Ατομική Διπλωματική Εργασία

ΑΣΦΑΛΕΙΑ ΗΛΕΚΤΡΟΝΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ΥΓΕΙΑΣ

Θεοχάρης Ψάλιος

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2017

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ασφάλεια Ηλεκτρονικού Συστήματος Υγείας

Θεοχάρης Ψάλιος

Επιβλέπων Καθηγητής

Δρ. Κωνσταντίνος Παττίχης

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου.

Μάιος 2017

Ευχαριστίες

Θα ήθελα να ευχαριστήσω ιδιαίτερα τον Επιβλέποντα Καθηγητή μου κ. Κωνσταντίνο Παττίχη για την εμπιστοσύνη που μου έδειξε ώστε να φέρω εις πέρας την παρούσα διπλωματική εργασία. Τον ευχαριστώ, επίσης, για την καθοδήγησή του κατά τη συγγραφή της εργασίας όπως και για τις πολύτιμες συμβουλές που μου παρείχε.

Ευχαριστίες θα ήθελα να εκφράσω και στον Επίκουρο Καθηγητή του τμήματός μας, κ. Ηλία Αθανασόπουλο, για την καθοδήγησή του στα θέματα ασφάλειας, με τα οποία ασχολήθηκε η διπλωματική μου εργασία.

Θα ήθελα, ακόμα, να εκφράσω τις ευχαριστίες μου στον κύριο Ιωάννη Κωνσταντίνου και κύριο Ζήνωνα Αντωνίου του εργαστηρίου e-health του Πανεπιστημίου μας για τις ώρες που θυσίασαν αφιλοκερδώς για να με βοηθήσουν στην επίλυση τεχνικών προβλημάτων που αντιμετώπισα κατά καιρούς.

Τέλος, θα ήταν αχαριστία μου να μην εκφράσω τις ευχαριστίες μου στα άτομα που ήταν δίπλα μου και με στήριζαν από την πρώτη μέρα φοίτησής μου στο Πανεπιστήμιο Κύπρου. Ευχαριστώ πολύ την οικογένειά μου, που ήταν πάντα στο πλάι μου σε κάθε καλή και δύσκολη στιγμή. Ευχαριστώ τον πνευματικό μου πατέρα, π. Θεοχάρη, που με τις ευχές και προσευχές του στηρίζει και ευλογεί κάθε μου βήμα. Τέλος, ευχαριστώ όλους τους φίλους και συμφοιτητές που κατά τη διάρκεια των τεσσάρων αυτών χρόνων μου χάρισαν τόσο όμορφες και ευχάριστες στιγμές.

Περίληψη

Με το πέρας των χρόνων, η ανάπτυξη της τεχνολογίας δίνει στον άνθρωπο όλο και περισσότερες δυνατότητες για να επιτύχει τους σκοπούς του. Νέα συστήματα για διεκπεραίωση διαφόρων εργασιών αναπτύσσονται χρόνο με το χρόνο και αυτό είναι κάτι ευχάριστο. Το δυσάρεστο είναι ότι υπάρχουν άνθρωποι, οι οποίοι προσπαθούν με διαφόρους τρόπους να θέσουν σε δυσλειτουργία τα προαναφερθέντα συστήματ. Έτσι, κρίνεται απαραίτητη η υλοποίηση ασφάλειας για τα συστήματα αυτά.

Η παρούσα διπλωματική εργασία έχει σαν στόχο να μελετήσει τα θέματα ασφάλειας που εφαρμόζονται σε συστήματα ανά τον κόσμο και να κάνει χρήση αυτών. Πιο συγκεκριμένα, γίνεται χρήση ενός ήδη υπάρχοντος διαδικτυακού συστήματος καταγραφής ασθενών με εμφράγματα μυοκαρδίου, στο οποίο προστίθενται επιπλέον κομμάτια ώστε να επιτευχθεί η ασφάλεια των δεδομένων του συστήματος.

Αναλυτικότερα, το σύστημα επεκτείνεται με την προσθήκη μιας επιπλέον σελίδας, στην οποία οι γιατροί θα μπορούν να κάνουν αίτηση για να τους επιτραπεί η πρόσβαση στο σύστημα. Η σελίδα αποτρέπει τη μαζική δημιουργία αιτήσεων για δημιουργία νέων χρηστών όπως, επίσης, δεν επιτρέπει στο νέο χρήστη να εισέλθει στο σύστημα μέχρις ότου εγκριθεί από τον διαχειριστή.

Επιπλέον, γίνεται εφαρμογή κρυπτογράφησης στις πληροφορίες των ασθενών που είναι καταχωρημένες στη βάση δεδομένων και τίθεται σε λειτουργία η τήρηση αρχείου καταγραφής ενεργειών (audit trail) για τους χρήστες με τη βοήθεια του OpenATNA.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Γενική Εισαγωγή

```
do {
    secure_system();
} while (1=1);
```

"Security is, I would say, our top priority because for all the exciting things you will be able to do with computers - organizing your lives, staying in touch with people, being creative - if we don't solve these security problems, then people will hold back." - Bill Gates

Μικρή αναδρομή στην Ιστορία της Ασφάλειας Πληροφορικών Συστημάτων και Δεδομένων

Η ασφάλεια δεδομένων είχε την πρώτη της εφαρμογή την περίοδο του Β' Παγκοσμίου Πολέμου από ομάδες στο στρατό, οι οποίες χρησιμοποιούσαν εφαρμογές της κρυπτογραφίας για να ανταλλάζουν πληροφορίες μεταξύ τους χωρίς να γίνονται αντιληπτές από τα εχθρικά στρατόπεδα.

Λέγοντας "Ασφάλεια δεδομένων" και κατ' επέκταση "Ασφάλεια Συστημάτων", εννοούμε τις ενέργειες που λαμβάνει ένα πρόσωπο ή μια ομάδα ανθρώπων ώστε να διαφυλάξει την ακεραιότητα (integrity), διαθεσιμότητα (availability) και εμπιστευτικότητα (confidentiality) των δεδομένων του συστήματός του. Αυτό

επιτυγχάνεται με την προστασία των πληροφορικών συστημάτων από κλοπή ή ζημιά του υλικού, λογισμικού ή πληροφοριών που αυτά κρατούν, όπως επίσης και την προστασία από αποδιοργάνωση και σύγχυση των υπηρεσιών που αυτά παρέχουν. [1]

Κατά τη δεκαετία του 1980 ξεκίνησε να υπάρχει μια αύξηση στη χρήση των ηλεκτρονικών υπολογιστών και εν συνεχεία στις αρχές της δεκαετίας του 1990 άρχισε να χρησιμοποιείται και να γίνεται ευρέως γνωστό και το Διαδίκτυο.

Αρχικά, το Διαδίκτυο παρείχε εφαρμογές για παιχνίδια, ήχο και βίντεο, τα οποία έλκυσαν αρκετούς χρήστες ηλεκτρονικών υπολογιστών.

Με την αύξηση της ζήτησης του Διαδικτύου, πολλές εταιρείες όπως ξενοδοχεία και αεροπορικές εταιρείες ξεκίνησαν να παρέχουν ηλεκτρονικές κρατήσεις (δωματίων και) θέσεων. Αυτό, όμως, έδωσε ώθηση στο να ξεκινήσουν οι πρώτες επιθέσεις κατά των χρηστών του Διαδικτύου, με σκοπό την κλοπή των πιστοποιητικών τους δεδομένων (credentials) και την απόκτηση πρόσβασης στους λογαριασμούς τους.

Έτσι, κρίθηκε επιτακτική η ανάγκη ανάπτυξης συστημάτων τα οποία να παρέχουν την απαιτούμενη ασφάλεια στα δεδομένα των χρηστών.

Ασφάλεια συστημάτων και δεδομένων σήμερα

Με την πρόοδο και ανάπτυξη της τεχνολογίας έχουν προσαρμοστεί και οι επιθέσεις προς τα συστήματα πληροφορικής, έτσι πιο σύνθετοι και έξυπνοι αλγόριθμοι χρησιμοποιούνται ώστε να ελαχιστοποιούνται και να αποτρέπονται (όπου είναι δυνατόν) οι κακόβουλες αυτές ενέργειες.

Ασφάλεια δεδομένων ως κλάδος της Πληροφορικής

Στον κλάδο της Πληροφορικής, ο τομέας της ασφάλειας δεδομένων αγκαλιάζει ένα μεγάλο εύρος τεχνολογιών, μιας και ένα σύστημα μπορεί να δεχθεί επιθέσεις από διάφορες πηγές. Έτσι, ο τομέας αυτός ασχολείται από το επίπεδο του δικτύου μέχρι και τους χρήστες του ιδίου του συστήματος. Αρμοδιότητά του είναι να εξασφαλίσει την ασφάλεια του συστήματος τόσο από εξωτερικές επιθέσεις, που προέρχονται από το δίκτυο, όσο και από εσωτερικές, που προέρχονται από τους ιδίους τους χρήστες του συστήματος,

10 πιο συχνά ρίσκα ασφάλειας σε εφαρμογές ιστού:

1. Injection
2. Broken Authentication and Session Management
3. Cross-Site Scripting (XSS)
4. Insecure Direct Object References
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Missing Function Level Access Control
8. Cross-Site Request Forgery (CSRF)
9. Using Components with Known Vulnerabilities
10. Unvalidated Redirects and Forwards [2]

Υπάρχει λύση για το πρόβλημα της ασφάλειας;

Μέχρι στιγμής έχουν αναπτυχθεί πολλοί αλγόριθμοι και πολλές τεχνικές οι οποίες αποτρέπουν πολλές από τις κακόβουλες ενέργειες. Συχνά, όμως, εμφανίζονται καινούριοι τρόποι επιθέσεων με διαφορετική φιλοσοφία ο κάθε ένας. Έτσι, είναι επιτακτικό να υπάρχει μια επαγρύπνηση όσον αφορά τα συστήματα πληροφορικής και τις επιθέσεις που δέχονται, ώστε να αντιμετωπίζονται όσο το δυνατό πιο γρήγορα.

1.2 Στόχος Διπλωματικής Εργασίας

Ο στόχος αυτής της διπλωματικής εργασίας είναι να μελετήσει τις τεχνικές που χρησιμοποιούνται για την ασφάλεια των πληροφορικών συστημάτων και με τη χρήση αυτών να γίνει μια επέκταση προηγουμένων διπλωματικών εργασιών που έγιναν στο Πανεπιστήμιο Κύπρου και αφορούσαν κομμάτια της Ηλεκτρονικής Υγείας, ώστε να επαυξηθεί η ασφάλεια των δεδομένων που θα κρατούν.

Πιο συγκεκριμένα, θα μελετηθεί η διπλωματική εργασία του Φίλιππου Κινδύνη (έχει παραδοθεί το Μάιο του 2016), η οποία αφορούσε καρδιολογικό τμήμα σε νοσοκομείο, και στην οποία θα προστεθεί το κομμάτι της ασφάλειας ώστε να αποτρέπονται επιθέσεις που αφορούν την ασφάλεια των δεδομένων. Επιπλέον, θα γίνει προσθήκη επιπλέον λειτουργικότητα στην εν λόγω εργασία ώστε να προσαρμοστεί στις απαιτήσεις του νοσοκομείου, στο οποίο και θα παραδοθεί.

Η υλοποίηση θα γίνει με τη χρήση των τεχνολογιών του PyCharm, υποβοηθούμενο από τις τεχνολογίες του Django και Python, του χώρου εργασίας Eclipse για Java με τη βοήθεια του TomCat Server, ως επίσης, και της MySQL για τη δημιουργία και συντήρηση της βάσης δεδομένων.

Επίσης, θα ακολουθηθεί η τακτική που συστήνεται από το testing guide, το οποίο έχει συγγραφεί και εκδοθεί από την OWASP (Open Web Application Security Project) [3] ώστε να γίνει μια πιο επαρκής και σωστή ανάπτυξη και αξιολόγηση των μεθόδων ασφάλειας που θα υλοποιηθούν.

1.3 Δομή Διπλωματικής Εργασίας

Η διπλωματική εργασία θα ακολουθήσει την ακόλουθη δομή:

Κεφάλαιο 1. Γενική Εισαγωγή.

Στο πρώτο κεφάλαιο εκτείνεται μια αναφορά για το τι είναι η ασφάλεια πληροφορικών συστημάτων και των πληροφοριών που αυτά κρατούν, πότε έκανε την πρώτη της εμφάνιση καθώς επίσης και το πώς προέκυψε. Επιπλέον, γίνεται αναφορά στον κλάδο της Πληροφορικής που ασχολείται με τα θέματα ασφάλειας σήμερα, όπως επίσης και κάποιες κοινές επιθέσεις που γίνονται προς τα προαναφερθέντα συστήματα. Τέλος, παρατίθενται οι τεχνολογίες και οι οδηγοί ανάπτυξης και ελέγχου των τεχνολογιών ασφαλείας, των οποίων θα γίνει χρήση ώστε να ολοκληρωθεί επιτυχώς η παρούσα διπλωματική εργασία.

Κεφάλαιο 2. Περιγραφή προβλήματος και ανασκόπηση Βιβλιογραφίας.

Στο κεφάλαιο 2 αναλύονται σε περισσότερο βάθος τα θέματα που αναφέρθηκαν στην εισαγωγή και γίνεται μια αναλυτική ανασκόπηση της βιβλιογραφίας.

- Αδυναμίες πληροφορικών συστημάτων ως προς το θέμα της ασφάλειας των πληροφοριών και τρόποι επίλυσης και ενίσχυσης της ασφάλειας.
- Ανασκόπηση βιβλιογραφίας – Άλλα παρόμοια Συστήματα

Κεφάλαιο 3. Απαιτούμενη Γνώση και Τεχνολογίες

Σε αυτό το κεφάλαιο γίνεται εκτενής αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν ώστε να έρθει εις πέρας η παρούσα εργασία. Γίνεται αναφορά στις δυνατότητες που παρέχει η κάθε τεχνολογία και στα χαρακτηριστικά που έχει η κάθε μία.

