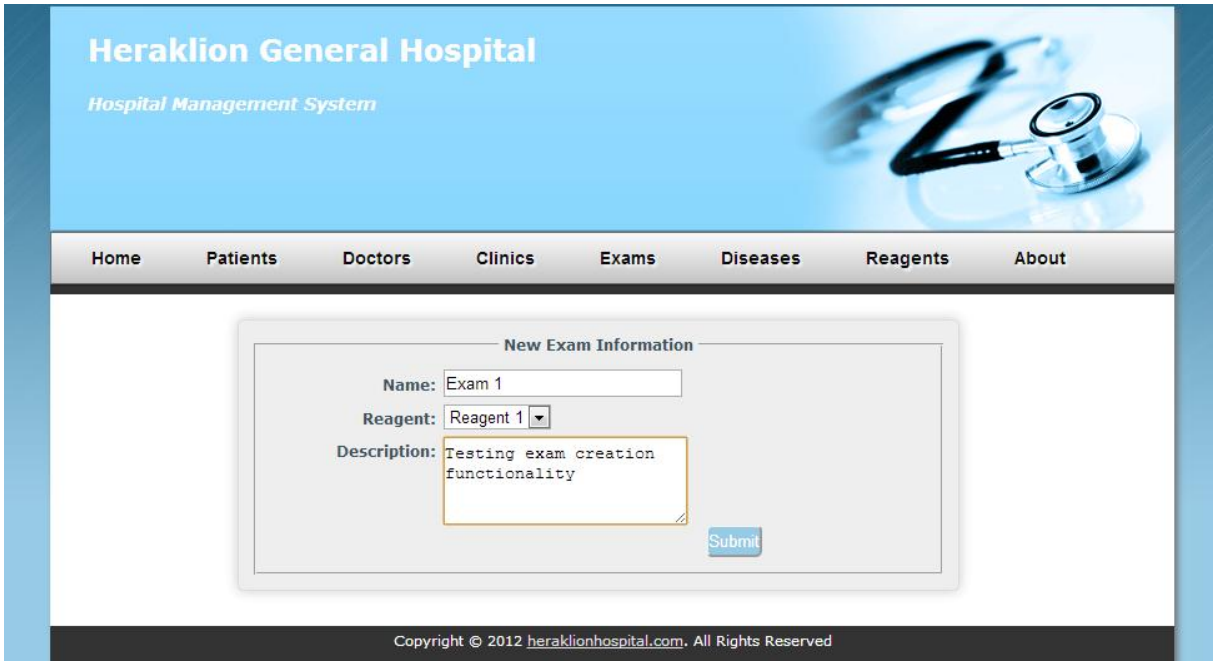
4.6 Εξετάσεις



Εικόνα 42: Exams menu.

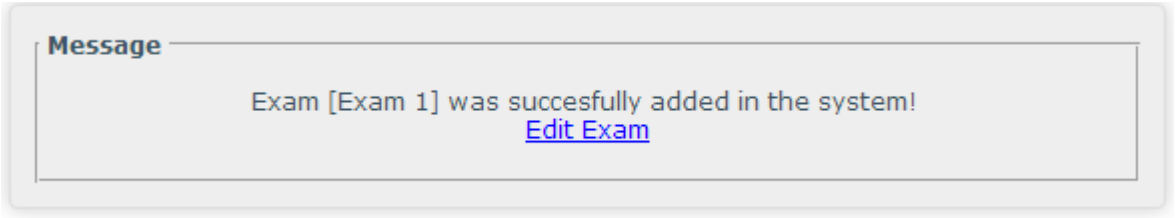
Η Εικόνα 42 δείχνει όλες τις διαδικασίες που μπορούμε να εκτελέσουμε από το μενού του **Exams**. Θα ακολουθήσει λεπτομερής περιγραφή των διαδικασιών αυτών στα πιο κάτω υποκεφάλαια.

4.6.1 Δημιουργία Εξέτασης



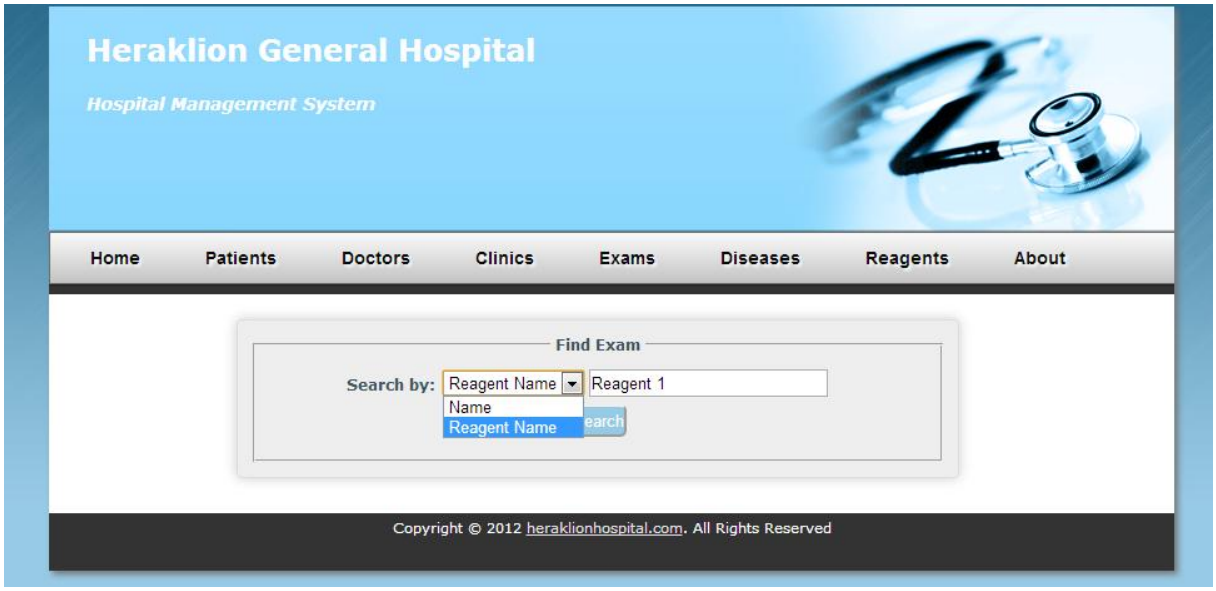
Εικόνα 43: Φόρμα για την δημιουργία εξέτασης.

Η Εικόνα 43 δείχνει την φόρμα για την καταχώρηση μίας εξέτασης στο σύστημα. Αυτή η διαδικασία δεν αναφέρεται στην δημιουργία εξέτασης για ένα ασθενή, αλλά τον καθορισμό ενός τύπου εξέτασης που μπορεί να γίνει σε πολλούς ασθενείς. Ο χρήστης πρέπει να δώσει το όνομα της εξέτασης, το αντιδραστήριο που χρησιμοποιεί και επίσης μια περιγραφή της εξέτασης (προαιρετικό). Όταν ο χρήστης συμπληρώσει τα απαιτούμενα πεδία και πατήσει το κουμπί **Submit**, τότε το σύστημα θα προσπαθήσει να καταχωρήσει την εξέταση στο σύστημα. Σε περίπτωση επιτυχίας το σύστημα θα ενημερώσει τον χρήστη με το πιο κάτω μήνυμα και επίσης, ένα link που οδηγεί στην εξέταση που μόλις δημιουργήθηκε.



Εικόνα 44: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή εξέτασης στο σύστημα.

4.6.2 Εύρεση Εξέτασης



Εικόνα 45: Φόρμα εύρεσης ασθένειας.

Η εφαρμογή επιτρέπει την εύρεση μίας καταχωρημένης κλινικής στο σύστημα με χρησιμοποιώντας ένα από τα διαθέσιμα κριτήρια:

- Name
- Reagent Name

Όταν ο χρήστης πατήσει το κουμπί Search, τότε το σύστημα ελέγχει κατά πόσο υπάρχει εξέταση ή εξετάσεις που πληρούν το συγκεκριμένο κριτήριο και εμφανίζει τα αποτελέσματα σε ένα πίνακα όπως φαίνεται στην πιο κάτω εικόνα.

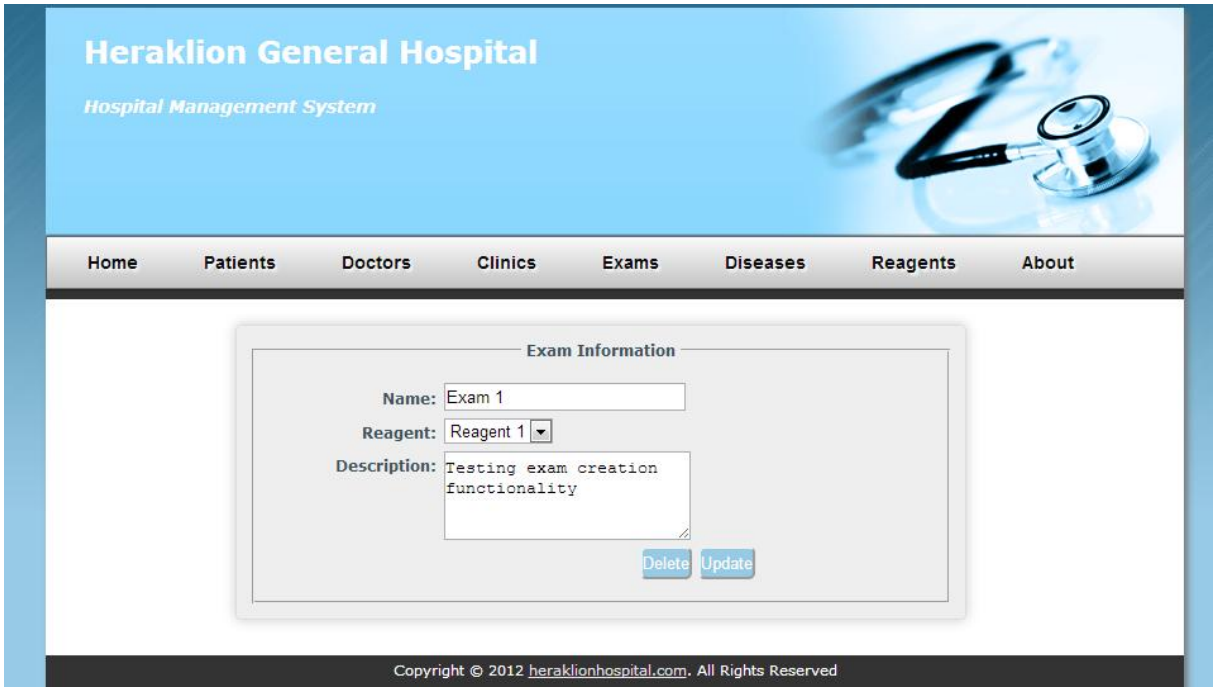
Εικόνα 46: Εμφάνιση αποτελεσμάτων κατά την εύρεση εξέτασης.

4.6.3 Προβολή όλων των εξετάσεων

Εικόνα 47: Σελίδα προβολής όλων των εξετάσεων.

Η πιο πάνω εικόνα δείχνει την σελίδα προβολής όλων των εξετάσεων που είναι καταχωρημένοι στο σύστημα. Για κάθε εξέταση προβάλλονται κάποιες βασικές πληροφορίες και επίσης τα αντίστοιχα links για την επεξεργασία ή την διαγραφή της εξέτασης από το σύστημα.

4.6.4 Επεξεργασία Εξέτασης

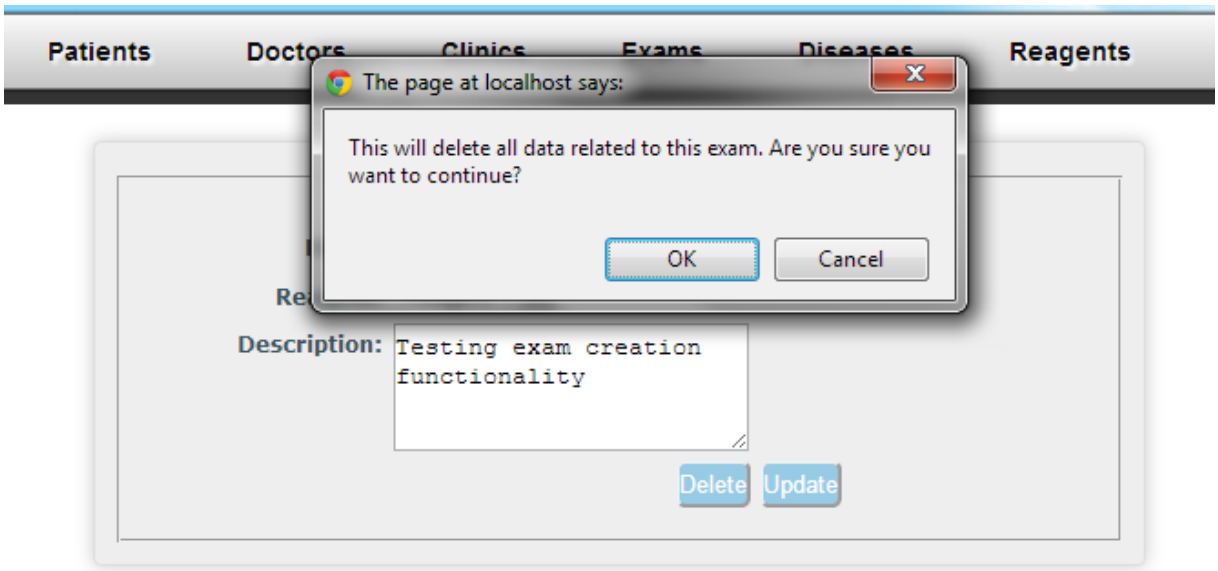


Εικόνα 48: Φόρμα επεξεργασίας εξέτασης.

Το σύστημα επιτρέπει στους χρήστες την επεξεργασία των στοιχείων μιας καταχωρημένης εξέτασης στο σύστημα. Η αποθήκευση των αλλαγών επιτυγχάνεται με το πάτημα του κουμπιού **Update**. Επίσης, υπάρχει διαθέσιμο το κουμπί **Delete** για την διαγραφή της συγκεκριμένης εξέτασης από το σύστημα.

4.6.5 Διαγραφή Εξέτασης

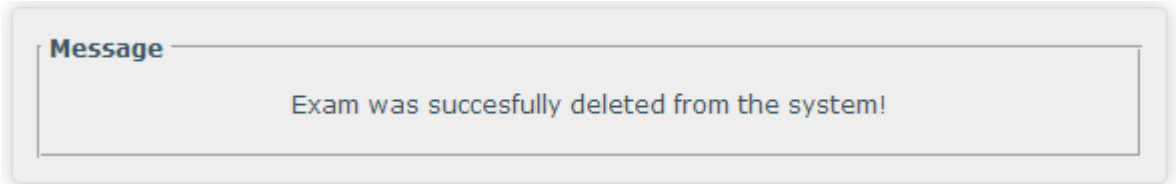
Σε περίπτωση που ο χρήστης επιλέξει την διαγραφή μιας εξέτασης, τότε το σύστημα θα εμφανίσει ένα παράθυρο επιβεβαίωσης ρωτώντας τον χρήστη αν όντως θέλει να προχωρήσει στην διαγραφή όπως φαίνεται πιο κάτω. Το παράθυρο επιβεβαίωσης εμφανίζεται για να αποφεύγεται η καταλάθος διαγραφή μίας καταχώρησης.



Εικόνα 49: Μήνυμα επιβεβαίωσης πριν την διαγραφή εξέτασης.

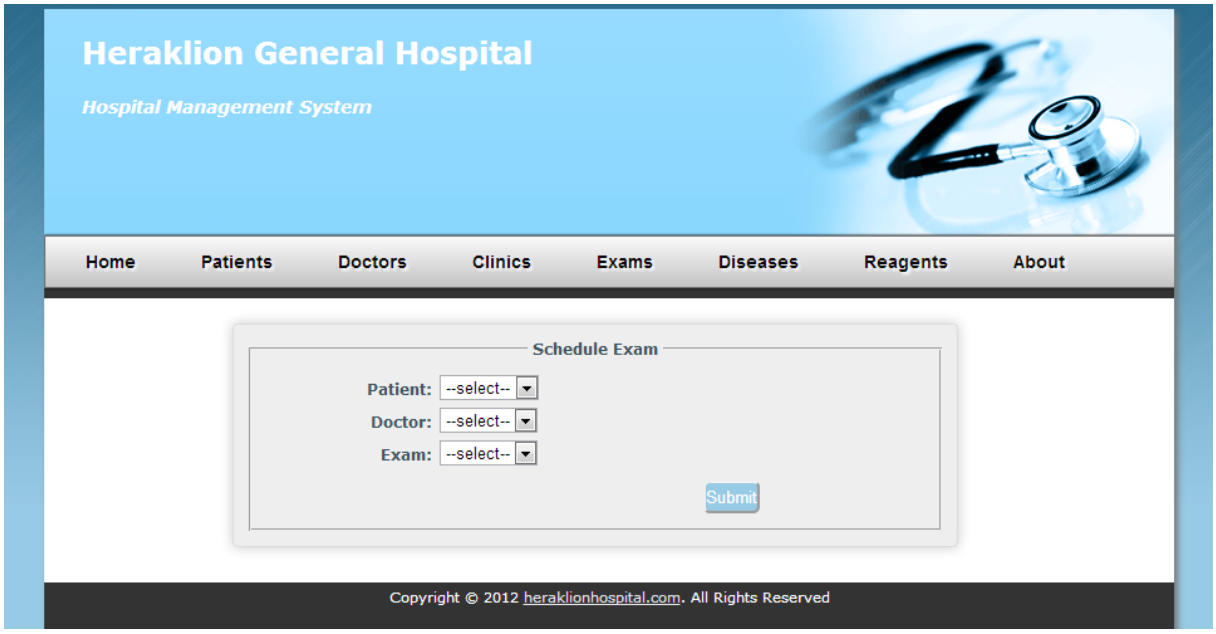
Πατώντας OK τότε το σύστημα θα ψάξει και θα προσπαθήσει να διαγράψει από την βάση την συγκεκριμένη εξέταση. Το σύστημα θα αποτύχει να διαγράψει μια εξέταση όταν υπάρχει έστω και μία αναφορά στην βάση δεδομένων για την συγκεκριμένη εξέταση.

Σε περίπτωση επιτυχούς διαγραφής θα εμφανιστεί το πιο κάτω μήνυμα.