Κεφάλαιο 4. Ανάλυση Απαιτήσεων και Προδιαγραφές Συστήματος

Στο κεφάλαιο αυτό γίνεται αναφορά στο στάδιο της Ανάλυσης Απαιτήσεων και στον τρόπο με τον οποίο ορίστηκαν οι προδιαγραφές που θα πρέπει να υποστηρίζονται από την εφαρμογή με το πέρας της παρούσας εργασίας.

Κεφάλαιο 5. Σχεδιασμός και Υλοποίηση

Το Κεφάλαιο 5 περιέχει όλες τις πληροφορίες που αφορούν την υλοποίηση της ασφάλειας στην εφαρμογή. Πιο συγκεκριμένα, γίνεται αναφορά στον τρόπο με τον οποίο επιτεύχθηκε η υλοποίηση, καθώς επίσης και το πώς λειτουργεί το κάθε επί μέρους κομμάτι της ασφάλειας. Επιπλέον, παρέχονται και φωτογραφίες από παραδείγματα εκτέλεσης της εφαρμογής.

Κεφάλαιο 6. Αξιολόγηση και Αξιοπιστία Συστήματος

Σε αυτό το κεφάλαιο παρουσιάζεται η αξιολόγηση της εφαρμογής όπως, επίσης, και το επίπεδο αξιοπιστίας που παρέχει. Πιο συγκεκριμένα, γίνεται αναφορά στο επίπεδο ασφαλείας του συστήματος μετά τη χρήση πρακτικών ασφαλείας και στο κατά πόσο επιτεύχθηκαν οι στόχοι που τέθηκαν στην αρχή.

Κεφάλαιο 7. Συμπεράσματα και Μελλοντική Εργασία

Στο Κεφάλαιο 7 γίνεται μια σύνοψη των όσων υλοποιήθηκαν στην παρούσα εργασία. Γίνεται, επίσης, καταγραφή εισηγήσεων για μελλοντική εργασία και ανάπτυξη στην παρούσα μελέτη και εφαρμογή.

Κεφάλαιο 2

Περιγραφή προβλήματος και ανασκόπηση Βιβλιογραφίας

2.1 Ασφάλεια Πληροφορικών Συστημάτων και Δεδομένων

Η παροχή ασφάλειας στα πληροφορικά συστήματα και τα δεδομένα που αυτά κρατούν είναι μια από τις πλέον βασικές αρμοδιότητες και ευθύνες ενός διαχειριστή ή/και ιδιοκτήτη τέτοιου συστήματος. Με την αύξηση των επιθέσεων και κρουσμάτων που δέχονται τα συστήματα πληροφορικής στις μέρες μας, επιστήμονες της Πληροφορικής ανά το παγκόσμιο κάνουν ό,τι καλύτερο μπορούν για να αποτρέψουν την κατάρρευση ή κατάληψη από μη αρμόδια άτομα του συστήματός τους. Πολλές είναι οι επιθέσεις που γίνονται κατ' αυτά τα συστήματα, πολλά είναι όμως και τα μέτρα που λαμβάνονται για αναχαίτισή τους.

2.1.1 Τύποι επιθέσεων προς Συστήματα Πληροφορικής

Ένας ορθός διαχωρισμός των επιθέσεων θα ήταν αυτός που αναφέρεται στη σελίδα της Wikipedia, ο οποίος διαχωρίζει τις επιθέσεις σε δύο κατηγορίες: τις παθητικές (passive) και τις ενεργητικές (active) [4]. Κατά τις παθητικές επιθέσεις, ο χρήστης που πρόκειται να επιτεθεί στο σύστημα έχει σαν μόνο στόχο να αντλήσει πληροφορίες από αυτό. Στις ενεργητικές επιθέσεις, όμως, σκοπός του χρήστη εκτός από την άντληση πληροφοριών (η οποία ενδεχομένως να μην τον ενδιαφέρει και ως εκ τούτου να μην αντλήσει κάποια δεδομένα) είναι να προξενήσει κάποια βλάβη στη λειτουργία του συστήματος. Οι δύο

αυτές κατηγορίες στη συνέχεια επιδέχονται ένα περαιτέρω διαχωρισμό, ο οποίος οδηγεί στην εξής κατηγοριοποίηση των επιθέσεων [4]:

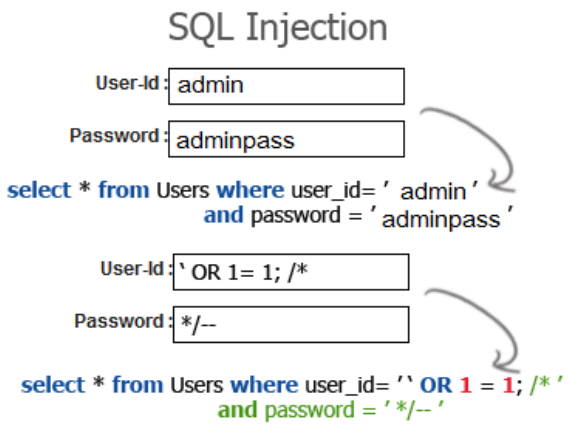
- Passive
    - Network
        - Wiretapping
        - Port scan
        - Idle scan
- Active
    - Denial-of-service attack
    - Spoofing
    - Network
        - Man in the middle
        - ARP poisoning
        - Ping flood
        - Ping of death
        - Smurf attack
    - Host
        - Buffer overflow
        - Heap overflow
        - Stack overflow
        - Format string attack

Η ομάδα της OWASP είναι άλλη μια οντότητα η οποία έχει κάνει εκτεταμένη έρευνα στον τομέα της ασφάλειας και η οποία έχει δημοσιεύσει ανά τακτά χρονικά διαστήματα (2004 [5], 2007 [6], 2010 [7], 2013 [2]) λίστες με τις 10 πιο συχνές επιθέσεις που επιδέχονται τα web application συστήματα πληροφορικής ανά τον κόσμο την παρούσα χρονολογία. Πιο κάτω αναφέρουμε τις 10 πιο συχνές επιθέσεις που καταγράφηκαν κατά τη χρονιά του 2013, η οποία είναι και η πιο πρόσφατη καταγραφή επιθέσεων από την ομάδα της OWASP:

10 πιο συχνές επιθέσεις σε εφαρμογές ιστού κατά το έτος 2013:

1. Injection
2. Broken Authentication and Session Management
3. Cross-Site Scripting (XSS)
4. Insecure Direct Object References
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Missing Function Level Access Control
8. Cross-Site Request Forgery (CSRF)
9. Using Components with Known Vulnerabilities
10. Unvalidated Redirects and Forwards [2]

Σε αυτό το σημείο θα μπορούσαμε να αναλύσουμε λίγο περισσότερο κάποιες από τις πιο πάνω επιθέσεις, τις πιο κοινές, ώστε να γίνει πιο αντιληπτός ο τρόπος με τον οποίο ενεργούν οι επιτεθέντες και να δούμε πώς ένα σύστημα μπορεί να αφήσει ανοιχτές «τρύπες» που να επιτρέπουν πρόσβαση σε αυτό στους εν λόγω χρήστες.



Σχήμα 2.1 - SQL Injection

Αρχικά, το *SQL Injection* αναφέρεται στη δυνατότητα που έχει ένας χρήστης να στείλει μέσω μιας ηλεκτρονικής φόρμας δικές του εντολές στη βάση δεδομένων που τρέχει στον application server. Μπορούμε να πάρουμε σαν παράδειγμα τη φόρμα που συμπληρώνει ο διαχειριστής του συστήματος. Το σύστημα ζητάει από το διαχειριστή να κάνει εισαγωγή του username και password του και με την επιτυχή τους εισαγωγή ο διαχειριστής αποκτά πρόσβαση στο

σύστημα. Αν η εντολή την οποία εκτελεί η βάση δεδομένων είναι αυτή που φαίνεται στη διπλανή εικόνα, δηλαδή η "select * from Users where user_id='admin' and password='adminpass'", τότε με την χρήση του 1=1 που συνεπάγεται True και με την προσθήκη σχολίων στο password δίνεται η δυνατότητα σε οποιοδήποτε μη εξουσιοδοτημένο πρόσωπο να εισέλθει στο σύστημα και μάλιστα με προνόμια διαχειριστή (σε αυτή την περίπτωση).

Κατά το *Broken Authentication and Session Management* κάποιος χρήστης έχει τη δυνατότητα να αποκτήσει πρόσβαση σε πληροφορίες του συστήματος μέσω του λογαριασμού ενός άλλου χρήστη. Αυτό μπορεί να επιτευχθεί με τους εξής τρόπους:

- Κάποιος χρήστης συνδέεται μέσω ενός δημοσίου ηλεκτρονικού υπολογιστή στο λογαριασμό του και με το πέρας της εργασίας του δεν κάνει logout παρά μόνο κλείνει τον browser. Φεύγοντας, κάποιος άλλος χρήστης θα μπορεί να μπει στο λογαριασμό του συγκεκριμένου χρήστη μιας και ο πρώτος δεν έχει αποσυνδεθεί.

- Στις περιπτώσεις που κάποιες ιστοσελίδες προσθέτουν session IDs στα URLs τους, αν κάποιο από τα URLs αυτά αποσταλεί από το χρήστη που είναι συνδεδεμένος εκείνη τη στιγμή σε άλλα πρόσωπα, τότε αυτοί θα έχουν πρόσβαση στις πληροφορίες του λογαριασμού του συνδεδεμένου χρήστη.

- Μέσω τροποποίησης του URL της ιστοσελίδας, στο οποίο αν προστεθεί επιπλέον κείμενο, να δίνεται η δυνατότητα πρόσβασης σε άλλες σελίδες του ιστοχώρου.

- Μέσω Brute Force επιθέσεων προς το σύστημα, όταν δεν υπάρχει περιορισμός στον αριθμό των λάθος εισαγωγών κωδικού.

Τρίτο έρχεται το *Cross Site Scripting (XSS)*. Αυτού του τύπου οι επιθέσεις εκμεταλλεύονται την εμπιστοσύνη που έχει ο browser του χρήστη προς τον server. Κακόβουλα κομμάτια κώδικα μπορεί να εκτελεστούν λόγω της προαναφερθείσας εμπιστοσύνης. Έτσι, παρατηρείται η προσθήκη ανεπιθύμητου κώδικα στην ιστοσελίδα, είτε μέσω HTML είτε μέσω Javascript. Ο κώδικας αυτός είναι δυνατό να προστεθεί μέσω φορμών ή άλλων παρόμοιων HTML tags που βρίσκονται στην ιστοσελίδα, όπως είναι τα <img src…>, <iframe …>, <bgsound scr…> κ.ά. [9]. Για σκοπούς καλύτερης κατανόησης, καλό θα ήταν να κάνουμε χρήση ενός απλού παραδείγματος. Ας υποθέσουμε το URL *http://hello.com?user=John* στην οποία ιστοσελίδα υπάρχει ο ακόλουθος κώδικας Javas-

cript: *<h1>Hello <%=request.getParameter("name")%></h1>*. Ο κώδικας αυτός μετατρέπει τον HTML κώδικα της ιστοσελίδας σε * Hello John *, όπου το "John" έχει παραληφθεί από το URL που δώσαμε πιο πάνω. Όταν ο HTML κώδικας γίνει render θα έχουμε στην οθόνη μας το κείμενο: "Hello **John**". Όταν, τώρα, στο URL της ιστοσελίδας προστεθεί κάποιο επιπλέον κείμενο, εκεί θα εμφανιστεί το πρόβλημα. Στο πιο πάνω παράδειγμα, ας υποθέσουμε το εξής URL: http://hello.com?user=**friend**

**
<form method="post"**

action="maliciousSite.com/login">

Login: <input type="text"

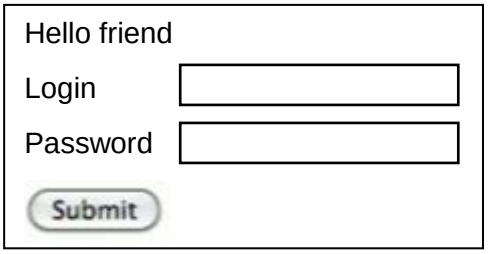
**name="username">
**

Password: <input type="Password"

name="Password">

<input type=="submit" value="Submit"/></form>

Με την προσθήκη του κώδικα αυτού, η πιο πάνω ιστοσελίδα έχει μετατραπεί σε μια εντελώς διαφορετική ιστοσελίδα:



Σχήμα 2.2 - Cross Site Scripting

Συνήθως, χρήση του Cross-site scripting γίνεται για την επίτευξη ενός ή περισσοτέρων στόχων από τους πιο κάτω:

1. Κλοπή session cookies (<script>alert(document.cookie)</script>)
2. Για να γίνει redirection προς μια άλλη ιστοσελίδα (<script>window.location = http://maliciousSite.com/</script>)
3. Τροποποίηση του κώδικα της ιστοσελίδας ώστε να παρουσιαστεί κάτι άλλο στον χρήστη

Θα ήταν καλό, μαζί με τις παραπάνω επιθέσεις, να αναφερθούμε και στο *Cross Site Request Forgery (CSRF, καλείται επίσης και C-Surf)*. Το CSRF είναι ένα είδος επίθεσης

κατά το οποίο ένα κακόβουλο web site, blog ή πρόγραμμα προκαλεί τον web browser του χρήστη να προβεί σε ανεπιθύμητες ενέργειες σε ένα διαδικτυακό τόπο στον οποίο την παρούσα στιγμή ο χρήστης είναι συνδεδεμένος (έχει γίνει authenticated). Μπορεί για παράδειγμα, αν η επίθεση CSRF είναι επιτυχής, να επιτευχθεί η μεταφορά χρημάτων από το λογαριασμό του χρήστη, η αλλαγή κωδικού του, η αγορά ενός αντικειμένου κ.ά. Ουσιαστικά, τέτοιου είδους επιθέσεις χρησιμοποιούνται από attackers για να προκαλέσουν το σύστημα-στόχο να εκτελέσει μια λειτουργία μέσω του web browser του χρήστη χωρίς τη γνώση και έγκριση του χρήστη. Οι επιπτώσεις που μπορεί να έχει μια τέτοιου είδους επίθεση εξαρτάται από τα δικαιώματα που έχει ο χρήστης, ο οποίος δέχτηκε την επίθεση. Αν ο χρήστης είναι απλός χρήστης στο σύστημα, τότε ο attacker θα μπορεί να έχει πρόσβαση μόνο στα δεδομένα του χρήστη. Αν, όμως, ο χρήστης που δέχτηκε την επίθεση είναι ο διαχειριστής του συστήματος τότε ο attacker θα μπορεί να έχει πρόσβαση στα δεδομένα ολόκληρης της εφαρμογής. [13]

Τέλος, θα αναφερθούμε στην *Έκθεση Ευαίσθητων Δεδομένων (Sensitive Data Exposure)*. Αυτό το κομμάτι ασφάλειας αφορά τα δεδομένα τα οποία βρίσκονται στη βάση δεδομένων μας και τα οποία αν πέσουν στα χέρια λάθος ανθρώπων υπάρχει κίνδυνος να βλαφτεί το πρόσωπο το οποίο αφορούν. Τέτοια δεδομένα μπορεί να είναι τα ονόματα και οι κωδικοί των χρηστών ενός συστήματος, αριθμοί πιστωτικών καρτών, ακόμα και τα session cookies. Είναι διάφοροι οι τρόποι με τους οποίους τα δεδομένα αυτά μπορούν να βρεθούν σε λάθος χέρια και κάποιοι από αυτούς είναι:

1. Η μη κρυπτογράφηση των δεδομένων αυτών εντός της βάσης ή η αυτόματη κρυπτογράφηση και αποκρυπτογράφηση των δεδομένων αυτών από την ίδια τη βάση.
2. Η μη χρήση ασφαλούς καναλιού για τη μεταφορά των δεδομένων αυτών.
3. Το αδύναμο hashing των δεδομένων

2.1.2 Τρόποι αντιμετώπισης

Για να προστατευτεί ένα υπολογιστικό σύστημα ή μια ηλεκτρονική εφαρμογή από επιθέσεις ιδίου ή παρόμοιου τύπου με αυτές που αναφέρθηκαν πιο πάνω χρειάζεται να ληφθούν κάποια μέτρα ώστε η συγγραφή του κώδικα να μην επιτρέπει την εμφάνισή τους. Σε αυτό το υποκεφάλαιο θα αναφέρουμε επιγραμματικά κάποιους τρόπους με τους οποίους μπορούμε να δημιουργήσουμε ένα επίπεδο προστασίας γύρω από το σύστημα ή την εφαρμογή μας ώστε να το κρατήσουμε σε ένα καλό επίπεδο ασφαλείας.

Αρχικά, θα πρέπει να γίνεται έλεγχος των δεδομένων εισόδου του χρήστη ώστε να κρατήσουμε τα μη αξιόπιστα δεδομένα μακριά από τα queries που δίνονται στη βάση δεδομένων. Κάνοντας αυτό μπορούμε να αποφύγουμε τον κίνδυνο για SQL Injection. Ο πιο ασφαλής τρόπος για να γίνει ο έλεγχος αυτός είναι η χρήση ενός ασφαλούς και παραμετροποιημένου API το οποίο να ελέγχει την είσοδο και να κάνει διαφυγή των ειδικών χαρακτήρων. Διάφορα APIs παρέχουν τη δυνατότητα για prepared statements [14], τα οποία είναι μια πολύ αποδοτική και ασφαλής προσέγγιση για αποφυγή του SQL Injection. Αν αυτή η λύση δεν είναι εφικτή τότε η αμέσως καλύτερη είναι να κάνουμε εμείς τη διαφυγή των ειδικών χαρακτήρων, ανάλογα με τον interpreter που χρησιμοποιούμε.

Κατά δεύτερον, καλό θα ήταν να γίνει μια προσεκτική συγγραφή του κώδικα της ιστοσελίδας ώστε ο κώδικας αυτός από μόνος του να αποτρέπει την εμφάνιση επιθέσεων Cross Site Scripting. Αυτό μπορεί να επιτευχθεί με παρόμοιο τρόπο όπως το SQL Injection, δηλαδή με τη διαφυγή των ειδικών χαρακτήρων και λέξεων που έχουν σχέση με το περιεχόμενο της HTML (όπως είναι το body, attribute, κώδικας JavaScript και CSS).

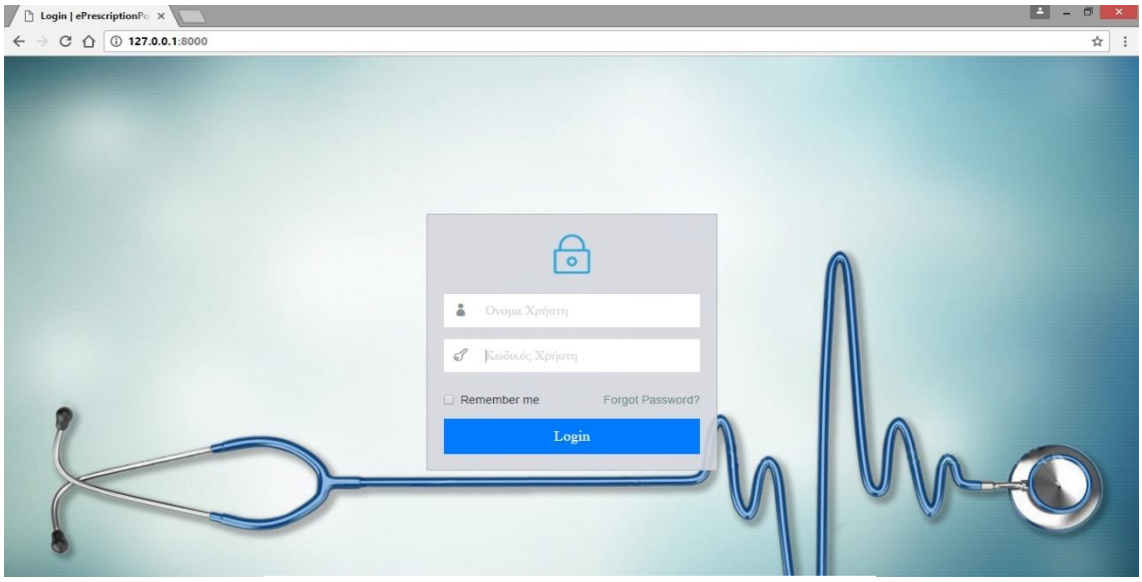
Όσον αφορά το Cross Site Request Forgery, η πιο ιδανική λύση θα ήταν να γίνει χρήση ενός CSRF token. Το CSRF token είναι ένα μοναδικό για κάθε χρήστη token, το οποίο στέλνεται με κάθε αίτημα του χρήστη και το οποίο δεικνύει την αυθεντικότητα του χρήστη.

Επιπλέον, όσον αφορά την ασφάλεια των δεδομένων στη βάση, θα πρέπει να υπάρχει μια καλή κωδικοποίησή τους ώστε σε περίπτωση που πέσουν σε λάθος χέρια να μην είναι κατανοητά και επομένως να μην μπορούν να αξιοποιηθούν. Όσον αφορά το κομμάτι αυτό, είναι πιο ασφαλής λύση να γίνεται hashing των δεδομένων παρά κρυπτογράφηση για τον εξής λόγο: Στην κρυπτογράφηση γίνεται χρήση ενός κλειδιού με τη γνώση του οποίου μπορεί να γίνει και κρυπτογράφηση και αποκρυπτογράφηση των δεδομένων. Αντιθέτως, το hashing είναι μια one-way διαδικασία στην οποία τα δεδομένα κωδικοποιούνται με ένα κλειδί αλλά το ίδιο κλειδί δεν μπορεί να χρησιμοποιηθεί για να αποκωδικοποιηθούν.

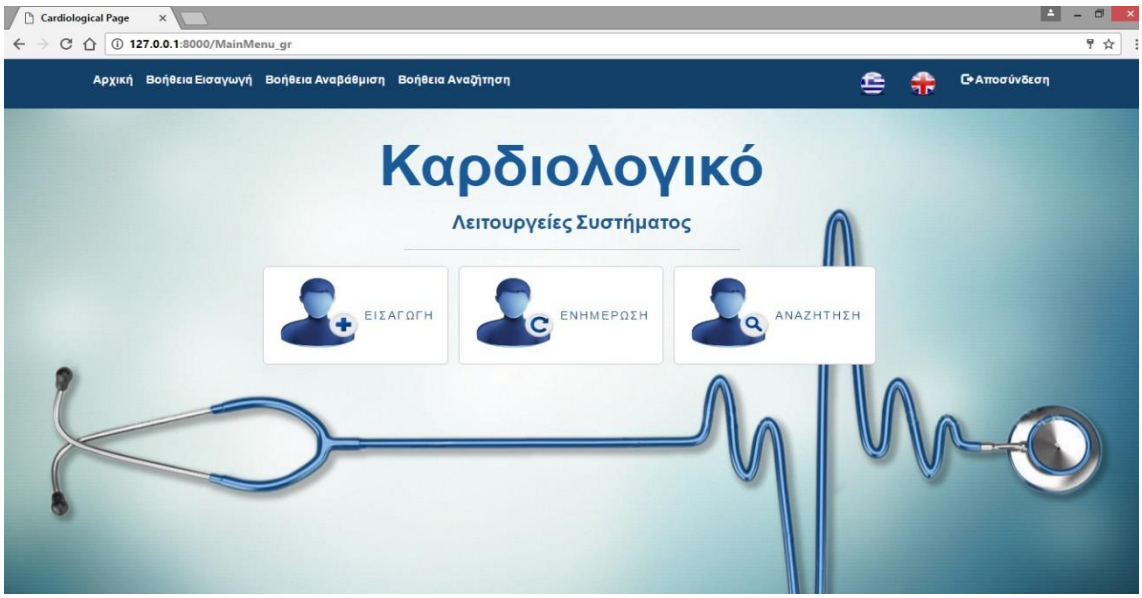
Τέλος, καλό θα ήταν για τη μεταφορά ευαίσθητων δεδομένων στο δίκτυο να γίνεται χρήση ενός ασφαλούς καναλιού, όπως είναι το SSL.

2.2 Ανασκόπηση βιβλιογραφίας – ΑΔΕ Φίλιππου Κινδύνη

Σε αυτό το υποκεφάλαιο θα κάνουμε μια μικρή αναφορά στο σύστημα το οποίο ανέπτυξε ο Φίλιππος Κινδύνης και το οποίο αφορά την εγγραφή ασθενών στο πρόγραμμα “stent for life”.