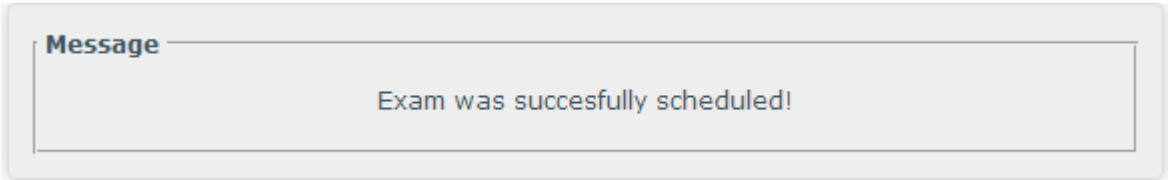
Εικόνα 50: Μήνυμα ενημέρωσης για επιτυχή διαγραφή εξέτασης.

4.6.6 Προγραμματισμός εξέτασης σε ασθενή



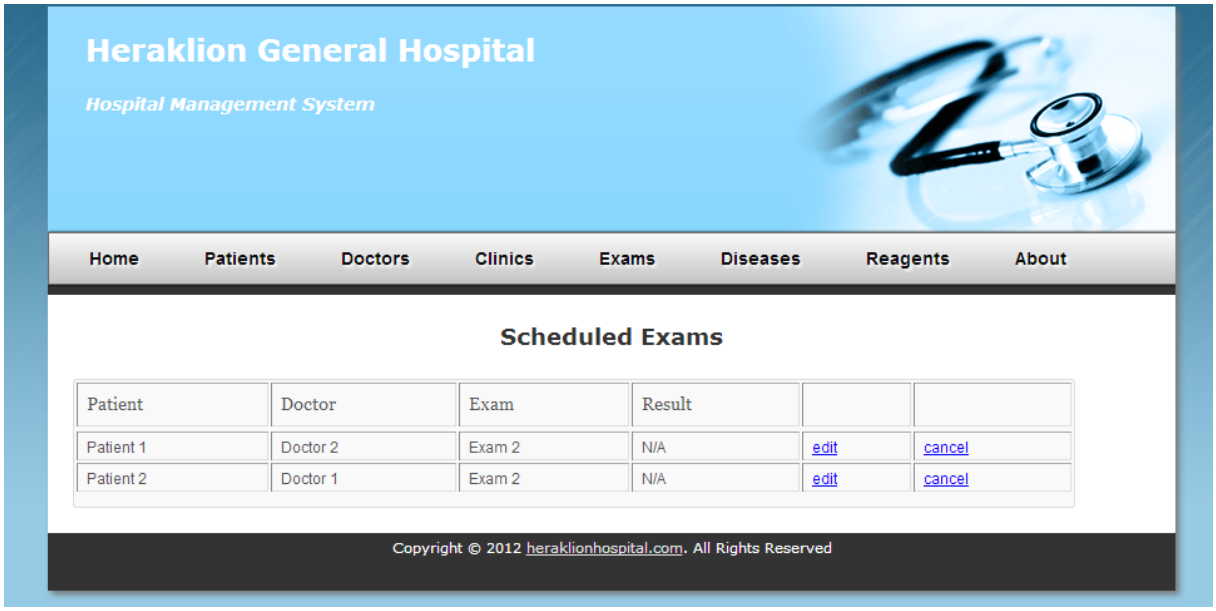
Εικόνα 51: Προγραμματισμός εξέτασης σε ασθενή.

Η Εικόνα 51 δείχνει την φόρμα με την οποία μπορεί να προγραμματιστεί μία νέα εξέταση για ένα ασθενή. Για κάθε προγραμματισμένη εξέταση ο χρήστης πρέπει να καθορίσει τον ασθενή που θα κάνει την εξέταση, τον γιατρό οποίος διέταξε την διεξαγωγή της εξέτασης και φυσικά, τι είδους εξέταση θα κάνει. Όταν ο χρήστης συμπληρώσει όλα τα απιατούμενα πεδία και πατήσει το κουμπί **Submit**, τότε το σύστημα θα προσπαθήσει να καταχωρήσει την εξέταση στο σύστημα. Σε περίπτωση επιτυχίας το σύστημα θα ενημερώσει τον χρήστη με το πιο κάτω μήνυμα.



Εικόνα 52: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή εξέτασης ασθενή.

4.6.7 Προβολή όλων των προγραμματισμένων εξετάσεων



Εικόνα 53: Σελίδα προβολής όλων των προγραμματισμένων εξετάσεων

Η πιο πάνω εικόνα δείχνει την σελίδα προβολής όλων των προγραμματισμένων εξετάσεων που είναι καταχωρημένες στο σύστημα. Για κάθε προγραμματισμένη εξέταση προβάλλονται ο ασθενής που θα την κάνει, ο γιατρός που την διέταξε και επίσης το είδος της εξέτασης. Επίσης για κάθε record υπάρχουν διαθέσιμα links για την επεξεργασία ή την ακύρωση της εξέτασης.

4.6.8 Ακύρωση προγραμματισμένης εξέτασης

Το σύστημα παρέχει την δυνατότητα ακύρωσης μιας προγραμματισμένης εξέτασης σε ένα ασθενή πατώντας τον σύνδεσμο **Cancel,** ο οποίος φαίνεται στην Εικόνα 53. Σε περίπτωση επιτυχούς ακύρωσης της εξέτασης θα εμφανιστεί το πιο κάτω μήνυμα.

Εικόνα 54: Μήνυμα ενημέρωσης για επιτυχή ακύρωση εξέτασης.

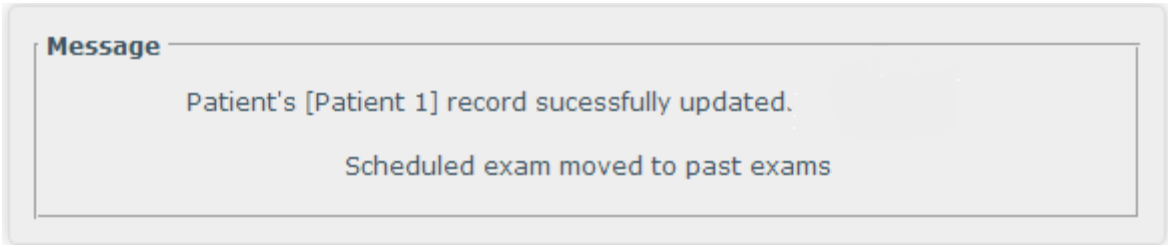
4.6.9 Επεξεργασία Προγραμματισμένης Εξέτασης

Εικόνα 55:Φόρμα επεξεργασίας προγραμματισμένης εξέτασης.

Το σύστημα επιτρέπει στους χρήστες την επεξεργασία των στοιχείων μιας προγραμματισμένης εξέτασης. Η αποθήκευση των αλλαγών επιτυγχάνεται με το πάτημα του κουμπιού **Update**. Επίσης, υπάρχει διαθέσιμο το κουμπί **Delete** για την διαγραφή/ακύρωση της συγκεκριμένης εξέτασης από το σύστημα.

Με αυτή την φόρμα καθορίζεται και το αποτέλεσμα μιας εξέτασης, το οποίο μπορεί να είναι θετικό ή αρνητικό. Το σύστημα ανάλογα με το αποτέλεσμα μιας προγραμματισμένης εξέτασης θα εκτελέσει τις ακόλουθες ενέργειες:

1. Θα δημιουργήσει μια νέα εγγραφή στο ιστορικό ασθενείας του ασθενή με όλες τις απαραίτητες πληροφορίες. (Result = TRUE).
2. Θα δημιουργήσει μια νέα εγγραφή στο ιστορικό εξετάσεων του ασθενή.
3. Θα δημιουργήσει μια νέα εγγραφή στο ιστορικό εξετάσεων του νοσοκομείου.



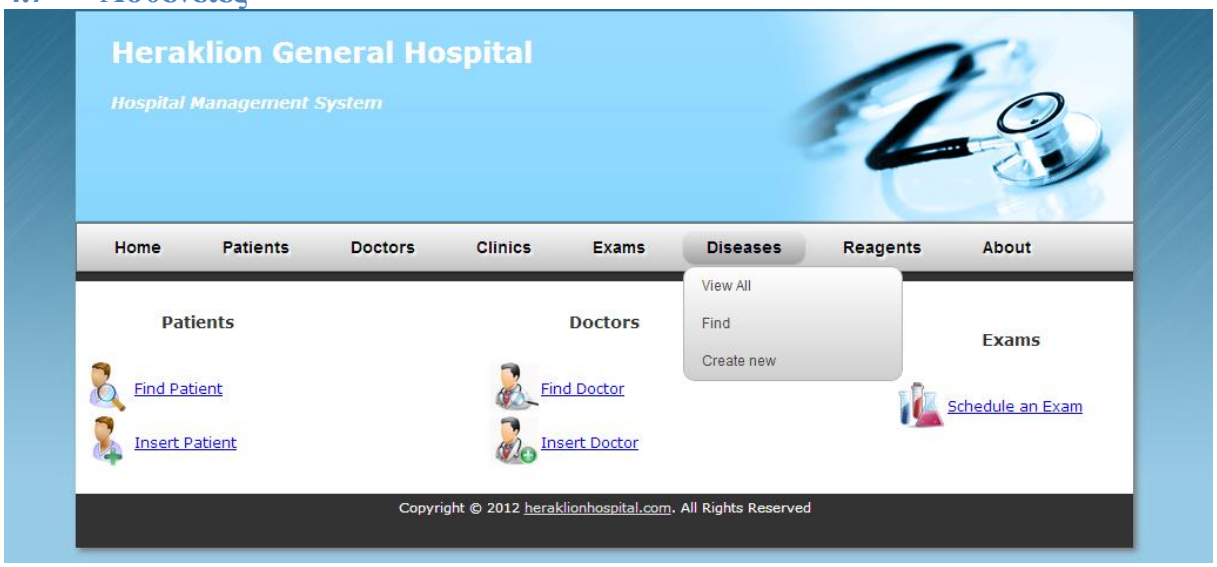
Εικόνα 56: Μήνυμα ενημέρωσης για τον καθορισμό αποτελέσματος μιας εξέτασης.

4.6.10 Προβολή ιστορικού εξετάσεων του νοσοκομείου

Εικόνα 57: Σελίδα προβολής προηγούμενων εξετάσεων

Η Εικόνα 57 δείχνει την σελίδα προβολής όλων των περασμένων εξετάσεων που έγιναν στο νοσοκομείο. Για κάθε εγγραφή προβάλλονται όλες οι απαραίτητες πληροφορίες καθώς και η ημερομηνία που βγήκανε τα αποξέσματα της κάθε εξέτασης.

4.7 Ασθένειες



Εικόνα 58: Diseases menu.

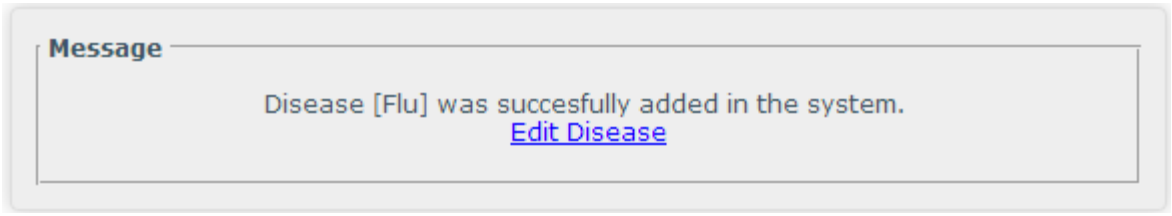
Η Εικόνα 58 δείχνει όλες τις διαδικασίες που μπορούμε να εκτελέσουμε από το μενού του **Diseases**. Θα ακολουθήσει λεπτομερής περιγραφή των διαδικασιών αυτών στα πιο κάτω υποκεφάλαια.

4.7.1 Δημιουργία Ασθένειας

Εικόνα 59: Φόρμα για την δημιουργία ασθένειας.

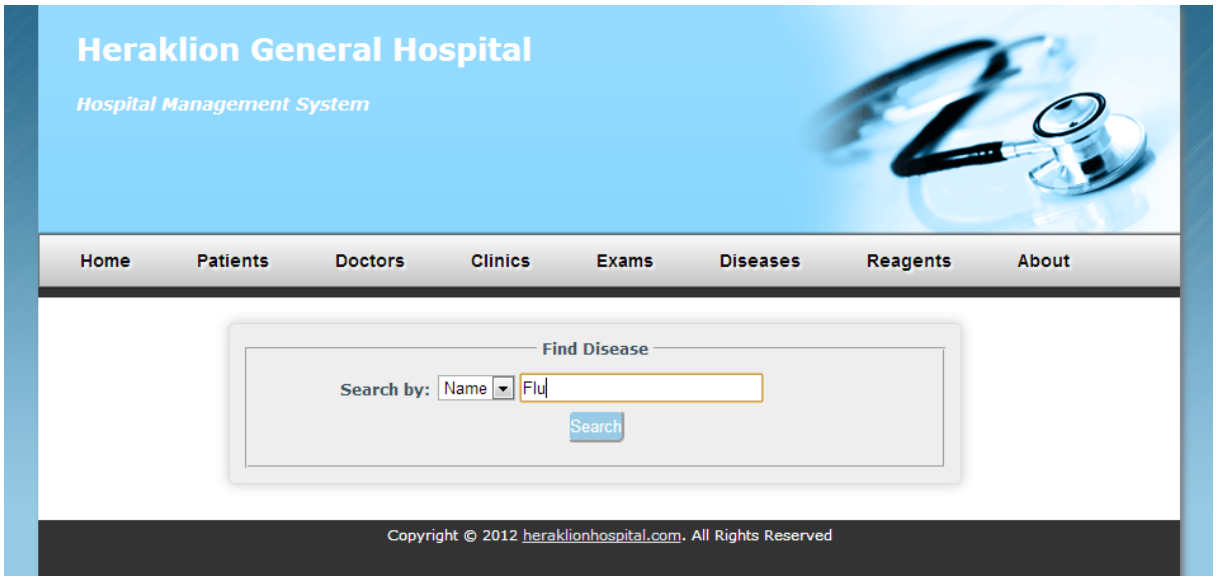
Η Εικόνα 59 δείχνει την φόρμα για την καταχώρηση μιας ασθένειας στο σύστημα. Όταν ο χρήστης συμπληρώσει το όνομα της ασθένειας και πατήσει το κουμπί **Submit**, τότε το σύστημα θα προσπαθήσει να καταχωρήσει την ασθένεια στο σύστημα. Σε περίπτωση

επιτυχίας το σύστημα θα ενημερώσει τον χρήστη με το πιο κάτω μήνυμα και επίσης, ένα link που οδηγεί στην ασθένεια που μόλις δημιουργήθηκε.



Εικόνα 60: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή γιατρού στο σύστημα.

4.7.2 Εύρεση Ασθένειας



Εικόνα 61: Φόρμα εύρεσης ασθένειας.

Η εφαρμογή επιτρέπει την εύρεση μιας καταχωρημένης ασθένειας στο σύστημα βάση του ονόματος της. Όταν ο χρήστης πατήσει το κουμπί Search, τότε το σύστημα ελέγχει κατά πόσο υπάρχει ασθένεια με αυτό όνομα και εμφανίζει τα αποτελέσματα σε ένα πίνακα όπως φαίνεται στην πιο κάτω εικόνα.

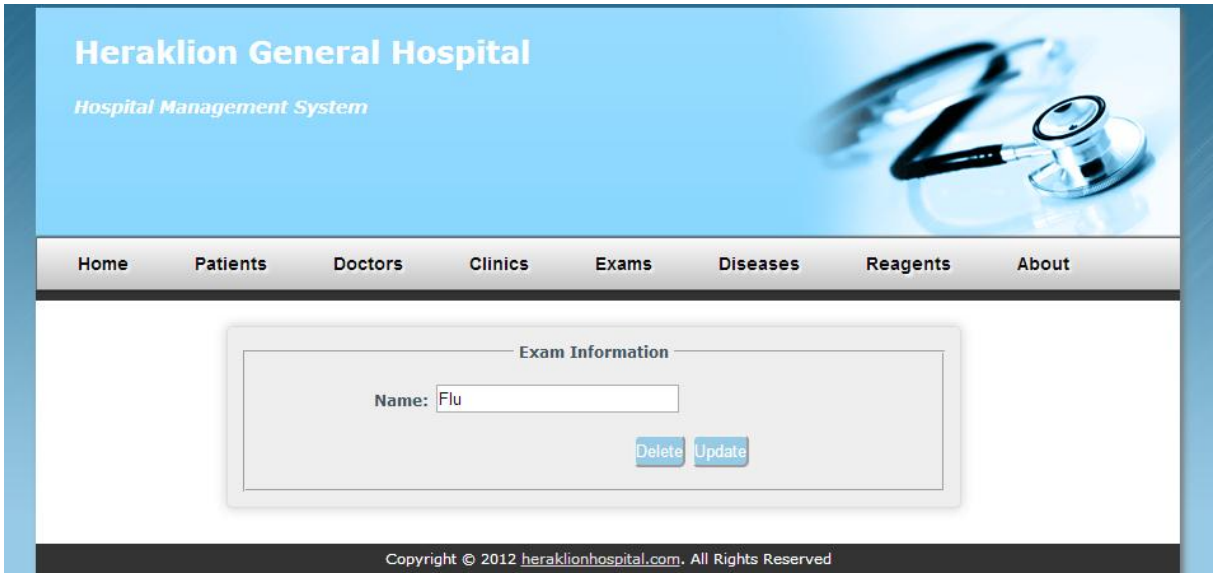
Εικόνα 62: Εμφάνιση αποτελεσμάτων κατά την εύρεση ασθένειας.

4.7.3 Προβολή όλων των Ασθενειών

Εικόνα 63: Σελίδα προβολής όλων των ασθενειών

Η πιο πάνω εικόνα δείχνει την σελίδα προβολής όλων των ασθενειών που είναι καταχωρημένοι στο σύστημα. Για κάθε ασθένεια προβάλλονται κάποιες βασικές πληροφορίες και επίσης τα αντίστοιχα links για την επεξεργασία ή την διαγραφή της ασθένειας από το σύστημα.

4.7.4 Επεξεργασία Ασθένειας

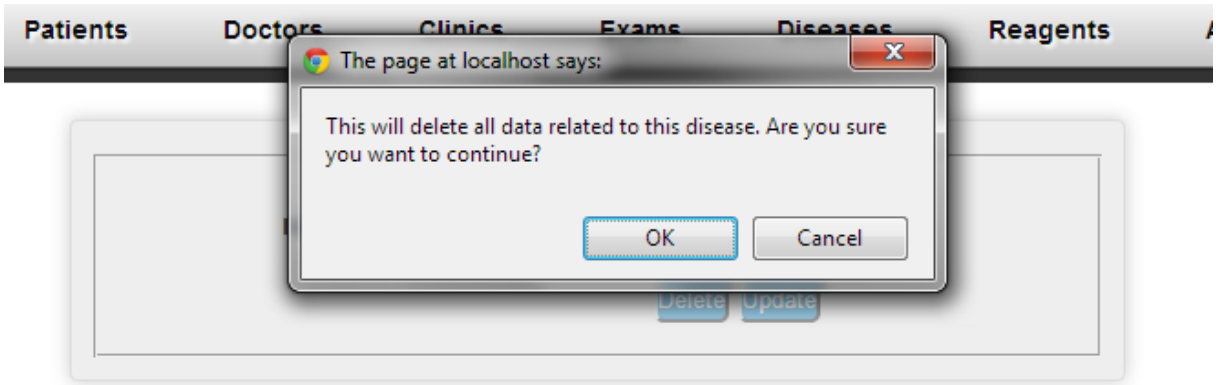


Εικόνα 64: Φόρμα επεξεργασίας ασθένειας.

Το σύστημα επιτρέπει στους χρήστες την αλλαγή ονομασίας μίας υπάρχουσας ασθένειας. Η αποθήκευση των αλλαγών επιτυγχάνεται με το πάτημα του κουμπιού **Update**. Επίσης, υπάρχει διαθέσιμο το κουμπί **Delete** για την διαγραφή της συγκεκριμένης ασθένειας από το σύστημα.

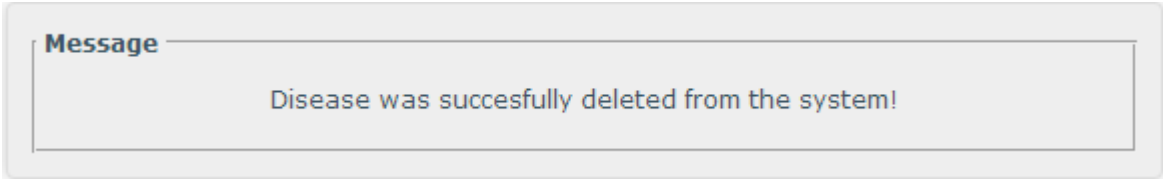
4.7.5 Διαγραφή Ασθένειας

Σε περίπτωση που ο χρήστης επιλέξει την διαγραφή μιας ασθένειας, τότε το σύστημα θα εμφανίσει ένα παράθυρο επιβεβαίωσης ρωτώντας τον χρήστη αν όντως θέλει να προχωρήσει στην διαγραφή όπως φαίνεται πιο κάτω.



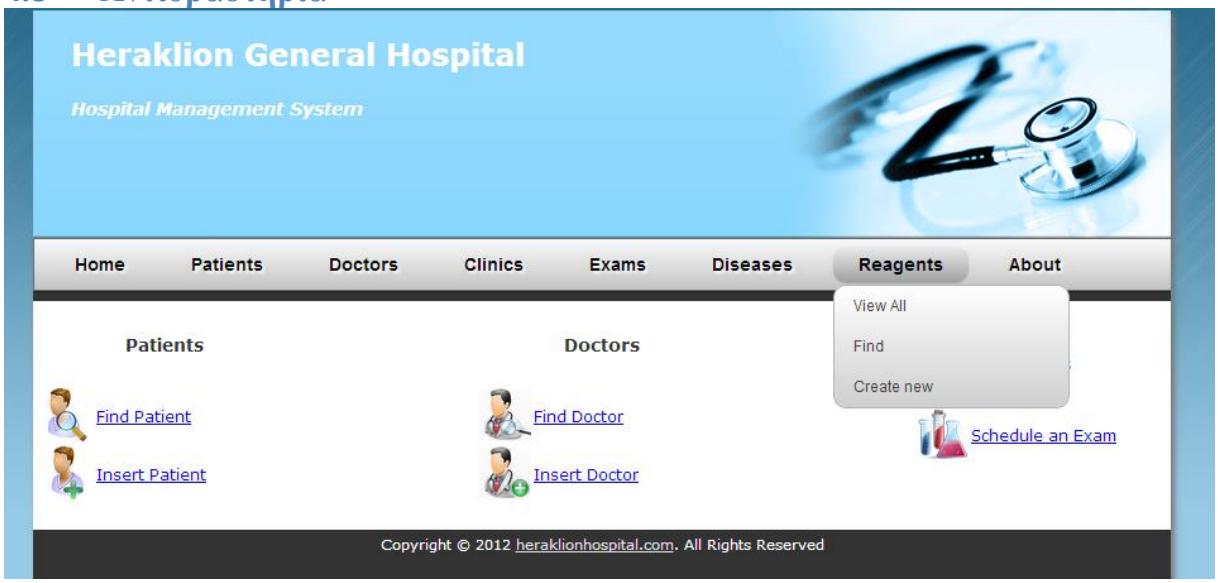
Εικόνα 65: Μήνυμα επιβεβαίωσης πριν την διαγραφή ασθένειας.

Πατώντας OK τότε το σύστημα θα ψάξει και θα προσπαθήσει να διαγράψει από την βάση την συγκεκριμένη ασθένεια. Σε περίπτωση επιτυχούς διαγραφής θα εμφανιστεί το πιο κάτω μήνυμα.



Εικόνα 66: Μήνυμα ενημέρωσης για επιτυχή διαγραφή ασθένειας.

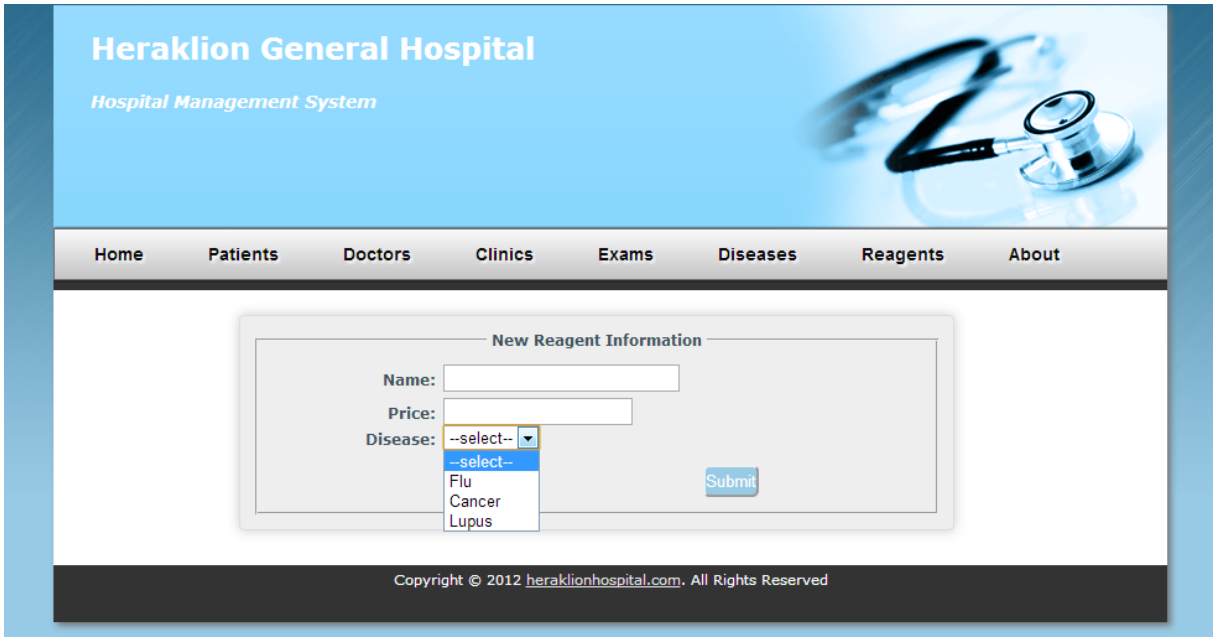
4.8 Αντιδραστήρια



Εικόνα 67: Reagents menu.

Η Εικόνα 67 δείχνει όλες τις διαδικασίες που μπορούμε να εκτελέσουμε από το μενού του **Reagents**. Θα ακολουθήσει λεπτομερής περιγραφή των διαδικασιών αυτών στα πιο κάτω υποκεφάλαια.

4.8.1 Δημιουργία Αντιδραστηρίου (Reagents->Create new)



Εικόνα 68: Φόρμα για την δημιουργία αντιδραστηρίου.

Η Εικόνα 68 δείχνει την φόρμα για την εισαγωγή ενός αντιδραστηρίου στο σύστημα. Για την δημιουργία ενός νέου αντιδραστηρίου ο χρήστης πρέπει να εισάγει το όνομα του, την τιμή και επίσης την ασθένεια την οποία ανιχνεύει το συγκεκριμένο αντιδραστήριο. Όταν ο χρήστης συμπληρώσει σωστά όλα τα απαιτούμενα πεδία και πατήσει το κουμπί **Submit**, τότε το σύστημα θα προσπαθήσει να εισάγει τα στοιχεία αυτά στην βάση δημιουργώντας ένα νέο αντιδραστήριο. Σε περίπτωση επιτυχίας το σύστημα θα ενημερώσει τον χρήστη με το πιο κάτω μήνυμα.

Εικόνα 69: Μήνυμα ενημέρωσης για επιτυχή εισαγωγή αντιδραστηρίου στο σύστημα.

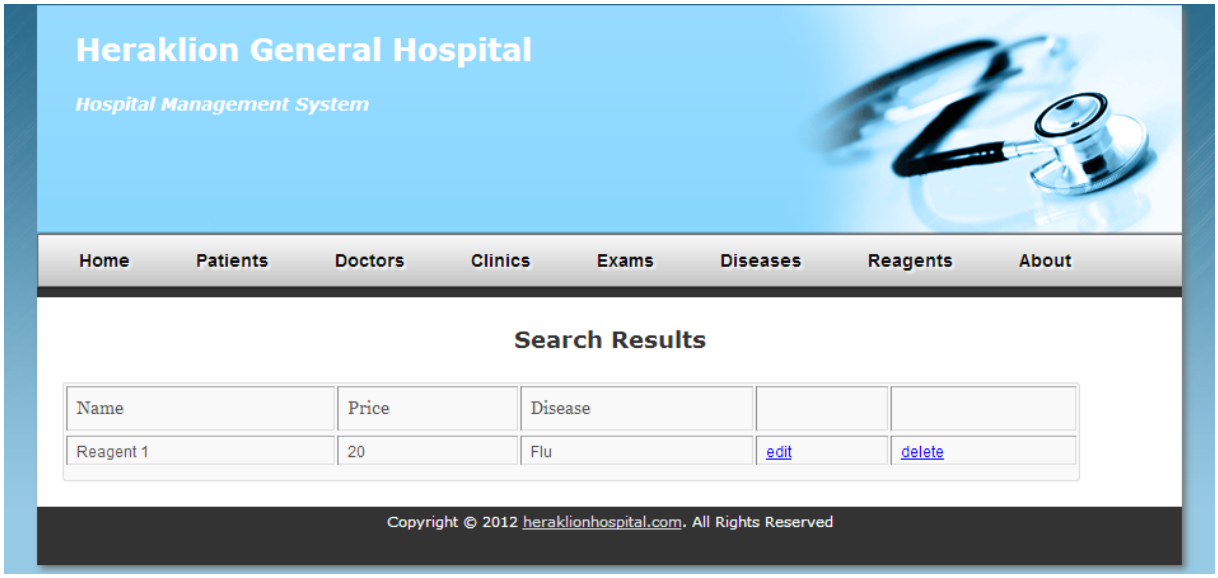
4.8.2 Εύρεση Αντιδραστηρίου(Reagent->Find)

Εικόνα 70: Φόρμα εύρεσης αντιδραστηρίου.

Η εφαρμογή επιτρέπει την εύρεση ενός καταχωρημένου αντιδραστηρίου στο σύστημα με χρησιμοποιώντας ένα από τα διαθέσιμα κριτήρια:

- Name
- Disease (την ασθένεια την οποία ανιχνεύει)

Όταν ο χρήστης πατήσει το κουμπί Search, τότε το σύστημα ελέγχει κατά πόσο υπάρχει αντιδραστήριο/α που πληρούν το συγκεκριμένο κριτήριο και εμφανίζει τα αποτελέσματα σε ένα πίνακα όπως φαίνεται στην πιο κάτω εικόνα.



Εικόνα 71: Εμφάνιση αποτελεσμάτων κατά την εύρεση αντιδραστηρίου.

4.8.3 Προβολή όλων των Αντιδραστηρίων (Reagents->View All)

Εικόνα 72: Σελίδα προβολής όλων των αντιδραστηρίων.

Η πιο πάνω εικόνα δείχνει την σελίδα προβολής όλων των αντιδραστηρίων που είναι καταχωρημένα στο σύστημα. Για κάθε αντιδραστήριο προβάλλονται κάποιες βασικές πληροφορίες και επίσης τα αντίστοιχα links για την επεξεργασία ή την διαγραφή του.

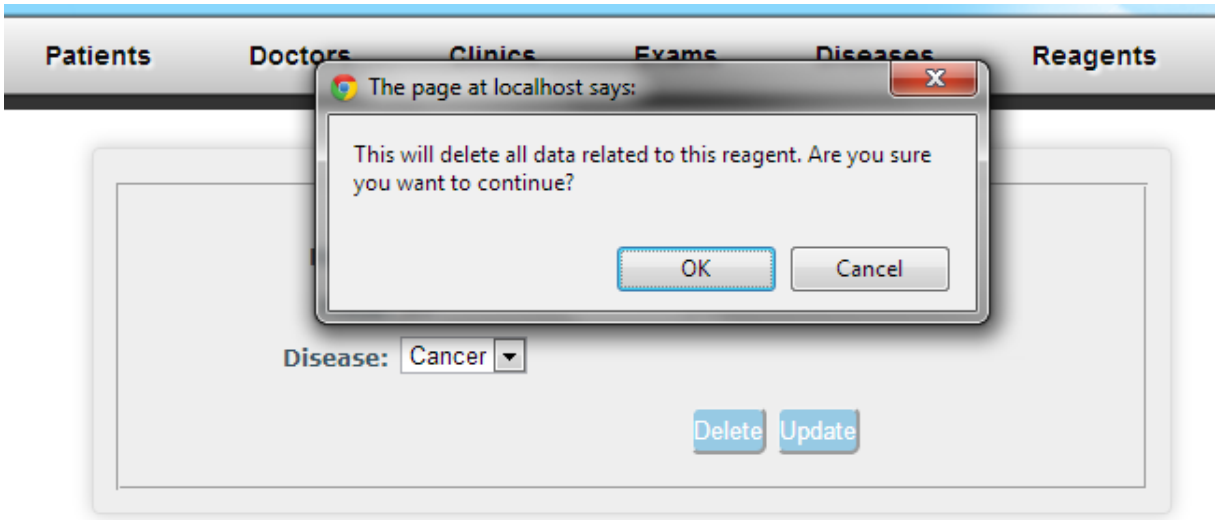
4.8.4 Επεξεργασία Αντιδραστηρίου

Εικόνα 73: Φόρμα επεξεργασίας αντιδραστηρίου.

Στην Εικόνα 73 βλέπουμε την σελίδα στην οποία μπορούμε να επεξεργαστούμε τα αντιδραστήρια που είναι ήδη καταχωρημένα στο σύστημα. Η αποθήκευση των αλλαγών στα στοιχεία ενός αντιδραστηρίου επιτυγχάνεται με το πάτημα του κουμπιού **Update**. Επίσης, υπάρχει διαθέσιμο το κουμπί **Delete** για την διαγραφή του συγκεκριμένου αντιδραστηρίου.

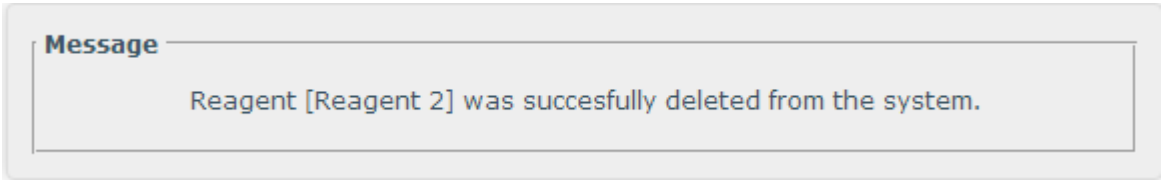
4.8.5 Διαγραφή Αντιδραστηρίου

Σε περίπτωση που ο χρήστης επιλέξει την διαγραφή ενός αντιδραστηρίου, τότε το σύστημα θα εμφανίσει ένα παράθυρο επιβεβαίωσης ρωτώντας τον χρήστη αν όντως θέλει να προχωρήσει στην διαγραφή όπως φαίνεται πιο κάτω. Το παράθυρο επιβεβαίωσης εμφανίζεται για να αποφεύγεται η καταλάθος διαγραφή μίας καταχώρησης.



Εικόνα 74: Μήνυμα επιβεβαίωσης πριν την διαγραφή αντιδραστηρίου

Πατώντας OK τότε το σύστημα θα ψάξει και θα προσπαθήσει να διαγράψει από την βάση το συγκεκριμένο αντιδραστήριο. Σε περίπτωση επιτυχούς διαγραφής θα εμφανιστεί το πιο κάτω μήνυμα.



Εικόνα 75: Μήνυμα ενημέρωσης για επιτυχή διαγραφή αντιδραστηρίου.

Κεφάλαιο 5: Αποτελέσματα

5.1 Συμπεράσματα

Στην παραπάνω εργασία περιγράφηκε μία βάση δεδομένων για την διαχείριση ενός νοσοκομείου καθώς και το σύστημα διαχείρισής της. Όπως έχει αναφερθεί, για την ανάπτυξη της βάσης δεδομένων και του συστήματος διαχείρισης χρησιμοποιήθηκαν εργαλεία και γλώσσες τελευταίας τεχνολογίας όπως το Adobe Dreamweaver CS6, HTML5, CSS3 και Javascript.

Κατά τον σχεδιασμό της βάσης, λήφθηκαν υπόψιν τόσο οι κανόνες καλού σχεδιασμού όσο και όλες οι πληροφοριακές ανάγκες της συγκεκριμένης βάσης δεδομένων. Δημιουργήθηκε το μοντέλο οντοτήτων σχέσεων, με βάση την περιγραφή των απαιτήσεων και αυτό μετατράπηκε σε σχεσιακό ακολουθώντας του αντιστοιχούν κανόνες. Περιγράφηκαν οι περιορισμοί ακεραιότητας και οι συναρτησιακές εξαρτήσεις ώστε το μοντέλο να μετατραπεί σε πρώτη, δεύτερη και τρίτη κανονική μορφή. Έτσι εξασφαλίστηκε η δημιουργία μίας αποδοτικής βάσης δεδομένων.