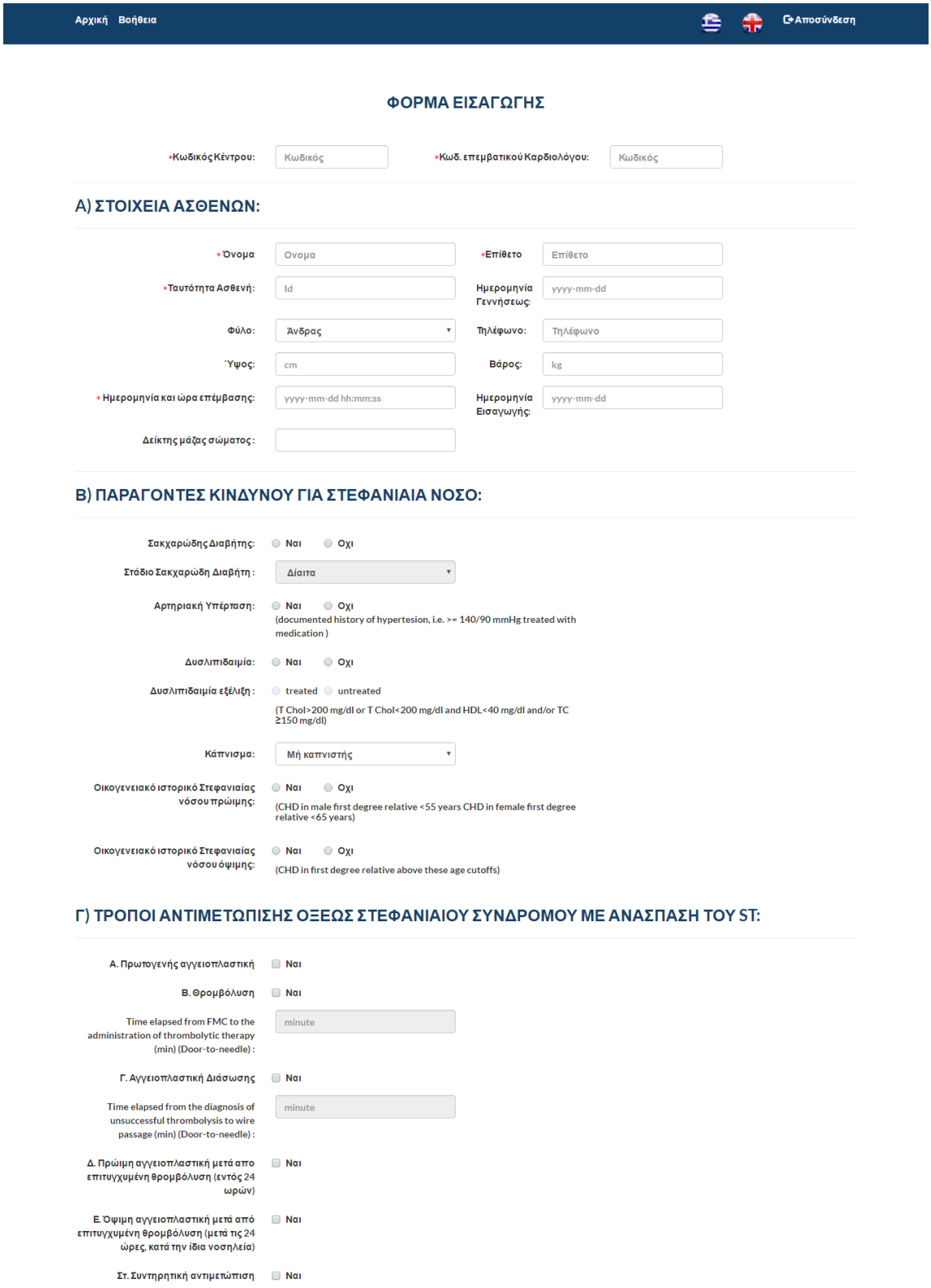
Σχήμα 2.3 – Σελίδα Ταυτοποίησης Χρήστη

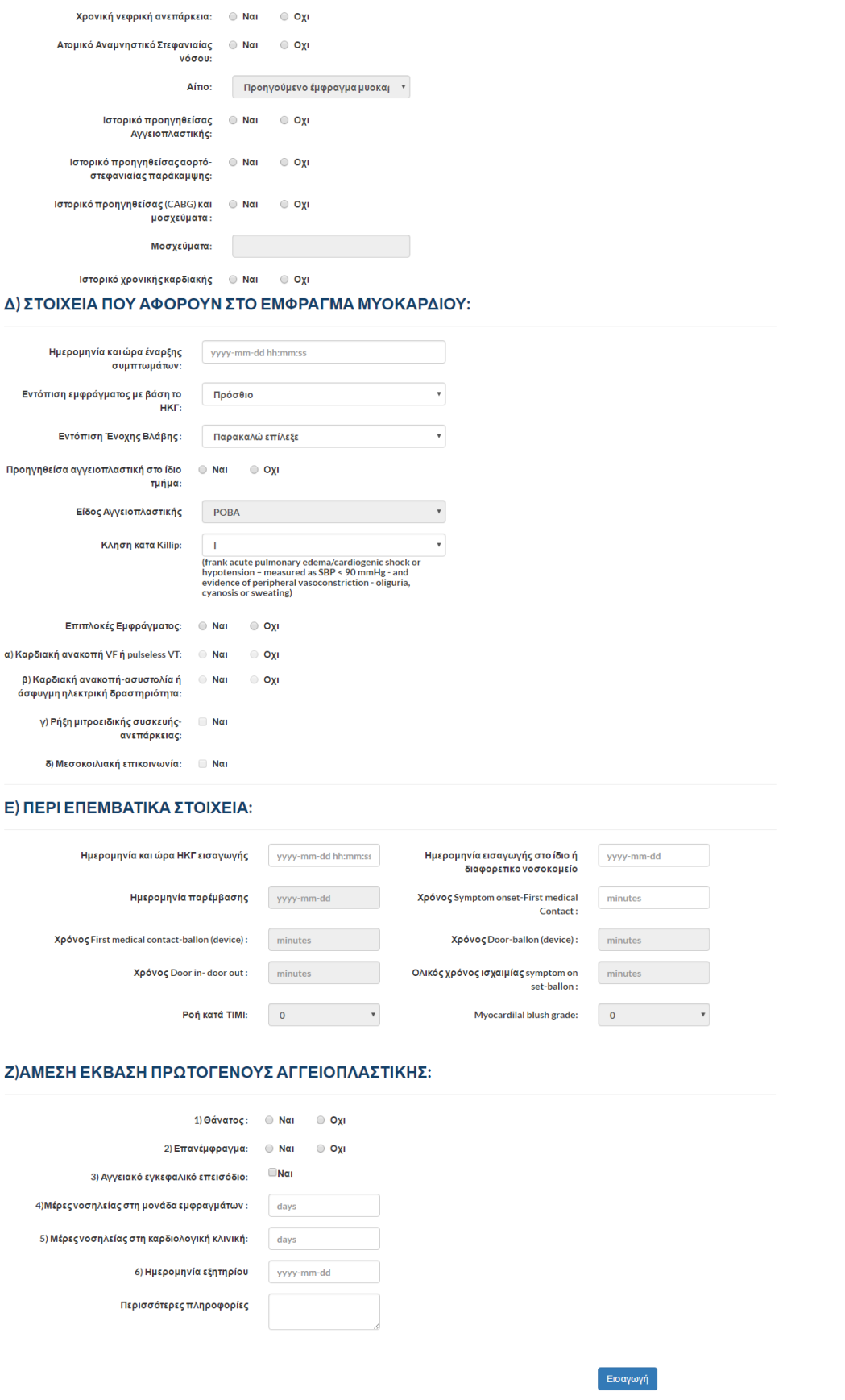


Σχήμα 2.4 – Αρχική Σελίδα

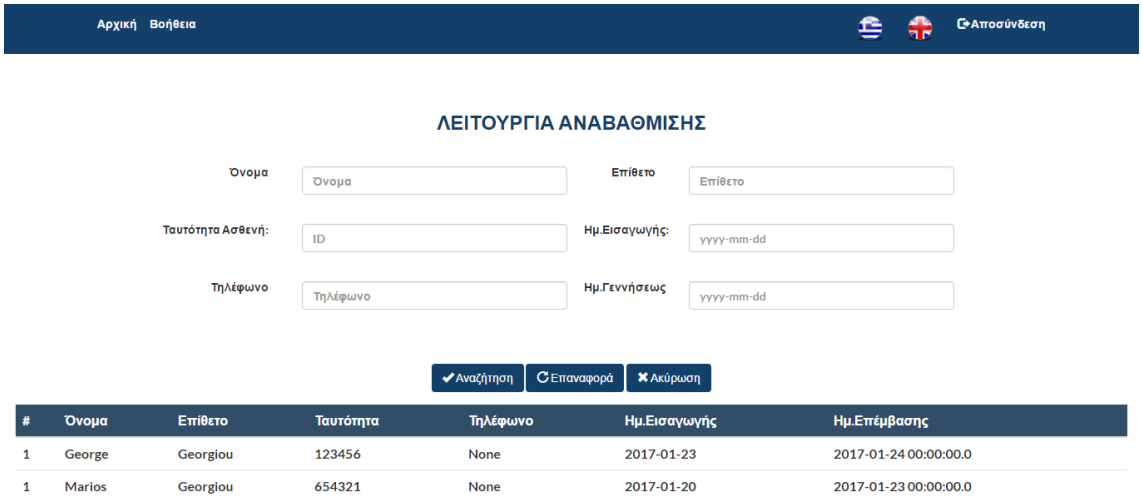
Όπως φαίνεται από το Σχήμα 2.3, η εφαρμογή του Φίλιππου επιτρέπει μόνο σε εξουσιοδοτημένους χρήστες να έχουν πρόσβαση στο σύστημα. Όταν ο χρήστης-ιατρός εισέλθει στο σύστημα εισάγοντας το username και password του, το σύστημα του δίνει τις δυνατότητες για εισαγωγή και ενημέρωση των πληροφοριών ενός ασθενή καθώς επίσης και την αναζήτηση ενός ασθενή, όπως φαίνεται στο Σχήμα 2.4.

Πιο κάτω, στα σχήματα 2.5-2.7, φαίνονται οι φόρμες για τις 3 επιλογές που έχει ο γιατρός μαζί με τα πεδία που θα χρειαστεί να συμπληρώσει:

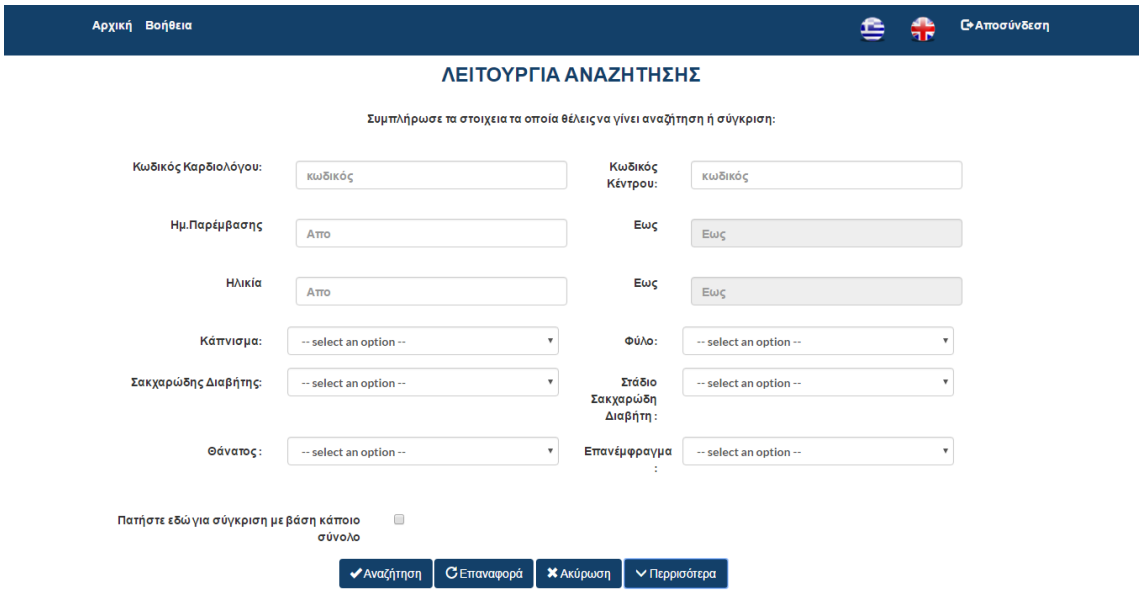




Σχήμα 2.5 – Φόρμα Εισαγωγής Στοιχείων Ασθενή



Σχήμα 2.6 – Φόρμα Ενημέρωσης Στοιχείων Ασθενή



Σχήμα 2.7 – Φόρμα Αναζήτησης Στοιχείων Ασθενή

Οι πιο πάνω φόρμες παρέχονται μεταφρασμένες τόσο στην ελληνική όσο και στην αγγλική γλώσσα και για να υλοποιηθεί το σύστημα έγινε χρήση των ακόλουθων τεχνολογιών:

- Django μέσα στο περιβάλλον ανάπτυξης PyCharm,
- Java, TomCat Server και Hibernate μέσα στο περιβάλλον ανάπτυξης Eclipse και,
- Της MySQL για τη βάση δεδομένων.

Χρήση του Django γίνεται για το rendering της ιστοσελίδας ενώ οι τεχνολογίες της Java, TomCat και Hibernate χρησιμοποιούνται ώστε να μπορεί να είναι εφικτή η επικοινωνία της ιστοσελίδας με τη βάση δεδομένων. Περισσότερες πληροφορίες για τις πιο πάνω τεχνολογίες υπάρχουν στο Κεφάλαιο 3.

Κεφάλαιο 3

Απαιτούμενη Γνώση και Τεχνολογίες

3.1 Περιβάλλοντα Ανάπτυξης Λογισμικού

Ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού (IDE) είναι μία σουίτα από λογισμικά, η οποία βοηθάει τον προγραμματιστή στην ανάπτυξη του λογισμικού του. Συνήθως περιέχει κάποιον επεξεργαστή πηγαίου κώδικα, εργαλεία για αυτόματη παραγωγή κώδικα και αποσφαλματωτή (debugger). Πιο σύγχρονα IDEs περιέχουν και μεταγλωττιστή, διερμηνέα (interpreter) όπως επίσης και εργαλεία αυτόματης συμπλήρωσης κώδικα.

3.1.1 JetBrains PyCharm

Το PyCharm είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού (IDE), το οποίο χρησιμοποιείται για την ανάπτυξη λογισμικού κυρίως σε γλώσσα Python. Έχει αναπτυχθεί από την εταιρεία JetBrains [10]. Παρέχει ανάλυση κώδικα, ολοκληρωμένες μονάδες δοκιμασιών, γραφικό debugger και στηρίζει την ανάπτυξη ιστοσελίδων με Django.

Επίσης, το PyCharm είναι μια πλατφόρμα, η οποία υποστηρίζεται από τα λειτουργικά συστήματα Windows, MacOS και Linux. Το Community Edition διατίθεται κάτω από την άδεια χρήσης της Apache, ενώ υπάρχει και το Professional Edition, το οποίο διατίθεται βάσει ιδιόκτητης άδειας και το οποίο περιέχει επιπλέον χαρακτηριστικά.

Για τους σκοπούς της διπλωματικής αυτής εργασίας έχει γίνει χρήση του Professional Edition, το οποίο δικαιούμαστε δωρεάν σαν φοιτητές.

3.1.2 Eclipse IDE for Java EE Developers

Η Eclipse [11] είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού (IDE) που χρησιμοποιείται για τον προγραμματισμό ηλεκτρονικών υπολογιστών. Περιέχει ένα βασικό χώρο εργασίας και ένα επεκτάσιμο σύστημα με plug-ins για διαμόρφωση του περιβάλλοντος με βάση τις ανάγκες του χρήστη. Η Eclipse είναι γραμμένη κυρίως σε Java και κύρια χρήση της είναι για την ανάπτυξη εφαρμογών σε Java, αν και μπορεί να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών και σε άλλες γλώσσες προγραμματισμού με την ανάλογη χρήση plugins. Γλώσσες για τις οποίες υπάρχουν plugins στην Eclipse είναι οι εξής: Ada, ABAP, C, C ++, COBOL, Fortran, Haskell , JavaScript, , Lua, Perl, PHP, Prolog, την Python, R, Ruby (συμπεριλαμβανομένου του πλαισίου Ruby on Rails). Τα περιβάλλοντα ανάπτυξης περιλαμβάνουν, μεταξύ άλλων, και τα εργαλεία ανάπτυξης Eclipse Java (JDT) για Java, Eclipse CDT για την C/C++ και Eclipse PDT για την PHP. Τέλος, η Eclipse αποτελεί λογισμικό ανοικτού κώδικα.