Κατά την εκτέλεση της εργασίας δημιουργήθηκε ένας ιστοχώρος που λειτουργεί ως σύστημα διαχείρισης της βάσης δεδομένων. Σε αυτό το σύστημα συμπεριλήφθηκαν λειτουργίες για εισαγωγή, διαγραφή, αλλαγή και εύρεση όλων των οντοτήτων και σχέσεων που υποστηρίζει η βάση. Σε κάθε περίπτωση έγινε προσπάθεια οι λειτουργίες να προσφέρονται με μία φιλική προς το χρήστη διεπαφή. Επιπλέον τα επερωτήματα της SQL δημιουργήθηκαν με κριτήριο την βελτιστοποίηση του χρόνου εκτέλεσης τους. Μέρος της εργασίας αποτελεί και ένα εκτενές εγχειρίδιο χρήσης που περιγράφει με λεπτομέρειες τον τρόπο χρήσης αυτού του συστήματος.

Η βάση δεδομένων και το σύστημα διαχείρισης της, είναι σχεδιασμένη με βάση την θεωρία που έχει προκύψει από τους αντίστοιχους ερευνητικούς τομείς και έτσι θα μπορούσε να χρησιμοποιηθεί ως πραγματικό σύστημα διαχείρισης ενός νοσοκομείου, με ορισμένες επεκτάσεις όπως αυτές περιγράφονται στην επόμενη ενότητα.

5.2 Μελλοντική Εργασία και Επεκτάσεις

Η εφαρμογή που υλοποιήθηκε, αν και ικανοποιεί όλες τις απαιτήσεις και στόχους που τέθηκαν στην αρχή της πτυχιακής δεν αποτελεί ένα ολοκληρωμένο πληροφοριακό σύστημα διαχείρισης ενός νοσοκομείου. Προφανώς ένα τέτοιο σύστημα είναι μεγαλύτερης πολυπλοκότητας και δεν θα μπορούσε να υλοποιηθεί στα πλαίσια μιας πτυχιακής εργασίας. Ωστόσο υπάρχουν διάφοροι τρόποι με τους οποίους η υφιστάμενη εφαρμογή θα μπορούσε να βελτιωθεί και να επεκταθεί όπως:

1. Όλα τα dropdown lists που εμφανίζονται στο σύστημα για επιλογή γιατρού, κλινικής κτλ μπορούν να αντικατασταθούν με την υλοποίηση ενός παραθύρου pop-up όπου ο χρήστης θα μπορούσε να περιηγηθεί στις διαθέσιμες επιλογές και να επιλέξει τον γιατρό ή κλινική που επιθυμεί. Αυτό γιατί σε περίπτωση που στο σύστημα καταχωρηθεί ένας μεγάλος αριθμός π.χ. γιατρών τότε δεν θα είναι καθόλου πρακτικά στην αναζήτηση και την επιλογή του γιατρού που επιθυμεί ο χρήστης.
2. Να δημιουργηθούν ομάδες χρηστών όπως διαχειριστές, γιατροί, διευθυντές κλινικών κτλ και επίσης ένας μηχανισμός login έτσι ώστε να υπάρχει ελεγχόμενη πρόσβαση στο σύστημα και επίσης, η κάθε ομάδα χρηστών να έχει πρόσβαση σε συγκεκριμένες λειτουργίες του συστήματος.
3. Να υλοποιηθεί η λειτουργία διαχείρισης των κρεβατιών μίας κλινικής και η ανάθεση τους σε κάποιο ασθενή.
4. Τέλος, το σύστημα θα μπορούσε επεκταθεί έτσι ώστε να υποστηρίζει και το κλείσιμο ραντεβού για την διεξαγωγή μιας εξέτασης.

Βιβλιογραφία

1] J. Ullman και J. Widom, First course in database systems, Prentice-Hall Inc., Simon & Schuster, 1997.

2] J. Simpson και E. Weiner , A New English Dictionary on Historical Principles (NED), London: Ofxord University Press, 1989.

3] E. F. Codd, «Derivability, Redundancy, and Consistency of Relations Stored in Large Data Banks,» IBM Research Report, 1969.

4] E. F. Codd, «A Relational Model of Data for Large Shared Data Banks,» σε *Communications of the ACM*, 1970.

5] P. Pin και S. Chen, «The Entity-Relationship Model: Toward a Unified View of Data,» σε *ACM Transactions on Database Systems*, 1976.

6] «ΣΤΑΤΙΚΗ VS ΔΥΝΑΜΙΚΗ ΙΣΤΟΣΕΛΙΔΑ,» [Ηλεκτρονικό]. Available: http://www.anaptyxis.net/services/pages/whattochoose.html.

7] «Στατική και δυναμική ιστοσελίδα, οι διαφορές και το κόστος,» [Ηλεκτρονικό]. Available: http://www.netrino.gr/reloaded/blog-post.php?bp_id=798.

8] «phpMyAdmin,» [Ηλεκτρονικό]. Available: http://en.wikipedia.org/wiki/PhpMyAdmin.

9] «MySQL Workbench,» [Ηλεκτρονικό]. Available: http://en.wikipedia.org/wiki/MySQL_Workbench.

10] «HTML5,» [Ηλεκτρονικό]. Available: http://en.wikipedia.org/wiki/HTML5.

11] «HTML5 Introduction,» [Ηλεκτρονικό]. Available: http://www.w3schools.com/html/html5_intro.asp.

«Cascading Style Sheets,» [Ηλεκτρονικό]. Available:

12] http://en.wikipedia.org/wiki/Cascading_Style_Sheets.

«JavaScript,» [Ηλεκτρονικό]. Available: http://en.wikipedia.org/wiki/JavaScript.

13]

«AJAX Introduction,» [Ηλεκτρονικό]. Available:

14] http://www.w3schools.com/ajax/ajax_intro.asp.

Παράρτημα

1. Βασικό template

Οι παρακάτω πίνακες περιέχουν τον κώδικα για την υλοποίηση του βασικού template του συστήματος. Το template αυτό χρησιμοποιήθηκε για την δημιουργία όλων των σελίδων του συστήματος. Αυτό επιτεύχθηκε δηλώνοντας editable sections.

```
<!doctype html>
<html lang="en">
<head>
        <meta charset="utf-8" />
        <title>Hospital Management System</title>
//Εισαγωγή των αρχείων CSS για καθορισμό του στυλ του site
<link rel="stylesheet" href="../styles.css" type="text/css" media="screen" />
<link rel="stylesheet" type="text/css" href="../table_styles.css" />
        <!--[if IE]><script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script><![endif]-->
<!-- TemplateBeginEditable name="head" -->
<!-- TemplateEndEditable -->
</head>
<body>
//Top banner του site
<div id="container"><!-- #container -->
        <header><!-- header -->
                <h1><a href="#">Heraklion General Hospital</a></h1>
                <h2>Hospital Management System</h2>
    <br><br><br><br><h3 align="right"><div id="dateTimeValue"></div></h3>
        </header><!-- end of header -->

//Δημιουργία του βασικού μενού της εφαρμογής
<nav><!-- top nav -->
<ul id="menu-bar">
    <li><a href="index.html">Home</a></li>
    <li><a href="#">Patients</a>
     <ul>
      <li><a href="viewPatients.php">View All</a></li>
      <li><a href="findPatientForm.html">Find</a></li>
      <li><a href="addPatientForm.php">Create new</a></li>
     </ul>
    </li>
    <li><a href="#">Doctors</a>
    <ul>
      <li><a href="viewDoctors.php">View All</a></li>
      <li><a href="findDoctorForm.html">Find</a></li>
      <li><a href="addDoctorForm.php">Create new</a></li>
    </ul>
    </li>
    <li><a href="#">Clinics</a>
    <ul>
      <li><a href="viewClinics.php">View All</a></li>
      <li><a href="findClinicForm.html">Find</a></li>
      <li><a href="addClinicForm.php">Create new</a></li>
    </ul>
    </li>
    <li><a href="#">Exams</a>
    <ul>
      <li><a href="viewExams.php">View All</a></li>
      <li><a href="findExam.html">Find</a></li>
      <li><a href="addExamForm.php">Create New</a></li>
      <li><a href="scheduleExamForm.php">Schedule Exam</a></li>
      <li><a href="viewScheduledExams.php">View Scheduled Exams</a></li>
```

```
          <li><a href="viewExamsHistory.php">View Past Exams</a></li>

        </ul>
          <li><a href="#">Diseases</a>
        <ul>
          <li><a href="viewDiseases.php">View All</a></li>
          <li><a href="findDiseaseForm.html">Find</a></li>
          <li><a href="addDiseaseForm.html">Create new</a></li>
        </ul>

          <li><a href="#">Reagents</a>
        <ul>
          <li><a href="viewReagents.php">View All</a></li>
          <li><a href="findReagent.html">Find</a></li>
          <li><a href="addReagentForm.php">Create new</a></li>
        </ul>
        </li>

        <li><a href="about.html">About</a></li>

       </ul>
            </nav><!-- end of top nav -->
//Κεντρικό τμήμα του site (content)
<section id="main"><!-- #main content  -->
// Δήλωση μίας περιοχής που μπορεί να τροποποιηθεί (editable). Εδώ θα εμφανίζεται το περιεχόμενο της κάθε σελίδας του συστήματος.
   <!-- TemplateBeginEditable name="main" -->
    main
            <!-- TemplateEndEditable -->
    </section><!-- end of #main content -->
            <footer>
                    <section id="footer-area">
                       <p align="center"> Copyright &copy; 2012 <a href="#">heraklionhospital.com</a>. All Rights Reserved</p>
                 </section><!-- end of footer-area -->
            </footer>

</div><!-- #container -->
// Δήλωση ακόμη μίας editable περιοχής για την προαιρετική προσθήκη JavasScript στις σελίδες.
<!-- TemplateBeginEditable name="JavaScript Code" --><!-- TemplateEndEditable -->

</body>
</html>
```

2. Φόρμα δημιουργίας νέου ασθενή (front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
 <div id="form_data">
  //Δήλωση μιας φόρμας για την εισαγωγή των στοιχείων του ασθενή και επίσης το αρχείο για την επεξεργασία εισόδου όταν γίνει submit
  <form action="addPatient_newRecord.php" method="post">
   <fieldset>
            <legend align="center" color="black" font="14px"> <b>New Patient Information</b> </legend><br>

            <div>
             <label>Full Name:</label> <input type="text" name="Name" required/>
   </div>

//Σε αυτό το πεδίο φαίνεται η χρήση νέων features της HTML5 όπως το required και το pattern. Επίσης εδώ γινεται και η χρήση javascript για την κλήση της μεθόδου που ελέγχει την μοναδικότητα της τιμής που δίνει ο χρήστης με την βάση. Η κλήση της συνάρτησης γίνεται κάθε φορά που ο χρήστης εισάγει ένα χαρακτήρα στο πεδίο εισόδου.
    <div>
             <label>Political ID:</label> <input title="8 Digits Number" onkeyup="checkPoliticalID(this.value)" name="idPatient" pattern="[0-9]{8}" autocomplete="off" required/> <caption><i>(8 digits number)</i></caption> <span id="txtHint"></span>
            </div>

            <div>
             <label>Insurance:</label> <input type="text" name="Insurance" required/>
    </div>

//Εδώ φαίνεται η υλοποίηση ενός dropdown list.
    <div>
             <label>Sex:</label>
     <select name="Sex">
           <option value="male">Male</option>
           <option value="female">Female</option>
     </select>
            </div>

//Εδώ φαίνεται η χρήση ενός νεού input type της HTML5, του date. Το date διαθέτει επίσης attibutes για τον έλεγχο της επιλογής του χρήστη, max και min.
    <div>
             <label>Birth Date:</label> <input type="date" name="dateBirth" max="<?php echo date('Y-m-d', strtotime(date('Y/m/d'))); ?>"  required/>
            </div>

    <div>
             <label>Address:</label> <input type="text" name="Street" required/>
            </div>

//Σε αυτό το πεδίο χρησιμοποιήθηκε πάλι το attribute pattern για τον έλεγχο της εισόδου χρήστη.
     <div>
             <label>Postal Code:</label> <input id="postcode" title="5 digits number" type="text" name="PostalCode" pattern="[1-9][0-9]{4}" required/> <caption><i>(5 digits number e.g. 30200)</i></caption>
            </div>

    <div>
             <label>City:</label> <input type="text" name="City" required/>
            </div>

    <div>
             <label>Hospitalized On:</label> <input type="date" name="dateHospitalizationStart"  max="<?php echo date('Y-m-d', strtotime(date('Y/m/d'))); ?>" />
            </div>
```

```
//Εδώ είναι ορατή η χρήση της PHP μεθόδου “fill_dropdown_list” που δημιουργήθηκε έτσι ώστε να δημιουργούνται δυναμικά dropdown lists με τιμές απο την βάση δεδομένων. Στο συγκεκριμένο πεδίο ανακτούνται και εμφανίζονται όλες τις καταχωρημένες κλινικές στο σύστημα.
    <div>
            <label>Clinic:</label>
   <?php
            $query = "SELECT idClinic, Name FROM clinics ORDER BY Name";
           fill_dropdown_list("idClinic", "Name", "idClinic", $query, 1);
   ?>
           </div>

//Το κουμπί submit που θα στείλει τα δεδομένα εισόδου για επεξεργασία.
    <p class="buttons">
                    <input type="submit">
    </p>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

3. Υλοποίηση μηχανισμού για Real-Time Validation (AJAX/JavaScript)

```
<!-- InstanceBeginEditable name="JavaScript Code" -->
<script type="text/javascript">

function checkPoliticalID(str){
            // Αν ο αριθμός ταυτότητας δεν αποτελείται από 8 ψηφία τότε δείξε το μήνυμα “non valid” με κόκκινα γράμματα και βγες απο την μέθοδο.
            if (str.length != 8)
              {
                    if(str.length ==0){
                            document.getElementById("txtHint").innerHTML = "";
                    }else{
                            document.getElementById("txtHint").innerHTML = "non valid";
                            document.getElementById("txtHint").style.color = "red";
                    }

                    return;
              }
            if (window.XMLHttpRequest)
              {// code for IE7+, Firefox, Chrome, Opera, Safari
              xmlhttp=new XMLHttpRequest();
              }
            else
              {// code for IE6, IE5
              xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
              }
            xmlhttp.onreadystatechange=function()
              {
              if (xmlhttp.readyState==4 && xmlhttp.status==200)
              {
            // Αν ο αριθμός ταυτότητας ήδη υπάρχει τότε δείξε το μήνυμα “already in use” με κόκκινα γράμματα
        if(xmlhttp.responseText){
                                        document.getElementById("txtHint").innerHTML = "already in use";
                                        document.getElementById("txtHint").style.color = "red";
                                }else{
            // Διαφορετικά δείξε το μήνυμα “valid”  με πράσινα γράμματα
     document.getElementById("txtHint").innerHTML = "valid";
                                        document.getElementById("txtHint").style.color = "green";
                                }
                    }
            }
      //Κλήση της μεθόδου realTimeVal.php για έλεγχο της εισόδου με την βάση
```

```
        xmlhttp.open("GET", "realTimeVal.php?val="+str,true);
        xmlhttp.send();
}
</script>
<!-- InstanceEndEditable -->
```

4. PHP αρχείο realTimeVal.php

```
<?php

//get the input value parameter from URL
$input_val=$_GET["val"];

// Σύνδεση στον mysql server
$con = mysql_connect('localhost','root', '');

// Έλεγχος επιτυχής σύνδεσης
if (!$con)
{
        die('Could not connect: ' . mysql_error());
}

// Επιλογή βάσης δεδομένων
mysql_select_db("hospital", $con);

// Query που επιστρέφει αν υπάρχει ασθενής με τον ίδιο αριθμό ταυτότητας.
$query = "SELECT 1 AS res FROM patients WHERE idPatient=".$input_val;

// Εκτέλεση query
$dbResult = mysql_query($query, $con);

$row = mysql_fetch_array($dbResult);
// Επιστροφή αποτελέσματος
echo $row['res'];
mysql_close($con);
?>
```

5. Δημιουργία νέου ασθενή (back-end)

```
<section id="main"><!-- #main content  -->
        <!-- InstanceBeginEditable name="main" -->
        <div id="form_data">
                <form>
        <fieldset>
                                        <legend align="left" color="black"
font="14px"> <b>Message</b> </legend><br>
//PHP κώδικας για την επεξεργασία ειδόδου
<?php
//Προσπάθεια για σύνδεση στον mysql server
        $con = mysql_connect('localhost','root', '');
    if (!$con)
    {
          echo '<p>Could not connect: ' . mysql_error() . '</p>';
        }
//Επιλογή της βάσης δεδομένων του συστήματος hospital
     mysql_select_db("hospital", $con);

//Ανάκτηση και αποθήκευση της εισόδου του χρήστη σε μεταβλητές
        $name = $_POST['Name'];
                $idPatient = $_POST['idPatient'];
                $ins = $_POST['Insurance'];
                $sex = $_POST['Sex'];
                $dateBirth = $_POST['dateBirth'];
                $street = $_POST['Street'];
                $pc = $_POST['PostalCode'];
```

```
                        $city = $_POST['City'];
                        $idClinic = $_POST['idClinic'];
                        $dhs = $_POST['dateHospitalizationStart'];

//Δυο διαφορετικά queries για την καταχώρηση του ασθενή στην βάση ανάλογα με το αν θα νοσηλευτεί ή όχι.
if($idClinic == '' || $dhs == ''){
                                $sql = "INSERT INTO patients (idPatient, Name, Insurance, Sex, dateBirth, Street,
PostalCode, City, idClinic, dateHospitalizationStart) VALUES ($idPatient, '$name', '$ins', '$sex', '$dateBirth', '$street', '$pc',
'$city', NULL, NULL)";

}else{
                                $sql = "INSERT INTO patients (idPatient, Name, Insurance, Sex, dateBirth, Street,
PostalCode, City, idClinic, dateHospitalizationStart) VALUES ($idPatient, '$name', '$ins', '$sex', '$dateBirth', '$street', '$pc',
'$city', '$idClinic', '$dhs')";
}

//Εκτέλεση του query για εισαγωγή του ασθενή με την εντολή mysql_query. Έαν αποτύχει τότε εμφανίζεται το μήνυμα
λάθους στον χρήστη.
                        if (!mysql_query($sql,$con))
                        {
                                echo "<p align='center'>Error: ".mysql_error()."!<br>Please try again.</p>";
                        }else{

//Διαφορετικά ο χρήστης ενημερώνεται για την επιτυχή εισαγωγή του ασθενή στο σύστημα.
        echo "<p align='center'>Patient [$name] was succesfully added in the system!
                                                <br><a
// Link που οδηγεί στον δημιουργημένο ασθενή.
href='editPatientForm.php?patient_id=$idPatient'>Edit Patient</a></p>";
                        }

//Διακοπή της σύνδεσης με τον mysql server.
      mysql_close($con);
        ?>
                        </fieldset>
                        </form>
  </div>
        <!-- InstanceEndEditable -->
  </section><!-- end of #main content -->
```

6. Φόρμα επεξεργασίας ασθενή (front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->

// Κώδικας PHP για την ανάκτηση των στοιχείων του ασθενή από την βάση δεδομένων.
<?php
        $patient_id=$_GET["patient_id"];

        $con = mysql_connect('localhost','root', '');
    if (!$con)
    {
                        die('Could not connect: ' . mysql_error());
         }

        mysql_select_db("hospital", $con);

$query = "SELECT * FROM patients
        WHERE idPatient=".$patient_id;

 $result = mysql_query($query, $con);

 $row = mysql_fetch_array($result);

// Αρχικοποίηση PHP μεταβλητών για την εμφάνιση των στοιχείων του ασθενή.
 $id = $row['idPatient'];
 $Name = $row['Name'];
```

```
 $Insurance = $row['Insurance'];
 $Sex = $row['Sex'];
 $dateBirth = $row['dateBirth'];
 $Street = $row['Street'];
 $PostalCode = $row['PostalCode'];
 $City = $row['City'];
 $idClinic = $row['idClinic'];
 $dateHospitalizationStart = $row['dateHospitalizationStart'];
?>

<div id="form_data">
// Δήλωση νέας φόρμας και το php αρχείο για την επεξεργασία της εισόδου
            <form action="editPatient.php" method="post">
              <fieldset>
                <legend align="center" color="black" font="14px"> <b>Patient Information</b> </legend><br>

// Τα controls της φόρμας είναι τα ίδια με την φόρμα δημιουργίας ασθενή με την μόνη διαφορά ότι αρχικοποιούνται με τα στοιχεία του ασθενή όπως αυτά ανακτήθηκαν από την βάση.