3.1.3 MySQL Workbench

Για τη διαχείριση της βάσης δεδομένων γίνεται χρήση του MySQL Workbench. Το MySQL Workbench [12] αποτελεί εργαλείο για προγραμματιστές και διαχειριστές βάσεων δεδομένων. Ουσιαστικότερα, το MySQL Workbench είναι μια γραφική διεπαφή για την εύκολη χρήση του MySQL sever που επιτρέπει 3 είδη εργασιών που επεκτείνονται σε διάφορες λειτουργείες. Τα 3 είδη εργασιών είναι :

- Διαχείριση του MySQL sever όπου μπορούμε να δημιουργήσουμε και να διαχειριστούμε τη βάση δεδομένων και να ελέγξουμε την σύνδεση μας.
- Σχεδίαση σχεσιακών βάσεων δεδομένων όπου δίνεται η δυνατότητας δημιουργίας πινάκων και συσχετίσεων μεταξύ πινάκων (primary key , foreign key etc. .)
- Δημιουργία και εκτέλεση SQL ερωτημάτων ,κάνοντας ερωτήσεις στην μορφή που αναγνωρίζει η SQL (from,where ,to) και περνώντας δεδομένα που είναι τα αποτελέσματα της αναζήτησης μας.

Από τις πιο πάνω τις ομάδες για την πρώτη και την τρίτη απαιτείται να τρέχει ήδη ο MySQL server αφού εκεί στέλνονται τα στοιχεία για να εκτελεστούν. Επιπρόσθετα από τις κύριες λειτουργείες παρέχονται οι δυνατότητες διατήρησης αντίγραφου ασφαλείας αλλά και μοντελοποίησης δεδομένων.

Το MySQL Workbench είναι μια αξιόπιστη και ασφαλής έκδοση ανοιχτού κώδικα για την δημιουργία μιας βάσης δεδομένων. Μπορούμε να εγκαταστήσουμε το MySQL Workbench στο λειτουργικό Windows, στο Linux και στο Mac. Για τα Windows χρειάζεται και το Microsoft .net framework και κάποιες βιβλιοθήκες.

3.2 Γλώσσες Προγραμματισμού

Γλώσσα προγραμματισμού είναι μια επίσημη γλώσσα του υπολογιστή σχεδιασμένη για να δίνει οδηγίες σε μηχανές, και κυρίως σε υπολογιστές. Οι γλώσσες προγραμματισμού μπορεί να χρησιμοποιηθούν για τη δημιουργία προγραμμάτων, τα οποία να ελέγχουν τη συμπεριφορά μιας μηχανής ή για να εκφράσουν έναν αλγόριθμο.

3.2.1 Django

Το Django είναι ένα δωρεάν και ανοικτού κώδικα web framework γραμμένο σε Python, το οποίο ακολουθεί την αρχιτεκτονική του Model-View-Template (MVT). Τα κύρια συστατικά της αρχιτεκτονικής αυτής είναι τα εξής:

- Model: Αναφέρεται στην πληροφορία που ανταλλάσσεται μεταξύ ενός διακομιστή και μιας σχεσιακής βάσης δεδομένων.
- View: Χρησιμοποιείται για την εμφάνιση της πληροφορίας στο χρήστη.
- Controller: Ασχολείται με τη ροή ελέγχου της πληροφορίας.

Κύριος στόχος του Django είναι να ευκολύνει τη δημιουργία περίπλοκων ιστοσελίδων που είναι συνδεδεμένες με κάποια βάση δεδομένων. Κάνει πολύ εκτεταμένη χρήση των

plug ins αλλά και της επαναχρησιμοποίησης κώδικα και δίνει πολλή σημασία στην ταχύτητα ανάπτυξης και στην αρχή “don’t repeat yourself”. Παρέχει την ευκολία στο χρήστη να επικοινωνήσει με τη βάση δεδομένων γράφοντας σε γλώσσα Python αντί SQL queries, παρέχει ευκολία και κομψότητα στη διαχείριση των URLs, όπως επίσης και σύστημα για caching. Επιπλέον, παρέχει τη δυνατότητα για δημιουργία πολύγλωσσης εφαρμογής.

Κάποιες γνωστές ιστοσελίδες που κάνουν χρήση του Django είναι οι Pinterest, Instagram, Mozilla, The Washington Times, Disqus, Bitbucket και Nextdoor. [15]

Στην παρούσα εργασία κάνουμε χρήση του Django ώστε με τις υλοποιήσεις και ευκολίες που παρέχει να γίνει πιο εύκολη η συγγραφή του κώδικα της ιστοσελίδας μας.

3.2.1.1 Python

Η Python είναι μια ψηλού επιπέδου γλώσσα προγραμματισμού, την οποία δημιούργησε ο Guido van Rossum το 1991. Έχει τις εξής δύο φιλοσοφίες:

1. Αναγνωσιμότητα: Γίνεται χρήση των whitespaces για την εσοχή του κώδικα αντί της χρήσης άγκιστρων ({,}) ή ειδικών λέξεων (begin, end).
2. Σύντομη Σύνταξη: Παρέχεται η δυνατότητα στους προγραμματιστές να εκφράσουν έννοιες σε λιγότερες γραμμές κώδικα από ότι θα εκφράζονταν σε άλλες γλώσσες όπως η C++ και Java.

Επιπλέον, η Python παρέχει σύστημα δυναμικών τύπων για τις μεταβλητές και αυτόματη διαχείριση μνήμης. Οι interpreters της γλώσσας είναι διαθέσιμοι για διάφορα λειτουργικά συστήματα, το οποίο επιτρέπει στην Python να τρέχει σε μια πληθώρα λειτουργικών συστημάτων.

Χρήση της Python γίνεται μέσα στο Django και μέσω αυτής είναι εφικτή η υλοποίηση κομματιών που αφορούν την ιστοσελίδα.

3.2.2 HTML

Η HTML, ακρώνυμο του HyperText Markup Language, είναι μια γλώσσα σήμανσης , η οποία χρησιμοποιείται για τη δημιουργία ιστοσελίδων και εφαρμογών ιστού. Μαζί με το CSS και Javascript (στα οποία θα αναφερθούμε στη συνέχεια) αποτελούν τον ακρογωνιαίο λίθο του World Wide Web. Ο κώδικας της HTML είναι γραμμένος σε μορφή απλού

κειμένου και γι’ αυτό δε γίνεται compile αλλά interpet. Αποτελείται από ετικέτες οι οποίες εμπερικλείονται σε < >. Ετικέτες είναι συνήθως δομικά στοιχεία μιας ιστοσελίδας όπως είναι οι επικεφαλίδες, οι φόρμες εισαγωγής, παραγράφοι, εικόνες κ.ά. Κάθε ιστοσελίδα γραμμένη σε HTML χρειάζεται να ξεκινά με την ετικέτα <html> και να τελειώνει με </html>. Η ετικέτα αυτή περιέχει όλη τη σελίδα και μέσα στο εύρος της ετικέτας αυτής περικλείονται οι δύο κύριες ενότητες, η <head></head> και <body></body>.

Μέσα στο Head υπάρχουν πληροφορίες που αφορούν το έγγραφο και οι οποίες συνήθως δεν είναι ορατές στο χρήστη. Τέτοιες πληροφορίες μπορεί να είναι η κωδικοποίηση του εγγράφου, ο τίτλος που θα φαίνεται στο tab του browser και άλλες μετα-πληροφορίες που αφορούν το έγγραφο. Στο Body υπάρχει το περιεχόμενο του αρχείου, αυτά δηλαδή που είναι ορατά στο χρήστη, μαζί με κάποιες μεθόδους ασφαλείας, όπως είναι τα cookies.

Στην παρούσα διπλωματική εργασία χρησιμοποιούμε την HTML για να δημιουργήσουμε και να παρουσιάσουμε τα διάφορα μέρη της ιστοσελίδας.

3.2.2.1 CSS

Ακρωνύμιο του Cascading Style Sheets, η CSS είναι μια γλώσσα η οποία έχει σαν σκοπό τη μορφοποίηση δομικών συστατικών μιας γλώσσας σήμανσης (όπως είναι η HTML). Με τη χρήση της CSS μπορούμε να ορίσουμε πως θα παρουσιάζονται, για παράδειγμα, τα γράμματα σε μια σελίδα, τι χρώμα και μέγεθος θα έχουν, τη στοίχισή τους και γενικά μπορούμε να ορίσουμε όλο το styling της σελίδας. Ο κώδικας της CSS μπορεί να βρίσκεται σε αρχείο με κατάληξη .css το οποίο θα κάνουμε import στην HTML ή να εμπεριέχεται απευθείας στον κώδικά της.

Στην εργασία μας γίνεται χρήση του CSS για να καταστήσει την ιστοσελίδα πιο φιλική προς το χρήστη μέσα από τις διάφορες μορφοποιήσεις που παρέχει.

3.2.2.2 Javascript

Η Javascript είναι μια ψηλού επιπέδου, δυναμική γλώσσα προγραμματισμού η οποία δεν γίνεται compile αλλά interpret. Χρησιμοποιείται κυρίως για να προσθέσει διαδραστικότητα σε μια ιστοσελίδα και αρχικά αναπτύχθηκε από τη Netscape. Είναι εμπνευσμένη από την Java με σύνταξη όμως που θυμίζει περισσότερο C και τρέχει στην πλευρά του client. Αυτό σημαίνει ότι ο πηγαίος κώδικας που βρίσκεται στη σελίδα που φορτώνει ο χρήστης εκτελείται από τον ίδιο τον browser του χρήστη χωρίς να χρειαστεί να γίνει σύνδεση με τον server. Όπως και με τη CSS, ο κώδικας της Javascript μπορεί να είναι

αποθηκευμένος σε αρχείο .js το οποίο θα κάνουμε import στον κώδικα της HTML ή μπορεί να γραφτεί στο ίδιο αρχείο μαζί με τον κώδικα της HTML. Η Javascript καλείται μέσα από το <script> και δίνει τη δυνατότητα στην ιστοσελίδα να αλληλεπιδρά με τον χρήστη ανάλογα με τις κινήσεις του τελευταίου. Μπορούμε να ορίσουμε να δημιουργείται ένα συμβάν όταν ο χρήστης κάνει κλικ με το mouse σε ένα μέρος της ιστοσελίδας, όταν συμπληρώσει ένα πεδίο, όταν μια φόρμα γίνει submit κλπ.

Στην παρούσα εργασία γίνεται χρήση της Javascript για να καταστήσει την ιστοσελίδα πιο ενεργητική προς τον χρήστη μέσα από τις διάφορες αλληλεπιδραστικές επιλογές που παρέχονται.

3.2.3 Java

Λέγοντας Java μπορούμε να εννοούμε ένα από τα δύο: τη γλώσσα προγραμματισμού ή την πλατφόρμα. Αρχικά, όσον αφορά τη γλώσσα προγραμματισμού, μπορούμε να αναφέρουμε τα εξής: Η Java είναι μια γενικού σκοπού γλώσσα προγραμματισμού, η οποία δημιουργήθηκε από τον James Gosling το 1995. Είναι μια μείξη των γλωσσών προγραμματισμού C και C++ και υποστηρίζει παράλληλη εκτέλεση διεργασιών, είναι βασισμένη σε κλάσεις και είναι object-oriented. Επίσης, έχει σχεδιαστεί με τέτοιο τρόπο ώστε να έχει όσο το δυνατό λιγότερες εξαρτήσεις που αφορούν την υλοποίηση. Επιπλέον, δίνει τη δυνατότητα στους προγραμματιστές να γράψουν μια φορά τα προγράμματά τους και να μπορούν να τα τρέξουν σε διάφορες μηχανές (write once, run anywhere). Αυτό σημαίνει ότι, κάθε μεταγλωττισμένος κώδικας Java μπορεί να τρέξει σε οποιαδήποτε πλατφόρμα η οποία υποστηρίζει Java χωρίς να χρειαστεί να γίνει για δεύτερη φορά μεταγλώττιση. Συνήθως, οι εφαρμογές που είναι γραμμένες σε Java μεταγλωττίζονται σε bytecode, το οποίο μετά μπορεί να τρέξει σε οποιοδήποτε Java Virtual Machine (JVM), ανεξαιρέτως της αρχιτεκτονικής της μηχανής. Σύμφωνα με στατιστικά στοιχεία που συλλέχθηκαν το 2017 και δημοσιεύθηκαν σε διάφορες ιστοσελίδες [16] [17], η Java βρίσκεται στις πρώτες θέσεις όσον αφορά τη δημοσιότητα και τη χρήση από τους προγραμματιστές και ειδικά όσον αφορά εφαρμογές ιστού τύπου πελάτη-εξυπηρετητή (client-server).

Principles:

There were five primary goals in the creation of the Java language:[16]

1. It must be "simple, object-oriented, and familiar".
2. It must be "robust and secure".

3. It must be "architecture-neutral and portable".
4. It must execute with "high performance".
5. It must be "interpreted, threaded, and dynamic".