<div>
            <label>Full Name:</label> <input type="text" name="Name" value="<?php echo $Name ?>" required/>
</div>

<div>
            <label>Political ID:</label> <input name="idPatient" value="<?php echo $id ?>" readonly="readonly"
required/><caption><i>read-only</i></caption>
                                                    </div>

<div>
            <label>Insurance No:</label> <input name="Insurance" value="<?php echo $Insurance ?>" required/>
</div>

<div>
            <label>Sex:</label> <select id="Sex" name="Sex" >
          <option value="male">Male</option>
          <option value="female">Female</option>
        </select>
</div>

<div>
    <label>Birth Date:</label> <input type="date" name="dateBirth" max="<?php echo date('Y-m-d',
strtotime(date('Y/m/d'))); ?>" value="<?php echo $dateBirth ?>" required/>
</div>

<div>
            <label>Address:</label> <input type="text" name="Street" value="<?php echo $Street ?>" required/>
</div>

<div>
            <label>Postal Code:</label> <input id="postcode" title="5 digits number" type="text" name="PostalCode"
pattern="[1-9][0-9]{4}" value="<?php echo $PostalCode ?>" required/> <caption><i>(5 digits number e.g.
30200)</i></caption>
</div>

<div>
            <label>City:</label> <input type="text" name="City" value="<?php echo $City ?>" required/>
</div>

<div>
            <label>Hospitalized On:</label> <input type="date" name="dateHospitalizationStart"  max="<?php echo date('Y-
m-d', strtotime(date('Y/m/d'))); ?>" value="<?php echo $dateHospitalizationStart ?>" />
</div>

<div>
            <label>Clinic:</label>
```

```
<?php
                        $query = "SELECT idClinic, Name FROM clinics ORDER BY Name";
            fill_dropdown_list("idClinic", "Name", "idClinic", $query, 1);
?>
</div>

<div>
// Δήλωση πίνακα για την εμφάνιση των κουμπιών Delete και Submit
  <table class="buttons" cellspacing="5">
    <tr>
      <td><input id="delete" name="action" value="Delete" type="submit" onClick="return confirm(
 'This will delete all data related to this patient. Are you sure you want to continue?');"/></td>
       <td><input id="update" name="action" value="Update" type="submit"/></td>
    </tr>
</table>
</div>
</fieldset>

// Δημιουργία των links που αφορούν τον ασθενή
<div>
            <label>Related Links:</label>
            <table cellspacing="5">
//Link για την προβολή του ιστορικού ασθενείας του ασθενή
<tr>
      <td><a class="rellinks" href="viewPatientHistory.php?patient_id=<?php echo $id ?>&report_type=1&name=<?php
echo $Name ?>">View Patient's Record</a>
</tr>

//Link για την προβολή του ιστορικού εξετάσεων του ασθενή
<tr>
      <td><a class="rellinks" href="viewPatientExamHistory.php?patient_id=<?php echo $id ?>&name=<?php echo $Name
?>">View Patient's Exam History</a>
</tr>

//Link για την προβολή του ιστορικού νοσηλείας του ασθενή
<tr>
    <td><a class="rellinks" href="viewPatientHistory.php?patient_id=<?php echo $id ?>&report_type=2&name=<?php echo
$Name ?>">View Patient's Hospitalization History</a></td>
</tr>
</table>
</div>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

7. Φόρμα προβολής ιστορικού του ασθενή

```
<section id="main"><!-- #main content  -->
            <!-- InstanceBeginEditable name="main" -->

//PHP κώδικας για την ανάκτηση και προβολή του ιστορικού ενός ασθενή.
<?php
 // Αποθήκευση των παραμέτρων
            $patient_id=$_GET["patient_id"];
    $report_type=$_GET["report_type"];
            $patient_name = $_GET["name"];

// Σύνδεση με τον mysql server
            $con = mysql_connect('localhost','root', '');

    if (!$con)
                {
                        die('Could not connect: ' . mysql_error());
                }
// Επιλογή βάσης δεδομένων
```

```
    mysql_select_db("hospital", $con);

// Εάν report_type = 1 τότε εκτελείται query για την ανάκτηση και προβολή του ιστορικού ασθενείας του ασθενή.
if($report_type == 1){
        $query = "SELECT d.Name, r.dateDiagnosis
                        FROM patients p
                                INNER JOIN records r ON r.idPatient = p.idPatient
                                INNER JOIN diseases d ON d.idDisease = r.idDisease
                         WHERE p.idPatient = $patient_id
                                ORDER BY r.dateDiagnosis DESC";

        $result = mysql_query($query, $con);

                                echo "<a class=\"rellinks\" href=\"editPatientForm.php?patient_id=$patient_id\"><< Back to
Patient</a>";

                                echo "<h2 align=\"center\">Patient's [$patient_name] Diseases Record</h2>";

                                //<tr><th colspan=\"2\"><b>PATIENT'S [$patient_id] DISEASES RECORD</b></th></tr>

                                echo "<table  border=\"1\" id=\"table-3\">";
                                echo "<thead><tr><th>Diagnosed Disease</th><th>Date</th></tr></thead>";
// Επεξεργασία και προβολή των αποτελεσμάτων
                                while($row = mysql_fetch_array($result))
                                {
                                        $Name = $row['Name'];
                                        $Date = $row['dateDiagnosis'];

                                        echo "<tr><td>$Name</td><td>$Date</td></tr>";
                                }
                                echo "</table>";


// Διαφορετικά εκτελείται query για την ανάκτηση και προβολή του ιστορικού νοσηλείας του ασθενή.

}else{
        $query = "SELECT c.Name, r.dateStart, r.dateEnd
                                FROM patients p
                                INNER JOIN cp_past r ON r.idPatient = p.idPatient
                                INNER JOIN clinics c ON c.idClinic = r.idClinic
                                WHERE p.idPatient = $patient_id
                                ORDER BY r.dateStart DESC";

// Εκτέλεση του query
        $result = mysql_query($query, $con);

// Σύνδεσμος για επιστροφή στον ασθενή
        echo "<a class=\"rellinks\" href=\"editPatientForm.php?patient_id=$patient_id\"><< Back to Patient</a>";

                                echo "<h2 align=\"center\">Patient's [$patient_name] Hospitalization History</h2>";

                                echo "<table  border=\"1\" id=\"table-3\">";
                                echo "<thead><tr><th>Clinic</th><th>From</th><th>To</th></tr></thead>";

// Επεξεργασία και προβολή των αποτελεσμάτων
        while($row = mysql_fetch_array($result))
        {
                $Name = $row['Name'];
                $SDate = $row['dateStart'];
                $EDate = $row['dateEnd'];

                echo "<tr><td>$Name</td><td>$SDate</td><td>$EDate</td></tr>";
    }
                        echo "</table>";
                }
```

```
                mysql_close($con);
        ?>
        <!-- InstanceEndEditable -->
    </section><!-- end of #main content -->
```

8. Φόρμα προβολής ιστορικού εξετάσεων του ασθενή

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
//PHP κώδικας για την ανάκτηση και προβολή του ιστορικού εξετάσεων ενός ασθενή.
<?php
   // Αποθήκευση των παραμέτρων
        $patient_id=$_GET["patient_id"];
        $patient_name = $_GET["name"];

// Σύνδεση με τον mysql server
    $con = mysql_connect('localhost','root', '');
     if (!$con)
                {
                        die('Could not connect: ' . mysql_error());
                }

// Επιλογή βάσης δεδομένων
     mysql_select_db("hospital", $con);

// Εκτέλεση query για την ανάκτηση του ιστορικού εξετάσεων του ασθενή
        $query = "SELECT * FROM exams_history WHERE  idPatient = $patient_id";

// Εκτέλεση του query
     $result = mysql_query($query, $con);

                echo "<a class=\"rellinks\" href=\"editPatientForm.php?patient_id=$patient_id\"><< Back to
Patient</a>";

                echo "<h2 align=\"center\">Patient's [$patient_name] Exams Record</h2>";

// Δημιουργία πίνακα για εμφάνιση των αποτελεσμάτων
                echo "<table  border=\"1\" id=\"table-3\">";
                echo "<thead><tr><th>Exam</th><th>Ordering
Doctor</th><th>Result</th><th>Date</th></tr></thead>";

// Επεξεργασία και προβολή των αποτελεσμάτων
                        while($row = mysql_fetch_array($result))
                        {
                                $dName = $row['dName'];
                                $eName = $row['eName'];
                                $res = $row['Result'];
                                $date = $row['resultDate'];

                                echo "<tr><td>$dName</td><td>$eName</td><td>$res</td><td>$date</td></tr>";
                        }
                        echo "</table>";

                mysql_close($con);
        ?>
        <!-- InstanceEndEditable -->
    </section><!-- end of #main content -->
```

9. Επεξεργασία ασθενή (back-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
```

```
<form>
   <fieldset>
            <legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

<?php

// Σύνδεση με τον mysql server
            $con = mysqli_connect('localhost','root', '');

            if (!$con)
            {
                        echo '<p>Could not connect: ' . mysqli_error() . '</p>';
            }

// Επιλογή βάσης δεδομένων
     mysqli_select_db($con, "hospital");

// Ανάκτηση της μεθόδου HTTP με την οποία κλήθηκε το php αρχείο
            $action = isset($_POST['action']) ? $_POST['action'] : $_GET['action'];

            $patient_id = isset($_POST['idPatient']) ? $_POST['idPatient'] :  $_GET['idPatient'];

// Block για διαγραφή ασθενή
            if($action == 'Delete'){

// Θέτουμε autocommit = false έτσι ώστε να γίνουν commit οι αλλαγές στην βάση μόνο όταν εκτελεστούν με επιτυχία όλα τα queries (υλοποίηση transaction)
            mysqli_autocommit($con, false);

            $flag = true;
// Query για διαγραφή του ιστορικού ασθενείας
$query1 = "DELETE FROM records WHERE idPatient=$patient_id";
// Query για διαγραφή του ιστορικού νοσηλείας
$query2 = "DELETE FROM cp_past WHERE idPatient=$patient_id";
// Query για διαγραφή προγραμματισμένων εξετάσεων
$query3 = "DELETE FROM pde WHERE idPatient=$patient_id";
// Query για διαγραφή ιστορικού εξετάσεων
$query5 = "DELETE FROM exams_history WHERE idPatient=$patient_id";
// Query για διαγραφή του ιστορικού ασθενείας
$query4 = "DELETE FROM patients WHERE idPatient=$patient_id";

// Εκτέλεση όλων των πιο πάνω queries και έλεγχος του αποτελέσματος.
                        $result = mysqli_query($con, $query1);

                        if(!$result){
                                    $flag = false;
                                    echo "<p align='center'>Error: ". mysqli_error($con)."!<br>Please try again.</p>";
                                }

                        $result = mysqli_query($con, $query2);

                        if(!$result){
                                    $flag = false;
                                    echo "<p align='center'>Error: ". mysqli_error($con)."!<br>Please try again.</p>";
                        }

                        $result = mysqli_query($con, $query3);

                        if(!$result){
                                    $flag = false;
                                    echo "<p align='center'>Error: ". mysqli_error($con)."!<br>Please try again.</p>";
                        }

        $result = mysqli_query($con, $query5);

                        if(!$result){
```

```
                                        $flag = false;
                                        echo "<p align='center'>Error: ". mysqli_error($con)."!<br>Please try again.</p>";
                        }


                        $result = mysqli_query($con, $query4);

                        if(!$result){
                                        $flag = false;
                                        echo "<p align='center'>Error: ". mysqli_error($con)."!<br>Please try again.</p>";
                        }

// Εάν όλα τα queries εκτελέστηκαν με επιτυχία τότε γίνονται commit τις αλλαγές.
                        if($flag){
                                mysqli_commit($con);
                                echo "<p align='center'>Patient was succesfully deleted from the system!</p>";

// Διαφορετικά εκτελείται rollback.
}else{
                                mysqli_rollback($con);
                                echo "<p align='center'>Error: One of the queries failed!<br>Please try again.<br><a
href='editPatientForm.php?patient_id=$patient_id'>Back to Patient</a></p>";
                                }

// 2. Ανανέωση των στοιχείων του ασθενή.
}else if($action == 'Update'){

// Αποθήκευση των δεδομένων εισόδου
                        $name = $_POST['Name'];
                        $ins = $_POST['Insurance'];
                        $sex = $_POST['Sex'];
                        $dateBirth = $_POST['dateBirth'];
                        $street = $_POST['Street'];
                        $pc = $_POST['PostalCode'];
                        $city = $_POST['City'];
                        $idClinic = $_POST['idClinic'];
                        $dhs = $_POST['dateHospitalizationStart'];

                        if($idClinic == '' || $dhs == ''){
                                        $sql = "UPDATE patients
                                                        SET Name = '$name', Insurance = '$ins', Sex = '$sex', dateBirth =
'$dateBirth', Street = '$street',
                                                                PostalCode = '$pc', City = '$city', idClinic = NULL,
dateHospitalizationStart = NULL
                                                        WHERE idPatient = ".$patient_id;
                        }else{
                                        $sql = "UPDATE patients
                                                        SET Name = '$name', Insurance = '$ins', Sex = '$sex', dateBirth =
'$dateBirth', Street = '$street',
                                                                PostalCode = '$pc', City = '$city', idClinic = $idClinic,
dateHospitalizationStart = '$dhs'
                                                        WHERE idPatient = ".$patient_id;
                                }

                                if (!mysqli_query($con, $sql))
                                {
                                        echo "<p align='center'>Error: ".mysqli_error()."!<br>Please try again.</p>";
                                }else{
                                        echo "<p align='center'>Patient [$name] details were succesfully updated!
                                                <br><a href='editPatientForm.php?patient_id=$patient_id'>Back to
Patient</a></p>";
                                }
// 2. Λήξη νοσηλείας ασθενή.
}else if($action == 'Checkout'){

// Αποθήκευση της τρέχουσας ημερομηνίας
```

```
$curr_date = date('Y-m-d', strtotime(date('Y/m/d')));

// Έναρξη ενός sql transaction
mysqli_autocommit($con, false);

$flag = true;

// Query για εισαγωγή μιας νέας εγγραφής στο ιστορικό νοσηλείας του ασθενή.
$query1 = "INSERT INTO cp_past (idClinic, idPatient, dateStart, dateEnd)
    SELECT idClinic,idPatient,dateHospitalizationStart,        '$curr_date' FROM patients
    WHERE idPatient=$patient_id";

// Εκτέλεση του query.
$result = mysqli_query($con, $query1);

if(!$result){
            $flag = false;
}

// Query για ανανέωση των πεδίων idClinic και dateHospitalizationStart του ασθενή
$query2 = "UPDATE patients
                        SET idClinic = NULL, dateHospitalizationStart = NULL
                        WHERE idPatient=$patient_id";

// Εκτέλεση του query.
$result = mysqli_query($con, $query2);

if(!$result){
                        $flag = false;
}

// Αν και τα 2 queries εκτελέστηκαν με επιτυχία τότε οι αλλαγές γίνονται commit.
if($flag){
                        mysqli_commit($con);
                        echo "<p align='center'>Patient was succesfully checked out from clinic!
                        <br><a href='editPatientForm.php?patient_id=$patient_id'>Back to Patient</a></p>";

// Αλλιώς οι αλλαγές γίνονται rollback
}else{
                        mysqli_rollback($con);
                        echo "<p align='center'>Error: One of the queries failed!<br>Please try again.<br><a
href='editPatientForm.php?patient_id=$patient_id'>Back to Patient</a></p>";
}
}
            mysqli_close($con);
?>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

10. Φόρμα εύρεσης ασθενή (front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
//Δήλωση νέας φόρμας και του php αρχείου για την προβολή των αποτελεσμάτων
            <form action="viewPatients.php" method="post">
<fieldset>
            <legend align="center" color="black" font="14px"> <b>Find Patient</b> </legend><br>

<div>
  <label>Search by:</label>
```

```
//Δήλωση drop down list με τις διαθέσιμες επιλογές αναζήτησης ενός ασθενή. Επίσης εδώ δηλώνεται η κλήση της javascript μεθόδου change_element_type κάθε φορά που ο χρήστης επιλέγει νέο κριτήριο για αναπροσαρμογή του πεδίου για την είσοδο της τιμής αναζήτησης.
    <select name="Criterion" id="Criterion" onChange="change_element_type()">
      <option value="Name">Name</option>
      <option value="idPatient">Political ID</option>
      <option value="dateBirth">Birth Date</option>
      <option value="dateHospitalizationStart">Hospitalized Date</option>
      <option value="idClinic">Clinic</option>
    </select>
// Textfield για είσοδο της τιμής αναζήτησης
    <input type="text" name="Crit_value" id="Crit_value" required/>
</div>
     <p class="submit" align="center">
                    <input type="submit" value="Search">
            </p>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
<footer>
                    <section id="footer-area">
                       <p align="center"> Copyright &copy; 2012 <a href="#">heraklionhospital.com</a>. All Rights Reserved</p>
             </section><!-- end of footer-area -->
            </footer>

</div><!-- #container -->

<!-- InstanceBeginEditable name="JavaScript Code" -->
// Javascript block
<script type="text/javascript">
// Javascript μέδοθος για την αλλαγή του input type του πεδίου 'Crit_value'
function change_element_type(){

            var e = document.getElementById("Criterion");
            var criterion = e.options[e.selectedIndex].value;

// Έαν ο χρήστης επέλεξε ως κριτήριο της αναζήτησης κάποια ημερομηνία τότε το input type αλλάζει σε date, αλλιώς σε text.
            if( criterion.match(/^date/) ){
                    document.getElementById('Crit_value').type = 'date';
            }else{
                    document.getElementById('Crit_value').type = 'text';
            }
}
</script>
<!-- InstanceEndEditable -->


</body>
<!-- InstanceEnd --></html>
```