Όσον αφορά την πλατφόρμα της Java, είναι ένα περιβάλλον στο οποίο μπορεί να τρέξει ένα πρόγραμμα και το οποίο είναι software-only, σε αντίθεση με άλλες πλατφόρμες που μπορεί να είναι και hardware. Τρέχει πάνω σε άλλες hardware πλατφόρμες και αποτελείται από 2 κύρια μέρη:

1. το Java Virtual Machine (JVM), το οποίο επιτρέπει την εκτέλεση προγραμμάτων Java σε οποιαδήποτε μηχανή και που με αυτό τον τρόπο διαχωρίζει το πρόγραμμα από τα διάφορα λειτουργικά συστήματα και CPUs.
2. το Java Application Programing Interface (Java API).

Στην παρούσα διπλωματική εργασία γίνεται χρήση της γλώσσας προγραμματισμού Java ώστε να καταστεί εφικτή η επικοινωνία μεταξύ του user interface και της βάσης δεδομένων. Λειτουργεί, δηλαδή, σαν ενδιάμεσος μεσολαβητής των δύο αυτών συστατικών στοιχείων.

3.2.3.1 Tomcat server v7.0

Ο Tomcat Server [18] είναι ένα ανοιχτού κώδικα Java Servlet Container, το οποίο αναπτύχθηκε από την Apache Software Foundation και σαν σκοπό έχει την εκτέλεση των Java servlets και τη σωστή λειτουργία του back-end της ιστοσελίδας.

Στην παρούσα διπλωματική εργασία με τη χρήση του Tomcat γίνεται εφικτή η λειτουργία του Java servlet μας, το οποίο διασυνδέει τη βάση δεδομένων με το User Interface.

3.2.3.2 Hibernate

Το Hibernate [19] είναι ένα ανοιχτού κώδικα λογισμικό, το οποίο σαν σκοπό έχει τη σύνδεση αντικειμένων μιας αντικειμενοστραφής γλώσσας προγραμματισμού (π.χ. Java) με τους πίνακες μιας σχεσιακής βάσης δεδομένων. Η σύνδεση αυτή επιτυγχάνεται με την χρήση επιπρόσθετης πληροφορίας (metadata), η οποία είτε τοποθετείται μέσα στον κώδικα της Java είτε σε ξεχωριστά αρχεία τύπου .xml και η οποία περιγράφει την αντιστοιχία μεταξύ των αντικειμένων και της βάσης δεδομένων. Πιο γενικά, το hibernate δίνει τη

δυνατότητα για αυτόματη μετατροπή της μιας μορφής (αντικείμενα) στην άλλη (σχεσιακή βάση δεδομένων).

Στην παρούσα διπλωματική εργασία το Hibernate χρησιμοποιείται για να γίνει η μετάφραση των αντικειμένων που έχουμε στην Java στα αντίστοιχα πεδία που έχουμε στη βάση δεδομένων μας.

3.2.4 MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων το οποίο αριθμεί περισσότερες από 11 εκατομμύρια εγκαταστάσεις. Είναι μια από τις κύριες τεχνολογίες του LAMP (Linux, Apache, MySQL, Perl/PHP/Python) stack και παρέχει στον προγραμματιστή ένα αριθμό από δυνατότητες:

1. Queries προς τη βάση με δυνατές επιλογές για:
    α. Δημιουργία πίνακα στη βάση.
    β. Διαγραφή πίνακα από τη βάση.
    γ. Εισαγωγή δεδομένων σε πίνακα.
    δ. Τροποποίηση (ενημέρωση, διαγραφή) αποθηκευμένων δεδομένων.
    ε. Παρουσίαση αποθηκευμένων δεδομένων με βάση κλειδιά αναζήτησης.
2. Triggers που αφορούν είτε τη βάση δεδομένων είτε ένα μεμονωμένο πίνακα και τα οποία μετά από μια συγκεκριμένη ενέργεια που γίνεται στη βάση (π.χ. εισαγωγή, ενημέρωση ή διαγραφή πεδίου από πίνακα) επιτρέπουν την εκτέλεση μιας δεύτερης ενέργειας μέσα στη βάση (π.χ. εισαγωγή, ενημέρωση ή διαγραφή πεδίου στον παρόντα ή σε άλλο πίνακα).

Στην παρούσα διπλωματική εργασία κάνουμε χρήση της MySQL για να δημιουργήσουμε και να αποθηκεύσουμε το audit trail με τις ενέργειες του κάθε γιατρού (εισαγωγή, τροποποίηση, ανάγνωση από τη βάση). Το audit trail είναι υλοποιημένο με βάση τις οδηγίες του RFC 3881 [20] και του ATNA [21]. Περισσότερες πληροφορίες σχετικά με τις οδηγίες αυτές βρίσκονται στο Κεφάλαιο 5.

Κεφάλαιο 4

Ανάλυση Απαιτήσεων και Προδιαγραφές Συστήματος

4.1 Εισαγωγή

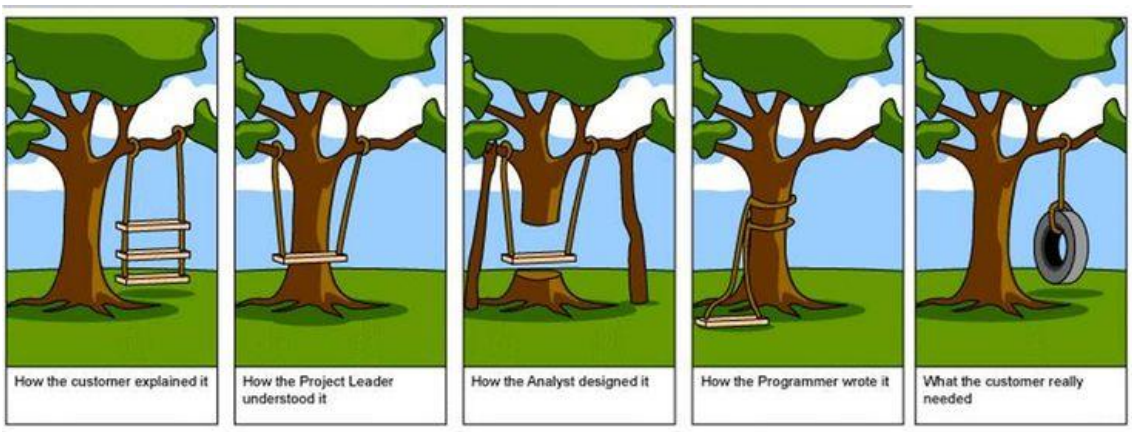
Η ανάλυση απαιτήσεων είναι το πρώτο από τα πέντε στάδια ανάπτυξης μιας εφαρμογής: 1) Ανάλυση, 2) Σχεδίαση, 3) Υλοποίηση, 4) Έλεγχος, 5) Συντήρηση. Σε αυτό το στάδιο, το άτομο που είναι υπεύθυνο επικοινωνεί με τα πρόσωπα που ενδιαφέρονται για την α-νάπτυξη ή αναβάθμιση ενός συστήματος ή μιας εφαρμογής. Στόχος και σκοπός είναι μέσα από την επικοινωνία του αυτή να εξακριβώσει ο αναλυτής και να κατανοήσει τι ανάγκες έχει ο μελλοντικός χρήστης του συστήματος.

Σχήμα 4.1 – Στάδιο Ανάλυσης Απαιτήσεων [33]

Πολλές φορές είναι πιθανό ο χρήστης να ζητά από το σύστημα να έχει κάποια χαρακτηριστικά ενώ στην πραγματικότητα να χρειάζεται άλλα. Γι' αυτό και στο στάδιο αυτό ο αναλυτής έχει ένα αριθμό από επιλογές, κάθε μία με τα υπέρ και τα κατά της, ώστε να εξακριβώσει ποιες είναι οι ανάγκες του χρήστη. Οι επιλογές αυτές είναι οι εξής:

- Ερωτηματολόγια
- Παρακολούθηση εν ώρα εργασίας
- Συζήτηση-συνέντευξη με τα άτομα που θα χρησιμοποιούν το σύστημα

Αυτό το στάδιο ίσως να είναι από τα πιο σημαντικά στον κύκλο ανάπτυξης ενός πληροφορικού συστήματος γιατί εδώ γίνεται η εξακρίβωση των αναγκών του χρήστη. Αν κάτι κατανοηθεί λάθος τότε όλα τα επόμενα στάδια θα πάνε λάθος με επιπτώσεις τόσο σε χρόνο όσο και σε κόστος. Καλό θα ήταν, επομένως, να γίνεται μια πολύ καλή κατανόηση των αναγκών του χρήστη ώστε η ανάλυση απαιτήσεων να είναι ακριβής και έτσι να τεθούν σωστές βάσεις για την ανάπτυξη του συστήματος.



Σχήμα 4.2 – Κατανόηση Απαιτήσεων Πελάτη [34]

4.2 Σκοπός Ανάλυσης Απαιτήσεων στην παρούσα εργασία

Σκοπός στο παρόν στάδιο είναι, όπως αναφέραμε και προηγουμένως, η εξακρίβωση των αναγκών που έχουν οι χρήστες του συστήματος. Στόχος είναι τα προγράμματα που θα αναπτυχθούν να ανταποκρίνονται στις ανάγκες και απαιτήσεις των μελλοντικών χρηστών.

Όπως αναφέρθηκε και σε προηγούμενα κεφάλαια, ο σκοπός της διπλωματικής εργασίας αλλά και ο στόχος της είναι να προστεθεί ασφάλεια γύρω από το σύστημα "stent for life",

το οποίο έχει υλοποιήσει ο Φίλιππος Κινδύνης και το οποίο θα χειρίζονται γιατροί καρδιολόγοι. Λόγω του ότι το κομμάτι αυτό της ασφάλειας δεν έγκειται στο πεδίο γνώσεως των γιατρών, η επικοινωνία για την εύρεση των απαιτήσεων του συστήματος έγινε με καθηγητές του Πανεπιστημίου Κύπρου, οι οποίοι έχουν γνώση/είναι γνώστες του αντικειμένου.

Γνωρίζοντας ότι τα δεδομένα τα οποία θα είναι αποθηκευμένα στο σύστημα θα είναι ιατρικής φύσεως, έγινε μια συζήτηση με τους καθηγητές του Πανεπιστημίου Κύπρου ώστε να οριστεί το επίπεδο ασφαλείας, το οποίο θα παρέχει την απαραίτητη ασφάλεια στα δεδομένα αυτά.

4.3 Ιδιότητες που χρειάζεται να πληροί το σύστημα

Μέσα από τη συζήτηση με τους καθηγητές Κωνσταντίνο Παττίχη και Ηλία Αθανασόπουλο αλλά και με άλλα πρόσωπα από το ακαδημαϊκό προσωπικό, έγινε ο ορισμός των μέτρων που πρέπει να παρθούν ώστε να επιτευχθεί η ασφάλεια του συστήματος.

Αρχικά, και λόγω του ότι ήταν κάτι που δεν υπήρχε στο σύστημα που υλοποίησε ο Φίλιππος, ορίστηκε η δημιουργία μιας επιπλέον σελίδας. Η σελίδα αυτή θα επιτρέπει την ασφαλή δημιουργία χρηστών στο σύστημα. Το σύστημα θα πρέπει να δέχεται τις αιτήσεις των νέων χρηστών με τρόπο που να μη διαρρέουν οι πληροφορίες και να φτάνουν σε λάθος άτομα. Επιπλέον, θα χρειαστεί να ληφθούν μέτρα ώστε να αποφευχθεί η μαζική δημιουργία αιτήσεων από κακόβουλα προγράμματα-ρομπότ. Όσον αφορά τις αιτήσεις, θα ήταν προτιμότερο οι νέοι χρήστες να μην έχουν πρόσβαση στο σύστημα ακριβώς μόλις κάνουν την αίτησή τους αλλά μετά από την έγκριση της αίτησης από το διαχειριστή.

Επιπλέον, το σύστημα θα πρέπει να είναι ανθεκτικό σε επιθέσεις που στόχο έχουν τα δεδομένα που κρατά. Τέτοιες επιθέσεις είναι το SQL Injection, Cross Site Scripting (XSS), Cross Site Request Forgery (CSRF) και Man in the Middle (MitM). Για όλες αυτές τις επιθέσεις θα πρέπει να ληφθούν μέτρα αντιμετώπισής τους.

Απαραίτητη θα είναι επίσης και η κωδικοποίηση κάποιων ή όλων των δεδομένων στη βάση ώστε να μην είναι αναγνώσιμα και κατανοητά από τον άνθρωπο.

Τέλος, για την πιο σωστή και πρακτική καταγραφή και έλεγχο των κινήσεων του κάθε γιατρού-χρήστη θα πρέπει να γίνει υλοποίηση ενός audit trail με βάση τα κριτήρια και

οδηγίες του RFC 3881 [20], όπου θα κρατούνται πληροφορίες σχετικά με τον γιατρό, τις ενέργειες που έκανε καθώς επίσης και την τοποθεσία από την οποία ενώθηκε (π.χ. DNS Server, IP address).

Κεφάλαιο 5

Σχεδιασμός και Υλοποίηση

5.1 Εισαγωγή

Στο παρόν κεφάλαιο θα αναφερθούμε στο πώς οι απαιτήσεις που αναφέρθηκαν/ορίστηκαν στο Κεφάλαιο 4 έτυχαν σχεδιασμού και υλοποίησης. Πιο συγκεκριμένα θα γίνει μια λεπτομερής περιγραφή του πώς έγινε ο σχεδιασμός και έπειτα η υλοποίηση της ασφάλειας στο σύστημα. Προτού αναφερθούμε, όμως, στο πώς έγινε η σχεδίαση και υλοποίηση του συστήματος θα αναφέρουμε κάποιες πληροφορίες που αφορούν τα δύο αυτά στάδια ανάπτυξης λογισμικού.

Η σχεδίαση αποτελεί το δεύτερο από τα πέντε στάδια ανάπτυξης μιας εφαρμογής και σαν σκοπό έχει να μπει σε μεγαλύτερο βάθος και να τελειοποιήσει τη δραστηριότητα της ανάλυσης ώστε να τη φέρει σε μορφή που να μπορεί να υλοποιηθεί από τους προγραμματιστές. Σε αυτό το στάδιο πρέπει να οριστούν και οι λειτουργικές και μη λειτουργικές απαιτήσεις, όπως είναι η γλώσσα προγραμματισμού που θα χρησιμοποιηθεί, οι δομές δεδομένων, όπως επίσης και η απόδοση, ασφάλεια και κόστος του έργου.

Το τρίτο μέρος του κύκλου ανάπτυξης της εφαρμογής είναι η υλοποίηση των προγραμμάτων που θα απαρτίσουν την ολοκληρωμένη εφαρμογή. Σε αυτό το στάδιο, και αφού

προηγουμένως έχουν οριστεί η γλώσσα προγραμματισμού και οι δομές δεδομένων που θα χρησιμοποιηθούν, ξεκινά η υλοποίηση. Σε αυτό το στάδιο δύναται να γίνει χρήση ολοκληρωμένων περιβαλλόντων ανάπτυξης (IDEs) τα οποία έχουν υλοποιηθεί για να βοηθούν τους προγραμματιστές στην ανάπτυξη των προγραμμάτων τους.

5.2 Τακτική που εφαρμόστηκε για την υλοποίηση

Όπως αναφέρθηκε και σε προηγούμενα κεφάλαια, στόχος της διπλωματικής εργασίας ήταν η διεκπεραίωση 3 βασικών λειτουργιών, οι οποίες θα επεκτείνουν το σύστημα που υλοποίησε ο Φίλιππος Κινδύνης. Οι τρεις αυτές λειτουργίες είναι οι εξής:

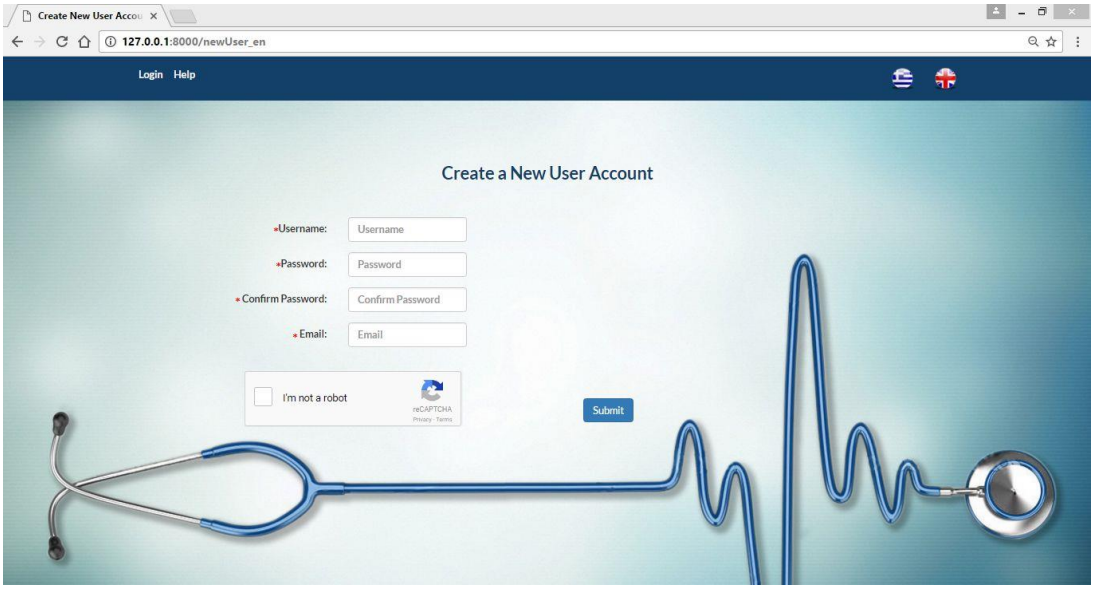
1. Δημιουργία μιας επιπλέον σελίδας στον ιστότοπο της εφαρμογής, η οποία θα επιτρέπει τη δημιουργία νέων χρηστών στο σύστημα.
2. Κωδικοποίηση στη βάση δεδομένων των πληροφοριών που αφορούν τους ασθενείς.
3. Δημιουργία audit trail για τις ενέργειες που γίνονται εντός της βάσης δεδομένων με βάση τις προδιαγραφές του OpenATNA.

Για να επιτευχθεί η υλοποίηση των πιο πάνω λειτουργιών ήταν απαραίτητη η επικοινωνία με διάφορα πρόσωπα από το ακαδημαϊκό προσωπικό. Με την βοήθεια και την κατεύθυνσή τους τέθηκαν οι προδιαγραφές και ξεκίνησε η υλοποίηση. Για την πιο εύκολη υλοποίηση απομονώθηκε η κάθε λειτουργία ώστε να τελειώνει το ένα κομμάτι πρώτα και μετά να ξεκινά η ανάπτυξη του επομένου. Έτσι, πρώτα έγινε η υλοποίηση της σελίδας που επιτρέπει τη δημιουργία των νέων χρηστών, έπειτα η κωδικοποίηση των πληροφοριών στη βάση δεδομένων και τέλος η δημιουργία του audit trail.

5.3 Υλοποίηση σελίδας που να επιτρέπει τη δημιουργία νέων χρηστών

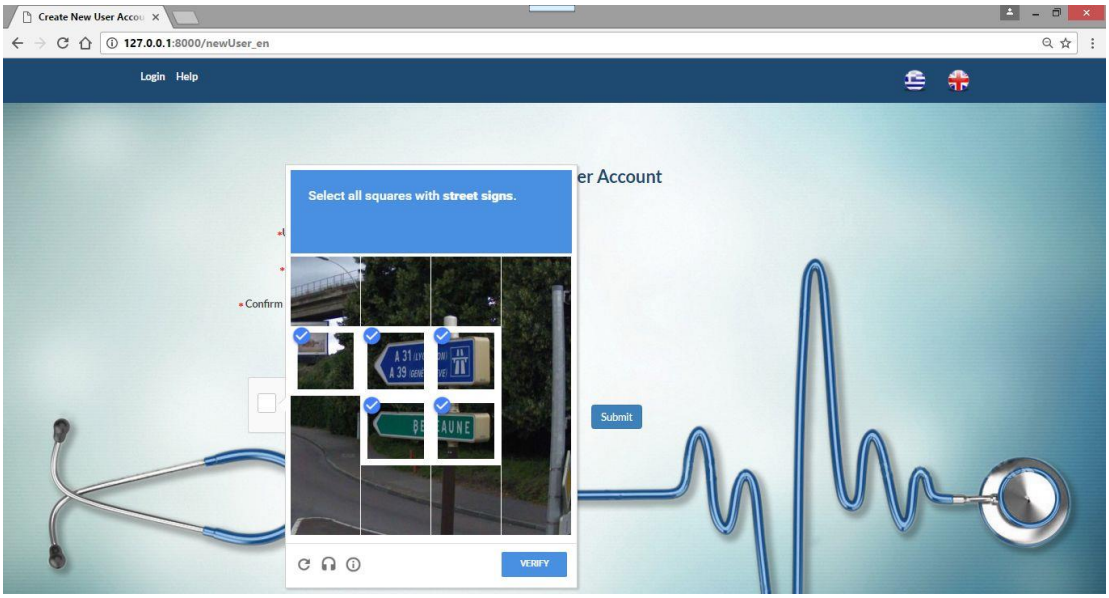
Για το σχεδιασμό της νέας σελίδας ακολουθήθηκε η μορφή που έχουν και οι υπόλοιπες σελίδες του ιστοχώρου. Η σελίδα ζητά από το άτομο που κάνει αίτηση να εισαγάγει στο σύστημα το username που επιθυμεί να χρησιμοποιεί καθώς επίσης και τον κωδικό και email του. Επιπλέον, περιέχει και μια εφαρμογή, τη reCAPTCHA της Google, η οποία χρησιμεύει στην αποτροπή των μαζικών αιτήσεων για δημιουργία νέων χρηστών από ρομπότ. Το όνομα χρήστη όπως επίσης και ο κωδικός για να γίνουν δεκτά θα πρέπει να περιέχουν τουλάχιστον 6 χαρακτήρες. Για τη σωστή εισαγωγή του κωδικού έχει προστεθεί και ένα επιπλέον πεδίο, το οποίο ζητά να εισαχθεί ξανά ο κωδικός. Επιπρόσθετα, η

σελίδα μπορεί να εμφανιστεί τόσο στα Ελληνικά όσο και στα Αγγλικά. Πιο κάτω φαίνεται η σελίδα στα Αγγλικά όταν φορτωθεί από τον browser.

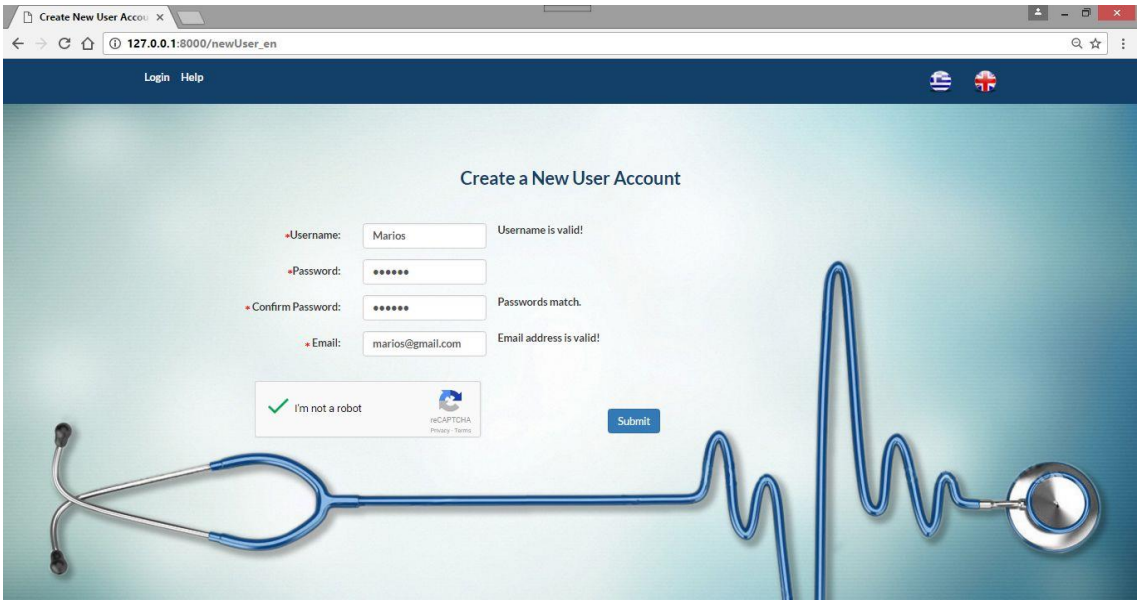


Σχήμα 5.1 – Σελίδα για δημιουργία νέου χρήστη

Αν τα πεδία και το reCAPTCHA συμπληρωθούν σωστά, τότε η δημιουργία του νέου χρήστη γίνεται με επιτυχία. Παρόλα αυτά, ο νέος χρήστης δε θα έχει ακόμα πρόσβαση στο σύστημα μέχρι ο διαχειριστής να τον εγκρίνει. Για αυτό το λόγο, μόλις η αίτηση υποβληθεί, αποστέλνεται ένα email προς το διαχειριστή ενημερώνοντάς τον ότι έχει γίνει αίτηση για νέο χρήστη, ο οποίος περιμένει να εγκριθεί. Μόνο όταν ο διαχειριστής εγκρίνει την αίτηση, το σύστημα επιτρέπει την είσοδο στο νέο χρήστη. Πιο κάτω παρατίθενται εικόνες από το πώς συμπληρώνεται το reCAPTCHA και από τα μηνύματα που παίρνει ο χρήστης με τη συμπλήρωση των στοιχείων.