11. Εύρεση/Προβολή ασθενών

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<?php

            $con = mysql_connect('localhost','root', '');
    if (!$con)
                    {
                            die('Could not connect: ' . mysql_error());
                    }

     mysql_select_db("hospital", $con);
```

```
		$where = '';

// Block για την αναζήτηση ασθενή
	if( isset($_POST['Criterion']) && isset($_POST['Crit_value'])){

		$criterion = $_POST['Criterion'];
		$value = $_POST['Crit_value'];

		$where = ' WHERE ';

// Ανάλογα με το κριτήριο που επέλεξε ο χρήστης για την αναζήτηση ενός ασθενή, δημιουργείται δυναμικά το query αναζήτησης
		switch($criterion)
		{
			case "Name":
				$where .= "d.Name LIKE '%$value%'";
				break;
			case "idPatient":
				$where .= "d.idPatient = '$value'";
				break;
			case "dateBirth":
				$where .= "d.dateBirth = '$value'";
				break;
			case "dateHospitalizationStart":
				$where .= "d.dateHospitalizationStart = '$value'";
				break;
			case "idClinic":
				$where .= "d.idClinic = '$value'";
				break;
		}

		echo "<h2 align=\"center\">Patients Search Results</h2>";
			echo "<h4 align=\"center\">($criterion='$value')</h4>";

		}else{
			echo "<h2 align=\"center\">Registered Patients</h2>";
		}



// Τελικό query αναζήτησης
$query = "SELECT d.*, c.Name AS clinic_name FROM patients d
		LEFT OUTER JOIN clinics c ON c.idClinic = d.idClinic $where";

// Εκτέλεση query
	$result = mysql_query($query, $con);

// Προβολή αποτελεσμάτων σε πίνακα.
	echo "<table  border=\"1\" id=\"table-3\"><col width=\"5%\" /><col width=\"15%\" /><col width=\"10%\" /><col
width=\"15%\" /><col width=\"10%\" />
			<col width=\"10%\" /><col width=\"5%\" /><col width=\"5%\" /><col width=\"5%\" />";
		echo "<thead><tr><th>Patient ID</th> <th>Full Name</th>
				<th>Insurance</th>		<th>Address</th> <th>Clinic</th> <th>Hospitalized
On</th>
			 </tr></thead>";

	while($row = mysql_fetch_array($result))
	{
			$Name = $row['Name'];
			$BirthDate = $row['dateBirth'];
			$Sex = $row['Sex'];
			$Insurance = $row['Insurance'];
			$id = $row['idPatient'];
			$Address = $row['Street'] . ", " . $row['PostalCode'] . ", " . $row['City'];
			$clinic = $row['clinic_name'];
```

```
                                $hd = $row['dateHospitalizationStart'];

                                $extra_col = '';

                                if( isset($clinic) && isset($hd) ){
                                        $extra_col = "<td><a
href='editPatient.php?idPatient=$id&action=Checkout'>checkout</a></td>";
                                }

                                echo "<tr><td>$id</td><td>$Name</td><td>$Insurance</td>
                                        <td>$Address</td><td>$clinic</td><td>$hd</td><td><a
href='editPatientForm.php?patient_id=$id'>edit</a></td>
                                                                                        <td><a
href='editPatient.php?idPatient=$id&action=Delete'>delete</a></td>$extra_col</tr>";
                        }
                        echo "</table>";
        mysql_close($con);
?>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

12. Φόρμα δημιουργίας νέου γιατρού(front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">

// Δήλωση νέας φόρμας και το php αρχείο για την επεξεργασία εισόδου και δημιουργία του γιατρού
  <form action="addDoctor_newRecord.php" method="post">
            <fieldset>
            <legend align="center" color="black" font="14px"> <b>New Doctor Information</b> </legend><br>

// Δημιουργία της φόρμας. Σε όλα τα fields γίνεται η χρήση του HTML5 attribute required.
<div>
            <label>Full Name:</label> <input type="text" name="Name" required/>
</div>

//Σε αυτό το πεδίο φαίνεται η χρήση του νέου feature της HTML5 pattern.
<div>
            <label>Political ID:</label> <input title="8 Digits Number" name="idDoctor" pattern="[0-9]{8}"
autocomplete="off" required/> <caption><i>(8 digits number)</i></caption> <span id="txtHint"></span>
</div>

<div>
            <label>Specialty:</label> <input type="text" name="Specialty" required/>
</div>

<div>
            <label>Clinic:</label>
<?php
            $query = "SELECT idClinic, Name FROM clinics ORDER BY Name";
            fill_dropdown_list("idClinic", "Name", "idClinic", $query, 0);
                                                ?>
</div>

<div>
            <label>Employment Date:</label> <input type="date" name="dateEmployment" max="<?php echo date('Y-m-d',
strtotime(date('Y/m/d'))); ?>" required/>
</div>

<div>
            <label>Address:</label> <input type="text" name="Street" required/>
</div>

<div>
```

```
            <label>Postal Code:</label> <input id="postcode" title="5 digits number" type="text" name="PostalCode"
pattern="[1-9][0-9]{4}" required/> <caption><i>(5 digits number e.g. 30200)</i></caption>
</div>

<div>
            <label>City:</label> <input type="text" name="City" required/>
</div>

<div>
            <label>Phone:</label> <input type="text" name="Phone" pattern="[0-9]{10}" required/>
</div>

<div>
    <label>Category:</label>
    <select name="DoctorType" id="docType" onChange="hide_show_fields()">
          <option value="Clinical">Clinical</option>
          <option value="Laboratory">Laboratory</option>
     </select>
</div>

<div id="clinicalBlock">
            <label>Experience:</label>
         <input type="number" name="Years" id="Years" min=0 value='0'/>  <caption>years</caption>
            </div>

<div id="labBlock" style="display:none">
            <label>Expertise:</label>
          <select name="DiseaseType">
            <option value="viral">Viral</option>
            <option value="hereditary">Hereditary</option>
          </select>
</div>

<p class="buttons">
                        <input type="submit">
</p>

</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
            <footer>
                        <section id="footer-area">
                           <p align="center"> Copyright &copy; 2012 <a href="#">heraklionhospital.com</a>. All Rights
Reserved</p>
              </section><!-- end of footer-area -->
            </footer>

</div><!-- #container -->

<!-- InstanceBeginEditable name="JavaScript Code" -->
<script type="text/javascript">
// Javascript μέθοδος για την εμφάνιση/απόκρυψη πεδίων ανάλογα με τον επιλεγμένο τύπο γιατρού
function hide_show_fields(){

            var e = document.getElementById("docType");
            var category = e.options[e.selectedIndex].value;

// Αν η επιλεγμένη κατηγορία γιατρού είναι Clinical τότε τα πεδία για τους Laboratory κρύβονται.
            if(category == 'Clinical'){
                        document.getElementById("clinicalBlock").style.display='block';
                        document.getElementById("labBlock").style.display='none';
                        //document.getElementById("Years").setAttribute('required',true);
            //          document.getElementById("DiseaseType").set setAttribute('required',false);
```

```
// Αν η επιλεγμένη κατηγορία γιατρού είναι Laboratory τότε τα πεδία εμφανίζονται στην φόρμα ενώ τα πεδία σχετικά με Clinical κρύβονται.
}else{
                document.getElementById("clinicalBlock").style.display='none';
                document.getElementById("labBlock").style.display='block';
                //document.getElementById("Years").setAttribute('required',false);
                //document.getElementById("DiseaseType").setAttribute('required',true);
        }
}

// Κλήση της μεθόδου hide_show_fields κάθε φορά που φορτώνει η σελίδα
window.onload = function () {
        hide_show_fields();
}
</script><!-- InstanceEndEditable -->
```

13. Δημιουργία νέου γιατρού(back-end)

```
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
<form>
    <fieldset>
        <legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

<?php

        $con = mysql_connect('localhost','root', '');
    if (!$con)
        {
            echo '<p>Could not connect: ' . mysql_error() . '</p>';
        }

    mysql_select_db("hospital", $con);

// Αποθήκευση δεδομένων εισόδου σε php μεταβλητές
    $name = $_POST['Name'];
        $doctor_id = $_POST['idDoctor'];
        $sp = $_POST['Specialty'];
        $dem = $_POST['dateEmployment'];
        $street = $_POST['Street'];
        $pc = $_POST['PostalCode'];
        $city = $_POST['City'];
        $phone = $_POST['Phone'];
        $doc_type = $_POST['DoctorType'];
        $clinic= $_POST['idClinic'];

        if( $doc_type == 'Clinical'){
            $years = $_POST['Years'];
        }else{
            $diseace_type = $_POST['DiseaseType'];
        }

// Query για νέα εγγραφή νέου γιατρού
        $sql = "INSERT INTO doctors (idDoctor, Name, Specialty, Street, PostalCode, City, Phone, dateEmployment, idClinic)
                VALUES ($doctor_id,'$name','$sp', '$street', '$pc', '$city', '$phone', '$dem', '$clinic')";

//Εκτέλεση query
        if (!mysql_query($sql,$con))
        {
            echo "<p align='center'>Error: ".mysql_error()."!<br>Please try again.</p>";
        }else{

// Ανάλογα με την κατηγορία του γιατρού εκτελείται το αντίστοιχο query για νέα εγγραφή στο table clinicaldoctor ή στο table laboratorydoctor.
        if( $doc_type == 'Clinical'){
```

```
                $sql = "INSERT INTO clinicaldoctor (idDoctor, Years) VALUES ($doctor_id, $years)";
        }else{
                $sql = "INSERT INTO laboratorydoctor (idDoctor, DiseaseType) VALUES ($doctor_id,
'$diseace_type')";
        }

        if (!mysql_query($sql,$con))
        {
                echo "<p align='center'>Error: ".mysql_error()."!<br>Please try again.</p>";
        }

// Εμφάνιση μηνύματος για την επιτυχή δημιουργία γιατρού.
        echo "<p align='center'>Doctor [$name] was succesfully added in the system!
<br><a href='editDoctorForm.php?doctor_id=$doctor_id'>Edit Doctor</a></p>";
}

//Κλείσιμο σύνδεσης με τον mysql server
    mysql_close($con);
?>
        <!-- InstanceEndEditable -->
   </section><!-- end of #main content -->
```

14. Φόρμα επεξεργασίας γιατρού(front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
// PHP Block για την ανάκτηση των στοιχείων του γιατρού προς επεξεργασία
<?php
        $doctor_id=$_GET["doctor_id"];

        $con = mysql_connect('localhost','root', '');
    if (!$con)
        {
                die('Could not connect: ' . mysql_error());
        }

        mysql_select_db("hospital", $con);

// Query για ανάκτηση των στοιχείων του γιατρού από την βάση
$query = "SELECT d.*, COALESCE(cd.Years, ld.DiseaseType) AS add_info, CASE WHEN cd.idDoctor IS NULL THEN
'Laboratory' ELSE 'Clinical' END AS DocType
  FROM doctors d
        LEFT OUTER JOIN clinicaldoctor cd ON cd.idDoctor = d.idDoctor
        LEFT OUTER JOIN laboratorydoctor ld ON ld.idDoctor = d.idDoctor
 WHERE d.idDoctor=".$doctor_id;

 $result = mysql_query($query, $con);

// Αποθήκευση των στοιχείων σε php μεταβλητές για προβολή τους στην φόρμα
                $row = mysql_fetch_array($result);

                $id = $row['idDoctor'];
                $name = $row['Name'];
                $sp = $row['Specialty'];
                $dem = $row['dateEmployment'];
                $street = $row['Street'];
                $pc = $row['PostalCode'];
                $city = $row['City'];
                $phone = $row['Phone'];
                $clinic= $row['idClinic'];
                $doc_type = $row['DocType'];

                $prev_doc_type = $doc_type;

                if($doc_type == 'Clinical'){
                        $years = $row['add_info'];
```

```
                        $disease_type = '';
                }else{
                        $years = '';
                        $disease_type = $row['add_info'];
                }
        ?>

   <div id="form_data">
                <form action="editDoctor.php" method="post">
                                        <fieldset>
<legend align="center" color="black" font="14px"> <b>Doctor Information</b> </legend><br>

<div>
        <input type="hidden" name="idDoctor" value="<?php echo $id ?>" required/>
</div>

<div>
        <input type="hidden" name="prevDocType" value="<?php echo $prev_doc_type ?>" required/>
</div>

<div>
        <label>Full Name:</label> <input type="text" name="Name" value="<?php echo $name ?>" required/>
</div>

<div>
        <label>Political ID:</label> <input title="8 Digits Number" name="idDoctor" value="<?php echo $id ?>"
readonly="readonly" required/><caption><i>read-only</i></caption>
</div>

<div>
        <label>Specialty:</label> <input type="text" name="Specialty" value="<?php echo $sp ?>" required/>
</div>

<div>
        <label>Clinic:</label>
<?php
        $query = "SELECT idClinic, Name FROM clinics ORDER BY Name";
        fill_dropdown_list("idClinic", "Name", "idClinic", $query, 0);
?>
</div>

<div>
        <label>Employment Date:</label> <input type="date" name="dateEmployment" max="<?php echo date('Y-m-d',
strtotime(date('Y/m/d'))); ?>" value="<?php echo $dem ?>" required/>
</div>

<div>
        <label>Address:</label> <input type="text" name="Street" value="<?php echo $street ?>" required/>
</div>

<div>
        <label>Postal Code:</label> <input id="postcode" title="5 digits number" type="text" name="PostalCode"
pattern="[1-9][0-9]{4}" value="<?php echo $pc ?>" required/> <caption><i>(5 digits number e.g.
30200)</i></caption>
</div>

<div>
        <label>City:</label> <input type="text" name="City" value="<?php echo $city ?>" required/>
</div>

<div>
        <label>Phone:</label> <input type="text" name="Phone" pattern="[0-9]{10}" value="<?php echo $phone ?>"
required/>
</div>

<div>
```

```
   <label>Category:</label>
      <select name="DoctorType" id="docType" onChange="hide_show_fields()">
        <option value="Clinical">Clinical</option>
        <option value="Laboratory">Laboratory</option>
       </select>
</div>
// Block εμφάνισης του αποκλειστικού πεδίου years για του κλινικούς γιατρούς
<div id="clinicalBlock">
            <label>Experience:</label>
      <input type="number" name="Years" id="Years" min=0 value="<?php echo $years ?>" />
 <caption>years</caption>
</div>
// Block εμφάνισης του αποκλειστικού πεδίου expertise για του κλινικούς γιατρούς
<div id="labBlock" style="display:none">
            <label>Expertise:</label>
        <select name="DiseaseType" id="dType">
           <option value="viral">Viral</option>
           <option value="hereditary">Hereditary</option>
        </select>
</div>

<div>
  <table class="buttons" cellspacing="5">
  <tr>
    <td><input id="delete" name="action" value="Delete" type="submit" onClick="return confirm(
  'This will delete all data related to this doctor. Are you sure you want to continue?');"/></td>
    <td><input id="update" name="action" value="Update" type="submit"/></td>
  </tr>
     </table>
</div>
                        </fieldset>
                        </form>
            </div>
            <!-- InstanceEndEditable -->
   </section><!-- end of #main content -->
            <footer>
                        <section id="footer-area">
                           <p align="center"> Copyright &copy; 2012 <a href="#">heraklionhospital.com</a>. All Rights
Reserved</p>
              </section><!-- end of footer-area -->
            </footer>

</div><!-- #container -->

<!-- InstanceBeginEditable name="JavaScript Code" -->
<script type="text/javascript">
// Javascript μέθοδος για την εμφάνιση/απόκρυψη πεδίων ανάλογα με τον επιλεγμένο τύπο γιατρού
function hide_show_fields(){

            var e = document.getElementById("docType");
            var category = e.options[e.selectedIndex].value;

            if(category == 'Clinical'){
                        document.getElementById("clinicalBlock").style.display='block';
                        document.getElementById("labBlock").style.display='none';
                        //document.getElementById("Years").setAttribute('required',true);
            //          document.getElementById("DiseaseType").set setAttribute('required',false);
            }else{
                        document.getElementById("clinicalBlock").style.display='none';
                        document.getElementById("labBlock").style.display='block';
                        //document.getElementById("Years").setAttribute('required',false);
                        //document.getElementById("DiseaseType").setAttribute('required',true);
            }
}
```

```
// Javascript μέθοδος που καλείται με το φόρτωμα της σελίδας και ανάλογα με την κατηγορία του γιατρού προς επεξεργασία, αρχικοποιεί, εμφανίζει και αποκρύβει τα κατάλληλα πεδία.
window.onload = function () {
		//Set default values to dropdown lists
		var clinic_value = "<?= $clinic ?>";

		if(clinic_value > 0){
			document.getElementById('idClinic').value =  clinic_value;
		}

		var doc_type = "<?= $doc_type ?>";

		if(doc_type){
			document.getElementById('docType').value =  doc_type;
		}

		var d_type = "<?= $disease_type ?>";
		if(d_type){
			document.getElementById('dType').value =  d_type;
		}

		hide_show_fields();
}

</script>
<!-- InstanceEndEditable -->
```

15. Επεξεργασίας γιατρού(back-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
<form>
    <fieldset>
<legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

//PHP block για την επεξεργασία των δεδομένων εισόδου και την δημιουργία νέας εγγραφής γιατρού στην βάση δεδομένων.
<?pup
    $con = mysqli_connect('localhost','root', '');

		if (!$con)
		{
			echo '<p>Could not connect: ' . mysqli_error() . '</p>';
		}

     mysqli_select_db($con, "hospital");

		$action = isset($_POST['action']) ? $_POST['action'] : $_GET['action'];

		$doctor_id = isset($_POST['idDoctor']) ? $_POST['idDoctor'] :  $_GET['idDoctor'];

//1.Block για διαγραφή γιατρού από την βάση δεδομένων
if($action == 'Delete'){
			mysqli_autocommit($con, false);

			$flag = true;

			$doc_type = isset($_POST['DoctorType']) ? $_POST['DoctorType'] :  $_GET['DoctorType'];

//Query για την διαγραφή όλων των προγραμματισμένων εξετάσεων που διέταξε ο γιατρός προς διαγραφή
			$query1 = "DELETE FROM pde WHERE idDoctor=$doctor_id";

// Ανάλογα με την κατηγορία του γιατρού, query για την διαγραφή του από τους αντίστοιχους πίνακες
        if(  $doc_type == 'Clinical'){
				$query2 = "DELETE FROM clinicaldoctor WHERE idDoctor=$doctor_id";
			}else{
```

```
                        $query2 = "DELETE FROM laboratorydoctor WHERE idDoctor=$doctor_id";
                }

//Query για την διαγραφή του γιατρού από τον πίνακα doctors
                $query3 = "DELETE FROM doctors WHERE idDoctor=$doctor_id";

//Εκτέλεση των queries
        $result = mysqli_query($con, $query1);

                if(!$result){
                        $flag = false;
                        $msg = "Doctor cannot be deleted since there are scheduled exams linked to this doctor";
                }

                $result = mysqli_query($con, $query2);

                if(!$result){
                                $flag = false;
                }

                $result = mysqli_query($con, $query3);

                if(!$result){
                        $flag = false;
                        $msg = "Doctor cannot be deleted because is a clinic's manager";
                }

                if($flag){
                        mysqli_commit($con);
                        echo "<p align='center'>Doctor was succesfully deleted from the system!</p>";
                }else{
                        mysqli_rollback($con);
                        echo "<p align='center'>Error: $msg!<br><a
href='editDoctorForm.php?doctor_id=$doctor_id'>Back to Doctor</a></p>";
                }
//2.Block για την ανανέωση των στοιχείων ενός γιατρού
}else if($action == 'Update'){

                $id = $_POST['idDoctor'];
                $name = $_POST['Name'];
                $sp = $_POST['Specialty'];
                $dem = $_POST['dateEmployment'];
                $street = $_POST['Street'];
                $pc = $_POST['PostalCode'];
                $city = $_POST['City'];
                $phone = $_POST['Phone'];
                $clinic= $_POST['idClinic'];
                $doc_type = $_POST['DoctorType'];
                $prev_doc_type = $_POST['prevDocType'];

                $flag = true;
                mysqli_autocommit($con, false);

                if($doc_type == 'Clinical'){
                        $years = $_POST['Years'];

                if($prev_doc_type != $doc_type){
                                $sql = "DELETE FROM laboratorydoctor WHERE idDoctor=$id"                if
(!mysqli_query($con, $sql))
                {               $flag = false;
                        echo "7";
                }

                $sql = "INSERT INTO clinicaldoctor (idDoctor, Years) VALUES($id, $years)";

                if (!mysqli_query($con, $sql))
```

```
                {       $flag = false;
                        echo "6";
                }

        }else{

                $sql = "UPDATE clinicaldoctor
                                SET Years = $years
                                WHERE idDoctor = $id";

                if (!mysqli_query($con, $sql))
                {       $flag = false;
                        echo "5";
                        }
        }

        }else{
                $disease_type = $_POST['DiseaseType'];

                if($prev_doc_type != $doc_type){
                        $sql = "DELETE FROM clinicaldoctor WHERE idDoctor=$id";

                if (!mysqli_query($con, $sql))
                        {       $flag = false;
                                echo "4";
                        }

                $sql = "INSERT INTO laboratorydoctor (idDoctor, DiseaseType) VALUES($id, '$disease_type')";

                if (!mysqli_query($con, $sql))
                {       $flag = false;
                                                        echo "3";
                }

        }else{

                $sql = "UPDATE laboratorydoctor
                        SET DiseaseType = '$disease_type'
                        WHERE idDoctor = $id";


if (!mysqli_query($con, $sql))
{       $flag = false;
                echo "1";
}
}
}

$sql = "UPDATE doctors
        SET Name = '$name', Specialty = '$sp', dateEmployment = '$dem', Street = '$street', PostalCode = '$pc', City =
'$city', Phone = '$phone', idClinic = $clinic
        WHERE idDoctor = $id";

        if (!mysqli_query($con, $sql))
        {
                $flag = false;
        // Σε περίπτωση που όλα τα queries για την ανανέωση των στοιχείων ενός γιατρού εκτελέστηκαν με επιτυχία, τότε
γίνονται commit οι αλλαγές, αλλιώς rollback.
    if($flag){
                mysqli_commit($con);
                echo "<p align='center'>Doctor [$name] details were succesfully updated!<br><a
href='editDoctorForm.php?doctor_id=$id'>Back to Doctor</a></p>";
        }else{
                mysqli_rollback($con);
                echo "<p align='center'>Error: One of the queries failed!<br>Please try again.<br><a
href='editDoctorForm.php?doctor_id=$doctor_id'>Back to Doctor</a></p>";
```

```
                        }
                }
                mysqli_close($con);
        ?>
                </fieldset>
                </form>
</div>
        <!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

16. Φόρμα εύρεσης γιατρού (front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
 <div id="form_data">
//Δήλωση νέας φόρμας και του php αρχείου για την προβολή των αποτελεσμάτων
<form action="viewDoctors.php" method="post">
<fieldset>
<legend align="center" color="black" font="14px"> <b>Find Doctor</b> </legend><br>

<div>
   <label>Search by:</label>
//Δήλωση drop down list με τις διαθέσιμες επιλογές αναζήτησης ενός γιατρού. Επίσης εδώ δηλώνεται η κλήση της javascript μεθόδου change_element_type κάθε φορά που ο χρήστης επιλέγει νέο κριτήριο για αναπροσαρμογή του πεδίου για την είσοδο της τιμής αναζήτησης.
<select name="Criterion" id="Criterion" onChange="change_element_type()">
        <option value="Name">Name</option>
        <option value="Specialty">Specialty</option>
        <option value="idClinic">Clinic</option>
        <option value="dateEmployment">Employment Date</option>
      </select>
   <input type="text" name="Crit_value" id="Crit_value" required/>
</div>
     <p class="submit" align="center">
                <input type="submit" value="Search">
        </p>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
<footer>
<section id="footer-area">
<p align="center"> Copyright &copy; 2012 <a href="#">heraklionhospital.com</a>. All Rights Reserved</p>
</section><!-- end of footer-area -->
</footer>

</div><!-- #container -->

<!-- InstanceBeginEditable name="JavaScript Code" -->
<script type="text/javascript">
// Javascript μέδοθος για την αλλαγή του input type του πεδίου 'Crit_value'
function change_element_type()
        var e = document.getElementById("Criterion");
        var criterion = e.options[e.selectedIndex].value;
// Εάν ο χρήστης επέλεξε ως κριτήριο της αναζήτησης κάποια ημερομηνία τότε το input type αλλάζει σε date, αλλιώς σε text.

        if( criterion.match(/^date/) ){
                document.getElementById('Crit_value').type = 'date';
        }else{
                document.getElementById('Crit_value').type = 'text';
        }
}
</script>
<!-- InstanceEndEditable -->
```

17. Εύρεση/Προβολή γιατρών

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
 <?php

// Σύνδεση με τον mysql server και την βάση
        $con = mysql_connect('localhost','root', '');
    if (!$con)
        {
            die('Could not connect: ' . mysql_error());
        }

    mysql_select_db("hospital", $con);

        $where = '';
// Block για την αναζήτηση ασθενή
        if( isset($_POST['Criterion']) && isset($_POST['Crit_value'])){

                        $criterion = $_POST['Criterion'];
                        $value = $_POST['Crit_value'];

                        $where = ' WHERE ';
// Ανάλογα με το κριτήριο που επέλεξε ο χρήστης για την αναζήτηση ενός γιατρού, δημιουργείται δυναμικά το query αναζήτησης
                        switch($criterion)
                        {
                            case "Name":
                                $where .= "d.Name LIKE '%$value%'";
                                break;
                            case "Specialty":
                                $where .= "d.Specialty LIKE '%$value%'";
                                break;
                            case "idClinic":
                                $where .= "c.Name = '$value'";
                                break;
                            case "dateEmployment":
                                $where .= "d.dateEmployment = '$value'";
                                break;

                        }

                echo "<h2 align=\"center\">Doctors Search Results</h2>";
                echo "<h4 align=\"center\">($criterion='$value')</h4>";

        }else{
                echo "<h2 align=\"center\">Registered Doctors</h2>";
        }

//Query για την επιλογή των γιατρών που πληρούν τα κριτήρια
        $query = "SELECT d.idDoctor, d.Name, d.Specialty, c.Name AS clinic_name, CASE WHEN c.idManager > 0
AND c.idManager = d.idDoctor THEN 1 ELSE 0 END AS is_manager,CASE WHEN cd.idDoctor IS NOT NULL THEN
'Clinical' ELSE 'Laboratory' END AS doc_type
        FROM doctors d
                INNER JOIN clinics c ON c.idClinic = d.idClinic
                LEFT OUTER JOIN clinicaldoctor cd ON cd.idDoctor = d.idDoctor
                LEFT OUTER JOIN laboratorydoctor ld ON ld.idDoctor = d.idDoctor
                $where
        ORDER BY d.idDoctor";

        $result = mysql_query($query, $con);

// Επεξεργασία και προβολή των αποτελεσμάτων σε πίνακα
        echo "<table border=\"1\" id=\"table-3\"><col width=\"5%\" /><col width=\"20%\" /><col width=\"20%\" /><col
width=\"20%\" /><col width=\"10%\" />
                        <col width=\"10%\" /><col width=\"5%\" /><col width=\"5%\" />";
```

```
                    echo "<thead><tr><th>Doctor ID</th> <th>Name</th>
                                <th>Specialty</th>          <th>Clinic</th> <th>Is
Manager</th><th>Type</th><th></th><th></th>
                                </tr></thead>";

                    while($row = mysql_fetch_array($result))
                    {
                            $id = $row['idDoctor'];
                            $Name = $row['Name'];
                            $Specialty = $row['Specialty'];
                            $clinic_name = $row['clinic_name'];
                            $is_manager = $row['is_manager'];
                            $doc_type = $row['doc_type'];

                            $flag='';

                            if($is_manager){
                                    $flag = 'YES';
                            }

                    //<input type=\"checkbox\" name\"cb\" $flag disabled=\"disabled\">

                    echo "<tr><td>$id</td><td>$Name</td><td>$Specialty</td>
                                <td>$clinic_name</td><td style=\"color:green\">$flag</td><td>$doc_type</td><td><a
href='editDoctorForm.php?doctor_id=$id'>edit</a></td>
                                                        <td><a
href='editDoctor.php?idDoctor=$id&action=Delete&DoctorType'>delete</a></td></tr>";
 }
echo "</table>";
mysql_close($con);
?>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

18. Φόρμα δημιουργίας νέας κλινικής(front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
// Δήλωση νέας φόρμας και το php αρχείο για την επεξεργασία εισόδου και δημιουργία της νέας κλινικής
<form action="addClinic_newRecord.php" method="post">
        <fieldset>
        <legend align="center" color="black" font="14px"> <b>New Clinic Information</b> </legend><br>

<div>
        <label>Name:</label> <input type="text" name="Name" required/>
</div>

// Σε αυτό το πεδίο φαίνεται η χρήση του νέου HTML5 input type number
<div>
        <label>Number of Beds:</label> <input type="number" name="BedsNo" min=0 required/>
</div>

<p class="buttons">
        <input type="submit">
</p>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

19. Δημιουργία νέας κλινικής(back-end)

```
<section id="main"><!-- #main content  -->
```

```
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
<form>
    <fieldset>
            <legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>
// PHP block για την επεξεργασία εισόδου και δημιουργία νέας κλινικής
<?php
            $con = mysql_connect('localhost','root', '');
     if (!$con)
            {
                    die('Could not connect: ' . mysql_error());
            }

    mysql_select_db("hospital", $con);

    $name = $_POST['Name'];
            $beds = $_POST['BedsNo'];
//Query για εισαγωγή νέας κλινικής στο σύστημα.
            $sql = "INSERT INTO clinics (Name, BedsNo, idManager, dateStart) VALUES ('$name', '$beds', NULL,
NULL)";

            if (!mysql_query($sql,$con))
            {
                    echo "<p align='center'>Error: ".mysql_error()."!<br>Please try again.</p>";
            }else{
                    $clinic_id = mysql_insert_id();

                    echo "<p align='center'>Clinic [$name] was succesfully added in the system!<br><a
href='editClinicForm.php?clinic_id=$clinic_id'>Edit Clinic</a></p>";
}
//Κλείσιμο σύνδεσης με τον mysql server
mysql_close($con);
?>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

20. Φόρμα επεξεργασίας κλινικής(front-end)

```
section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
// PHP Block για την ανάκτηση των στοιχείων της κλινικής προς επεξεργασία
<?php
            $clinic_id=$_GET["clinic_id"];

            $con = mysql_connect('localhost','root', '');
     if (!$con)
            {
                    die('Could not connect: ' . mysql_error());
            }

            mysql_select_db("hospital", $con);

// Query για την ανάκτηση των στοιχείων της κλινικής απο την βάση.
            $query = "SELECT * FROM clinics WHERE idClinic=".$clinic_id;

     $result = mysql_query($query, $con);

            $row = mysql_fetch_array($result);
// Αποθήκευση των στοιχείων σε php μεταβλήτες για προβολή τους στην φόρμα

                    $id = $row['idClinic'];
                    $Name = $row['Name'];
                    $BedsNo = $row['BedsNo'];
```

```
                        $idManager = $row['idManager'];
                        $dateStart = $row['dateStart'];
                ?>

<div id="form_data">
                <form action="editClinic.php" method="post">
<fieldset>
<legend align="center" color="black" font="14px"> <b>Clinic Information</b> </legend><br>

<div>
                <input type="hidden" name="idClinic" value="<?php echo $id ?>" required/>
</div>

<div>
                <label>Name:</label> <input type="text" name="Name"  value="<?php echo $Name ?>" required/>
</div>

<div>
                <label>Number of Beds:</label> <input type="number" name="BedsNo" min=0 value="<?php echo $BedsNo ?>"
required/>
</div>

<div>
                <label>Manager:</label>
<?php
                $query = "SELECT d.idDoctor, d.Name FROM doctors d WHERE d.idClinic = $id ORDER BY d.Name";
                fill_dropdown_list("idDoctor", "Name", "idManager", $query, 1); ?>
</div>
<div>
<label>Manager Empl Date:</label> <input type="date" name="dateStart"  max="<?php echo date('Y-m-d',
strtotime(date('Y/m/d'))); ?>" value="<?php echo $dateStart ?>" />
</div>

<div>
  <table class="buttons" cellspacing="5">
  <tr>
    <td><input id="delete" name="action" value="Delete" type="submit" onClick="return confirm(
 'This will delete all data related to this clinic. Are you sure you want to continue?');"/></td>

<td><input id="update" name="action" value="Update" type="submit"/></td>
</tr>
</table>
</div>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
                <footer>
                        <section id="footer-area">
                           <p align="center"> Copyright &copy; 2012 <a href="#">heraklionhospital.com</a>. All Rights
Reserved</p>
                          </section><!-- end of footer-area -->
                </footer>

</div><!-- #container -->

<!-- InstanceBeginEditable name="JavaScript Code" -->
<script type="text/javascript">
//Javascript μέθοδος για τον καθορισμό default τιμής στο dropdown list.
window.onload = function () {
                //Set default values to dropdown lists
                var manager = "<?= $idManager ?>";

                if(manager > 0){
                        document.getElementById('idManager').value =  manager;
```

```
		}
}
</script>
<!-- InstanceEndEditable -->
```

21. Επεξεργασίας κλινικής(back-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
<form>
   <fieldset>
<legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

//PHP block για την επεξεργασία των δεδομένων εισόδου και την δημιουργία νέας εγγραφής κλινικής στην βάση δεδομένων.
<?php

// Σύνδεση με την βάση δεδομένων.
		$con = mysql_connect('localhost','root', '');

		if (!$con)
		{
			echo '<p>Could not connect: ' . mysql_error() . '</p>';
		}

    mysql_select_db("hospital", $con);

		$action = isset($_POST['action']) ? $_POST['action'] : $_GET['action'];

		$clinic_id = isset($_POST['idClinic']) ? $_POST['idClinic'] :  $_GET['idClinic'];

// 1. Block για διαγραφή μιας κλινικής από την βάση δεδομένων.
		if($action == 'Delete'){

		$sql = "DELETE FROM clinics WHERE idClinic=$clinic_id";

		if (!mysql_query($sql,$con)){
			echo "<p align='center'>Error: Clinic cannot be deleted!<br>Other entities like doctors and/or patients
are linked to this clinic.<br><a href='editClinicForm.php?clinic_id=$clinic_id'>Back to Clinic</a></p>";

}else{
		echo "<p align='center'>Clinic was succesfully deleted from the system!</p>";
}

// 2. Block για ανανέωση των στοιχείων μιας κλινικής από την βάση δεδομένων.
		}else if($action == 'Update'){
				$name = $_POST['Name'];
				$beds = $_POST['BedsNo'];
				$dateStart = $_POST['dateStart'];
				$manager_id = $_POST['idManager'];

				if($dateStart == '' || $manager_id == ''){
					$sql = "UPDATE clinics
						SET Name = '$name', BedsNo = $beds, dateStart = NULL, idManager =
NULL
						WHERE idClinic = ".$clinic_id;
				}else{
					$sql = "UPDATE clinics
						SET Name = '$name', BedsNo = $beds, dateStart = '$dateStart', idManager
= $manager_id
						WHERE idClinic = ".$clinic_id;
				}

			if (!mysql_query($sql,$con))
			{
				echo "<p align='center'>Error: ".mysql_error()."!<br>Please try again.</p>";
```

```
                }else{
                        echo "<p align='center'>Clinic [$name] details were succesfully updated!<br><a
href='editClinicForm.php?clinic_id=$clinic_id'>Back to Clinic</a></p>";
}
}
        mysql_close($con);
?>
                </fieldset>
                </form>
   </div>
        <!-- InstanceEndEditable -->
   </section><!-- end of #main content -->
```

22. Φόρμα εύρεσης κλινικής (front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
//Δήλωση νέας φόρμας και του php αρχείου για την προβολή των αποτελεσμάτων
<form action="viewClinics.php" method="post">
<fieldset>
<legend align="center" color="black" font="14px"> <b>Find Clinic</b> </legend><br>

<div>
//Δήλωση drop down list με τις διαθέσιμες επιλογές αναζήτησης μιας κλινικής.
  <label>Search by:</label>
     <select name="Criterion" id="Criterion">
        <option value="Name">Name</option>
        <option value="Manager">Manager</option>
      </select>
  <input type="text" name="Crit_value" id="Crit_value" required/>
</div>

<p class="submit" align="center">
        <input type="submit" value="Search">
</p>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

23. Εύρεση/Προβολή κλινικών

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->

<?php

        $con = mysql_connect('localhost','root', '');
    if (!$con)
        {
                die('Could not connect: ' . mysql_error());
        }

        mysql_select_db("hospital", $con);

        $where = '';
        // Block για την αναζήτηση ασθενή
        if( isset($_POST['Criterion']) && isset($_POST['Crit_value'])){

                $criterion = $_POST['Criterion'];
                $value = $_POST['Crit_value'];

                $where = ' WHERE ';
```

```
// Ανάλογα με το κριτήριο που επέλεξε ο χρήστης για την αναζήτηση μιας κλινικής, δημιουργείται δυναμικά το query αναζήτησης
                        switch($criterion)
                        {
                                case "Name":
                                        $where .= "c.Name LIKE '%$value%'";
                                        break;
                                case "Manager":
                                        $where .= "d.Name = '$value'";
                                        break;
                        }

                        echo "<h2 align=\"center\">Clinics Search Results</h2>";
                        echo "<h4 align=\"center\">($criterion='$value')</h4>";
                }else{
                        echo "<h2 align=\"center\">Registered Clinics</h2>";
                }
//Query για ανάκτηση των κλινικών που πληρούν τα κριτήρια.
                $query = "SELECT c.*, d.Name AS doc_name FROM clinics c
                                LEFT OUTER JOIN doctors d ON d.idDoctor =c.idManager $where";

                $result = mysql_query($query, $con);

                echo "<table  border=\"1\" id=\"table-3\">";
                echo "<thead><tr><th>Clinic ID</th> <th>Name</th>
                                <th>Beds</th>       <th>Manager</th> <th>Manager start date</th><th></th><th></th>
                </tr></thead>";
//Επεξεργασία και προβολή των αποτελεσμάτων
                while($row = mysql_fetch_array($result))
                {
                        $Name = $row['Name'];
                        $BedsNo = $row['BedsNo'];
                        $Manager = $row['doc_name'];
                        $dateStart = $row['dateStart'];
                        $id = $row['idClinic'];

                        echo "<tr><td>$id</td><td>$Name</td><td>$BedsNo</td>
                                        <td>$Manager</td><td>$dateStart</td><td><a
href='editClinicForm.php?clinic_id=$id'>edit</a></td>
                                                                                <td><a
href='editClinic.php?idClinic=$id&action=Delete'>delete</a></td></tr>";
                        }
                echo "</table>";
   mysql_close($con);
?>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

24. Φόρμα δημιουργίας νέας ασθένειας(front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
  <div id="form_data">
<form action="addDisease_newRecord.php" method="post">
<fieldset>
<legend align="center" color="black" font="14px"> <b>New Disease Information</b> </legend><br>

<div>
        <label>Name:</label> <input type="text" name="Name" id="Name" required/>
</div>

<p class="buttons">
                <input type="submit">
</p>
</fieldset>
</form>
```

```
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

25. Δημιουργία νέας ασθένεια(back-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
<form>
  <fieldset>
<legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

<?php

        $con = mysql_connect('localhost','root', '');
    if (!$con)
    {
            die('Could not connect: ' . mysql_error());
        }

    mysql_select_db("hospital", $con);

    $name = $_POST['Name'];
//Query για εισαγωγή της ασθένειας στην βάση
    $sql = "INSERT INTO diseases (Name) VALUES ('$name')";

   if (!mysql_query($sql,$con))
        {
                die('Error: ' . mysql_error());
        }
//Ανάκτηση του id της δημιουργημένης ασθένειας
   $idDisease = mysql_insert_id();

mysql_close($con);
        echo "<p align='center'>Disease [$name] was succesfully added in the system. <br><a
href='editDiseaseForm.php?disease_id=$idDisease'>Edit Disease</a></p>";

        ?>
        <!-- InstanceEndEditable -->
  </section><!-- end of #main content -->
```

26. Φόρμα επεξεργασίας ασθένειας(front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
// PHP Block για την ανάκτηση των στοιχείων της ασθένειας προς επεξεργασία
<?php
            $disease_id=$_GET["disease_id"];

            $con = mysql_connect('localhost','root', '');
    if (!$con)
            {
                die('Could not connect: ' . mysql_error());
            }

            mysql_select_db("hospital", $con);

// Query για την ανάκτηση των στοιχείων της ασθένειας από την βάση.
            $query =  "SELECT * FROM diseases WHERE idDisease=".$disease_id;
            $result = mysql_query($query, $con);

            $row = mysql_fetch_array($result);
```

```
                    $name = $row['Name'];
                    $idDisease = $row['idDisease'];

?>

<div id="form_data">
        <form action="editDisease.php" method="post">
<fieldset>
<legend align="center" color="black" font="14px"> <b>Exam Information</b> </legend><br>

<div>
        <input id="idDisease" type="hidden" name="idDisease" value="<?php echo $idDisease ?>" required/>
</div>

<div>
        <label>Name:</label> <input type="text" name="Name" value="<?php echo $name ?>" required/>
</div>

<div>
   <table class="buttons" cellspacing="5">
   <tr>
     <td><input id="delete" name="action" value="Delete" type="submit" onClick="return confirm(
 'This will delete all data related to this disease. Are you sure you want to continue?');"/></td>
                    <td><input id="update" name="action" value="Update" type="submit"/></td>
                </tr>
            </table>
                                                    </div>
          </fieldset>
        </form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

27. Επεξεργασία ασθένειας(back-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
<form>
   <fieldset>
<legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

<?php

        $con = mysqli_connect('localhost','root', '');

        if (!$con)
        {
                echo '<p>Could not connect: ' . mysqli_error() . '</p>';
        }

    mysqli_select_db($con, "hospital");

        $action = isset($_POST['action']) ? $_POST['action'] : $_GET['action'];

        $disease_id = isset($_POST['idDisease']) ? $_POST['idDisease'] :  $_GET['disease_id'];

//1. Block για την διαγραφή ασθένειας
if($action == 'Delete'){
                mysqli_autocommit($con, false);

                $query4 = "DELETE FROM diseases WHERE idDisease=$disease_id";

                $result = mysqli_query($con, $query4);
```

```
                        if($result){

                                echo "<p align='center'>Disease was succesfully deleted from the system!</p>";
                        }else{
                                $query = "SELECT * FROM diseases WHERE idDisease=$disease_id";
                                $r= mysqli_query($con, $query);
                                $row = mysqli_fetch_array($r);
                                $name = $row['Name'];
                                echo "<p align='center'>Disease [$name] is conntected to a reagent.
                        <br>You will have to delete this first.
                        <br><a href='editDiseaseForm.php?disease_id=$disease_id'>Back to Disease</a></p>";
}


//2. Block για την ανανέωση στοιχείων μιας ασθένειας
}else if($action == 'Update'){

                        $name = $_POST['Name'];

                        $sql = "UPDATE diseases SET Name='$name' WHERE idDisease=$disease_id";


                        if (!mysqli_query($con, $sql))
                        {
                                echo $sql;
                                echo "<p align='center'>Error: ".mysqli_error()."!<br>Please try again.</p>";
                        }else{
                                echo "<p align='center'>Disease [$name] details were succesfully updated!<br><a
href='editDiseaseForm.php?disease_id=$disease_id'>Back to Disease</a></p>";
                        }
                }
                mysqli_close($con);
?>
                </fieldset>
                </form>
  </div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

28. Φόρμα εύρεσης ασθένειας (front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
        <form action="findDisease.php" method="post">
<fieldset>
<legend align="center" color="black" font="14px"> <b>Find Disease</b> </legend><br>

<div>
//Δήλωση drop down list με τις διαθέσιμες επιλογές αναζήτησης μιας ασθένειας.
    <label>Search by:</label>
    <select name="Criterion" id="Criterion">
        <option value="Name">Name</option>
    </select>

    <input type="text" name="Crit_value" id="Crit_value" required/>
</div>

<p class="submit"  align="center">
        <input type="submit" value="Search">
</p>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
```

```
</section><!-- end of #main content -->
```

29. Εύρεση/Προβολή ασθενειών

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
// php block για την αναζήτηση και παρουσίαση ασθενειών.
<?php

        $con = mysql_connect('localhost','root', '');
    if (!$con)
        {
            die('Could not connect: ' . mysql_error());
        }

    mysql_select_db("hospital", $con);

    $criterion = $_POST['Criterion'];
        $value = $_POST['Crit_value'];
        $query = "SELECT * FROM diseases WHERE ";

        switch($criterion)
        {
            case "Name":
                $query = $query."Name LIKE '%$value%'";
                break;

        }
        $result = mysql_query($query, $con);

        echo "<h2 align=\"center\">Diseases Search Results</h2>";
        echo "<h4 align=\"center\">($criterion='$value')</h4>";

        echo "<table  border=\"1\" id=\"table-3\">";
        echo "<thead><tr><th>Disease ID</th> <th>Name</th>
                    <th></th><th></th>
                </tr></thead>";


        if(!$result)
        {
            die('Error: ' . mysql_error());
        }
        while($row = mysql_fetch_array($result))
        {
            $Name = $row['Name'];
            $idDisease = $row['idDisease'];

            echo "<tr><td>$idDisease</td><td>$Name</td>
                    <td><a href='editDiseaseForm.php?disease_id=$idDisease'>edit</a></td>

                                                                <td><a
href='editDisease.php?disease_id=$idDisease&action=Delete'>delete</a></td></tr>";
            }
            echo "</table>";
     mysql_close($con);
?>

<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

30. Φόρμα δημιουργίας νέου αντιδραστηρίου(front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
```

```
<div id="form_data">
 <form action="addReagent_newRecord.php" method="post">
            <fieldset>
<legend align="center" color="black" font="14px"> <b>New Reagent Information</b> </legend><br>

<div>
  <label>Name:</label>
   <input type="text" name="Name" id="Name" required/>
</div>

<div>
   <label>Price:</label>
   <input type="numeric" name="Price" id="Price" required/>

<div>
            <label>Disease:</label>
<?php
            $query = "SELECT d.idDisease, d.Name FROM diseases d
                        LEFT OUTER JOIN reagents r ON d.idDisease = r.idDisease WHERE r.idDisease IS NULL";

fill_dropdown_list("idDisease", "Name", "idDisease", $query, 1);
?>
</div>

<p class="buttons">
            <input type="submit">
</p>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

31. Δημιουργίας νέου αντιδραστήριου(back-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
  <div id="form_data">
<form>
<fieldset>
            <legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

<?php

            $con = mysql_connect('localhost','root', '');
    if (!$con)
            {
                        die('Could not connect: ' . mysql_error());
            }

    mysql_select_db("hospital", $con);

    $name = $_POST['Name'];
            $price = $_POST['Price'];
            $idDisease = $_POST['idDisease'];

    $sql = "INSERT INTO reagents (ReagentName, Price, idDisease) VALUES ('$name', $price, $idDisease)";

    if (!mysql_query($sql,$con))
            {
                        die('Error: ' . mysql_error());
            }

    $reagent_id = mysql_insert_id();

            echo "<p align='center'>Reagent [$name] successfully added in the system!
```

```
                    <br><a href='editReagentForm.php?reagent_id=$reagent_id'>Edit Reagent</a></p>";

    mysql_close($con);
?>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

32. Φόρμα επεξεργασίας αντιδραστήριου(front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<?php
        $reagent_id=$_GET["reagent_id"];

        $con = mysql_connect('localhost','root', '');
    if (!$con)
        {
            die('Could not connect: ' . mysql_error());
        }

        mysql_select_db("hospital", $con);

        $query =  "SELECT * FROM reagents WHERE idReagent=".$reagent_id;
        $result = mysql_query($query, $con);

        $row = mysql_fetch_array($result);

        $idReagent = $row['idReagent'];
        $price = $row['Price'];
        $idDisease = $row['idDisease'];
        $reagentName =$row['ReagentName'];

?>

<div id="form_data">
        <form action="editReagent.php" method="post">
         <fieldset>
        <legend align="center" color="black" font="14px"> <b>Exam Information</b> </legend><br>

<div>
        <input id='idReagent' type="hidden" name="idReagent" value="<?php echo $idReagent ?>" required/>
</div>

<div>
        <label>Name:</label> <input type="text" name="ReagentName" value="<?php echo $reagentName ?>"
required/>
</div>

<div>
        <label>Price:</label> <input type="numeric" name="Price" value="<?php echo $price ?>" required/>
</div>

<div>
        <label>Disease:</label>
<?php
        $query = "SELECT idDisease, Name FROM diseases ORDER BY Name";
        fill_dropdown_list("idDisease", "Name", "idDisease", $query, 0);
?>
</div>

<div>
   <table class="buttons" cellspacing="5">
    <tr><td><input id="delete" name="action" value="Delete" type="submit" onClick="return confirm(
 'This will delete all data related to this reagent. Are you sure you want to continue?');"/></td>
    <td><input id="update" name="action" value="Update" type="submit"/></td>
 </tr>
```

```
</table>
</div>
</fieldset>
                    </form>
            </div>
            <!-- InstanceEndEditable -->
    </section><!-- end of #main content -->
            <footer>
                    <section id="footer-area">
                      <p align="center"> Copyright &copy; 2012 <a href="#">heraklionhospital.com</a>. All Rights
Reserved</p>
                </section><!-- end of footer-area -->
            </footer>

</div><!-- #container -->

<!-- InstanceBeginEditable name="JavaScript Code" -->
<script type="text/javascript">
//Javascript μέθοδος για τον καθορισμό default τιμής στο dropdown list.
window.onload = function () {
            //Set default values to dropdown lists
            var disease = "<?= $idDisease ?>";

            if(disease > 0){
                    document.getElementById('idDisease').value =  disease;
            }
}
```

33. Επεξεργασία αντιδραστήριου (back-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
<form>
   <fieldset>
            <legend align="left" color="black" font="14px"> <b>Message</b> </legend><br>

<?php

            $con = mysqli_connect('localhost','root', '');

            if (!$con)
            {
                    echo '<p>Could not connect: ' . mysqli_error() . '</p>';
            }

     mysqli_select_db($con, "hospital");

            $action = isset($_POST['action']) ? $_POST['action'] : $_GET['action'];

            $reagent_id = isset($_POST['idReagent']) ? $_POST['idReagent'] :  $_GET['reagent_id'];

// 1. Block για την διαγραφή ενός αντιδραστηρίου
            if($action == 'Delete'){
                    $query2 = "SELECT * FROM reagents WHERE  idReagent = $reagent_id";
                    $rr = mysqli_query($con, $query2);
                    $row = mysqli_fetch_array($rr);
                    $name = $row['ReagentName'];

                    $query = "SELECT * FROM exams WHERE idReagent = $reagent_id";
                    $r = mysqli_query($con, $query);
                    if(mysqli_fetch_array($r)!= NULL)
                    {
                            echo "<p align='center'>Reagent [$name] is used in an exam.<br>You will have to delete this
first.
                            <br><a href='editReagentForm.php?reagent_id=$reagent_id'>Back to Reagent</a></p>";
```

```
                    }
                    else
                    {
                            $query4 = "DELETE FROM reagents WHERE idReagent=$reagent_id";
                            $result = mysqli_query($con, $query4);

                            if($result){
                                    echo "<p align='center'>Reagent [$name] was succesfully deleted from the
system.</p>";
                                    }
                            }

// 2. Block για την ανανέωση των στοιχείων ενός αντιδραστηρίου
}else if($action == 'Update'){
                    $name = $_POST['ReagentName'];
                    $idDisease = $_POST['idDisease'];
                    $price = $_POST['Price'];

                    $sql = "UPDATE reagents
                                            SET ReagentName = '$name', Price = '$price', idDisease = '$idDisease'
                                            WHERE idReagent = ".$reagent_id;


                    if (!mysqli_query($con, $sql))
                    {
                            echo "<p align='center'>Error: ".mysqli_error()."!<br>Please try again.</p>";
                    }else{
                            echo "<p align='center'>Reagent [$name] details were succesfully updated!
                                    <br><a href='editReagentForm.php?reagent_id=$reagent_id'>Back to
Reagent</a></p>";
                    }
            }
            mysqli_close($con);
?>
</fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

34. Φόρμα εύρεσης αντιδραστηρίου (front-end)

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
<div id="form_data">
            <form action="findReagent.php" method="post">
                    <fieldset>
                    <legend align="center" color="black" font="14px"> <b>Find Reagent</b> </legend><br>
//Δήλωση drop down list με τις διαθέσιμες επιλογές αναζήτησης μιας ασθένειας.
<div>
    <label>Search by:</label>
     <select name="Criterion" id="Criterion">
          <option value="Name">Name</option>
          <option value="Disease">Disease</option>
     </select>
      <input type="text" name="Crit_value" id="Crit_value" required/>
</div>

<p class="submit"  align="center">
            <input type="submit" value="Search">
                                                    </p>
                                            </fieldset>
</form>
</div>
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```

35. Εύρεση/Προβολή αντιδραστήριων

```
<section id="main"><!-- #main content  -->
<!-- InstanceBeginEditable name="main" -->
// php block για την αναζήτηση και παρουσίαση αντιδραστηρίων.
<?php

        $con = mysql_connect('localhost','root', '');
    if (!$con)
        {
            die('Could not connect: ' . mysql_error());
        }

    mysql_select_db("hospital", $con);

    $criterion = $_POST['Criterion'];
        $value = $_POST['Crit_value'];

        $query = "SELECT r.idReagent AS idReagent, r.ReagentName, r.Price, d.Name AS dName
        FROM reagents r
            INNER JOIN diseases d ON r.idDisease = d.idDisease
        WHERE ";

// Ανάλογα με το κριτήριο που επέλεξε ο χρήστης για την αναζήτηση ενός αντιδραστηρίου, δημιουργείται δυναμικά το query αναζήτησης
                    switch($criterion)
                    {
                        case "Name":
                            $query = $query."r.ReagentName LIKE '%$value%'";
                            break;

                        case "Disease":
                            $query = $query."d.Name LIKE '%$value%'";
                            break;

                    }
                    $result = mysql_query($query, $con);
                        echo "<h2 align=\"center\">Search Results</h2>";
                    echo "<table border=\"1\" id=\"table-3\">";
                    echo "<thead><tr><th>Name</th><th>Price</th> <th>Disease</th>
                            <th></th><th></th>
                     </tr></thead>";

                    if(!$result)
                    {
                        die('Error: ' . mysql_error());
                    }
//Επεξεργασία και προβολή των αποτελεσμάτων
                    while($row = mysql_fetch_array($result))
                    {
                        $Name = $row['ReagentName'];
                        $Disease = $row['dName'];
                        $price = $row['Price'];
                        $idReagent = $row['idReagent'];

                        echo "<tr><td>$Name</td><td>$price</td><td>$Disease</td>
                              <td><a href='editReagentForm.php?reagent_id=$idReagent'>edit</a></td>

                                                                    <td><a
href='editReagent.php?reagent_id=$idReagent&action=Delete'>delete</a></tr>";
                    }
                    echo "</table>";
      mysql_close($con);
        ?>
```

```
<!-- InstanceEndEditable -->
</section><!-- end of #main content -->
```