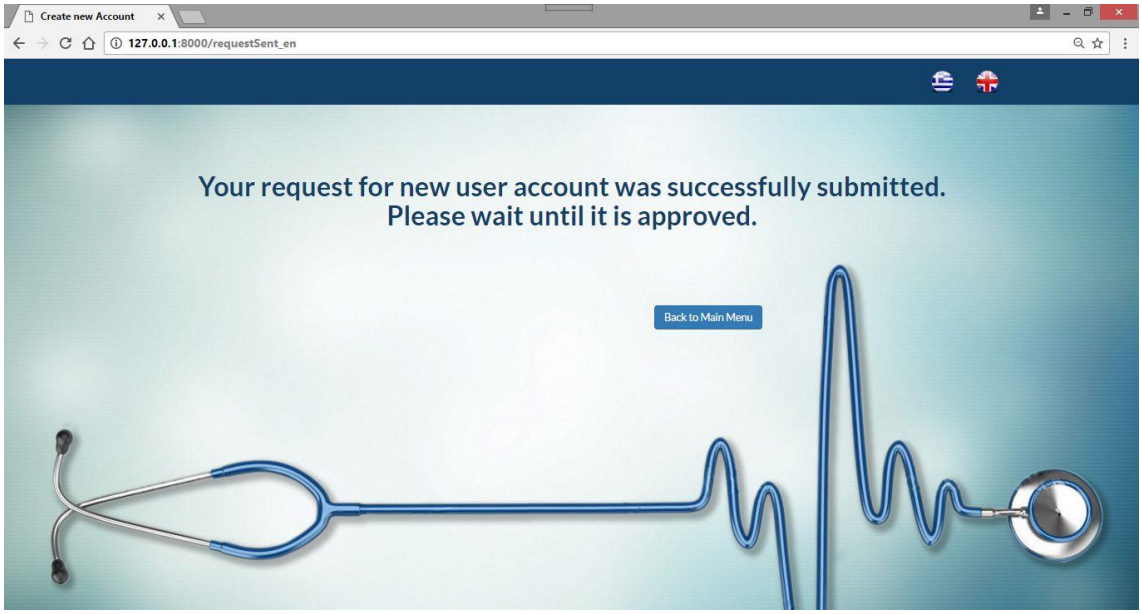


Σχήμα 5.2 – Ταυτοποίηση με το reCAPTCHA



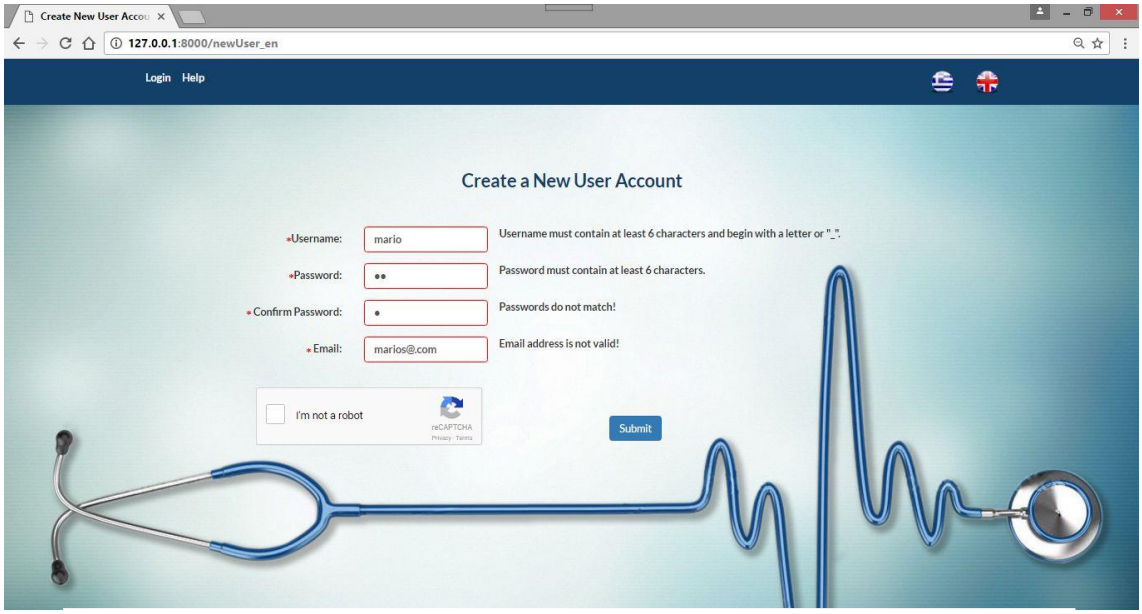
Σχήμα 5.3 – Σωστή συμπλήρωση όλων των πεδίων

Όταν ο χρήστης καταχωρήσει την αίτησή του στο σύστημα και εφόσον τα πεδία είναι σωστά συμπληρωμένα, τότε ο χρήστης μεταφέρεται σε δεύτερη σελίδα όπου ενημερώνεται σχετικά με την καταχώρηση της αίτησής του. Η σελίδα αυτή φαίνεται πιο κάτω στο Σχήμα 5.4. Παράλληλα με τη μεταφορά του χρήστη στη δεύτερη σελίδα αποστέλνεται και ένα μήνυμα στο email που καταχωρήθηκε στην αίτηση. Αυτό γίνεται για να γνωρίζει ο κάτοχος του email για την ενέργεια αυτή, σε περίπτωση που κάποιος άλλος χρησιμοποιήσει το email χωρίς να είναι δικό του.



Σχήμα 5.4 – Σελίδα με πληροφορίες σχετικά με την αίτηση του χρήστη

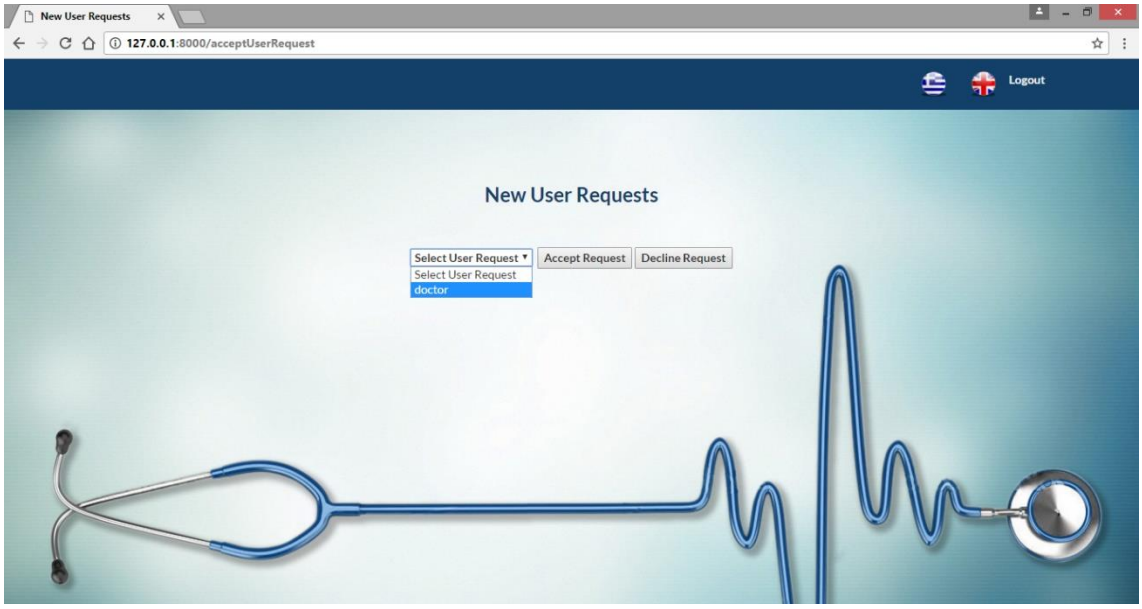
Στις πιο πάνω εικόνες παρουσιάζονται οι ενέργειες στις οποίες προβαίνει η ιστοσελίδα όταν τα στοιχεία που εισαγάγει ο χρήστης είναι ορθά. Σε περίπτωση, όμως, που κάποιο από τα πεδία συμπληρωθεί λανθασμένα, τότε η σελίδα ενημερώνει το χρήστη με ένα μήνυμα δίπλα από το πεδίο για τη λάθος συμπλήρωσή του. Τα μηνύματα αυτά παρουσιάζονται πιο κάτω στο Σχήμα 5.5.



Σχήμα 5.5 – Λανθασμένη συμπλήρωση πεδίων

Υπάρχει ακόμα μία περίπτωση κατά την οποία η σελίδα δε θα δεχτεί το αίτημα για νέο χρήστη, και αυτή είναι όταν το username που επέλεξε ο χρήστης βρίσκεται ήδη σε χρήση στο σύστημα από άλλο χρήστη. Σε τέτοια περίπτωση, η σελίδα θα ενημερώσει τον χρήστη που κάνει το αίτημα ώστε να επιλέξει διαφορετικό username.

Επιπρόσθετα με τις δύο προαναφερθείσες σελίδες, έχει γίνει υλοποίηση και μιας τρίτης σελίδας στην οποία θα έχει πρόσβαση ο διαχειριστής του συστήματος και μέσω της οποίας θα μπορεί να εγκρίνει ή να απορρίπτει τα αιτήματα για νέους χρήστες. Η λειτουργία αυτή υπάρχει ήδη υλοποιημένη στη σελίδα του /admin, αλλά λόγω του ότι εκεί εμφανίζονται όλοι οι χρήστες του συστήματος, θα ήταν πολύ δύσκολο για το διαχειριστή να ψάχνει σε όλα τα ονόματα ώστε να βρει τις νέες αιτήσεις και να τις εγκρίνει. Η σελίδα αυτή δείχνει στο διαχειριστή μόνο τις αιτήσεις για τις οποίες θα πρέπει να παρθεί απόφαση κατά πόσο θα εγκριθούν ή όχι. Η σελίδα αυτή φαίνεται στο Σχήμα 5.6.



Σχήμα 5.6 – Σελίδα για αποδοχή/απόρριψη αιτημάτων νέου χρήστη

5.4 Υλοποίηση κωδικοποίησης πληροφοριών στη βάση δεδομένων

Για την υλοποίηση αυτού του κομματιού έγινε χρήση τριών διαφορετικών αλγορίθμων για να επιτευχθεί η κωδικοποίηση των πληροφοριών. Αυτό έγινε για το λόγο ότι στη βάση μας υπάρχουν τρεις διαφορετικοί τύποι δεδομένων, οι οποίοι χρειάστηκε να κωδικοποιηθούν ο κάθε ένας με διαφορετικό τρόπο. Οι τύποι αυτοί είναι οι εξής:

1. Συμβολοσειρές (Strings)
2. Ακέραιοι αριθμοί (Integers)
3. Ημερομηνίες (Dates)

Για την κωδικοποίηση των συμβολοσειρών έγινε χρήση του αλγορίθμου AES (Advanced Encryption Standard) [22,23], γνωστός, επίσης, και ως Rijdael. Ο AES είναι αλγόριθμος κρυπτογράφησης συμμετρικού κλειδιού, δηλαδή, το κλειδί που χρησιμοποιείται για να γίνει η κρυπτογράφηση είναι το ίδιο κλειδί που χρησιμοποιείται και για την αποκρυπτογράφηση. Τα κλειδιά που χρησιμοποιούνται για την κρυπτογράφηση είναι μεγέθους 128 bits, 192 bits ή 256 bits και τα μπλοκ δεδομένων που μπορεί να επεξεργαστεί ο αλγόριθμος είναι μεγέθους 128 bits και αποτελούνται από ψηφία 0 και 1.

Ο αλγόριθμος είναι βασισμένος στο substitution-permutation network [24], το οποίο χρησιμοποιεί αντικαταστάσεις και μεταθέσεις των bits του μπλοκ δεδομένων ώστε να επιτευχθεί η κωδικοποίηση. Ανάλογα με το μέγεθος του κλειδιού κρυπτογράφησης είναι και

ο αριθμός των αντικαταστάσεων και μεταθέσεων. Για μέγεθος κλειδιού 128 bits πραγματοποιούνται 10 κύκλοι επαναλήψεων, για 192 bits πραγματοποιούνται 12 κύκλοι επαναλήψεων, ενώ για 256 bits πραγματοποιούνται 14 κύκλοι επαναλήψεων.

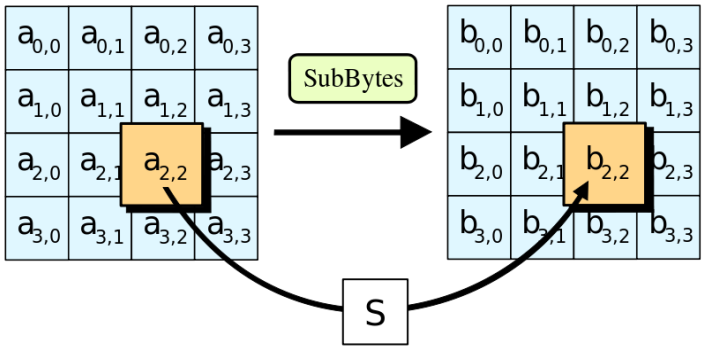
Για να επιτευχθεί η υλοποίηση του AES, το μπλοκ δεδομένων μεγέθους 128 bits=16 bytes μετατρέπεται σε ένα πίνακα διαστάσεων 4x4, καλούμενος state. Σε αυτό το σημείο, ο πίνακας αυτός περνά από διάφορα στάδια κωδικοποίησης ώστε να καταλήξει στο τέλος η κωδικοποιημένη συμβολοσειρά. Τα στάδια αυτά είναι 4 και είναι τα εξής:

1. KeyExpansions: Από το κλειδί κωδικοποίησης γίνεται παραγωγή κλειδιών (καλούμενα round keys) για κάθε γύρο κωδικοποίησης συν ένα επιπλέον. Δηλαδή γίνεται παραγωγή 11, 13 ή 15 κλειδιών, ανάλογα με το μέγεθος του κλειδιού κωδικοποίησης.

2. InitialRound: Γίνεται χρήση του αλγορίθμου AddRoundKey, στον οποίο κάθε byte του πίνακα state συνδυάζεται με ένα μπλοκ του round key μέσω δυαδικού XOR.

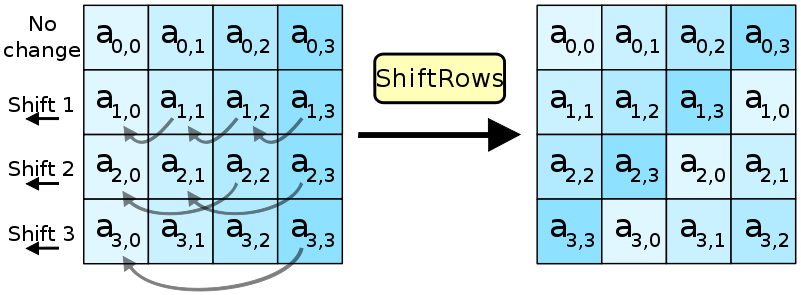
3. Rounds: Το στάδιο αυτό χωρίζεται σε 4 επί μέρους βήματα, κάθε ένα από τα οποία υλοποιεί μια ξεχωριστή λειτουργία στον πίνακα state. Τα βήματα είναι τα εξής:

    1) SubBytes: Βήμα στο οποίο κάθε byte του πίνακα state αντικαθίσταται με μη γραμμικό τρόπο με ένα άλλο byte σύμφωνα με ένα πίνακα αναζήτησης (lookup table).



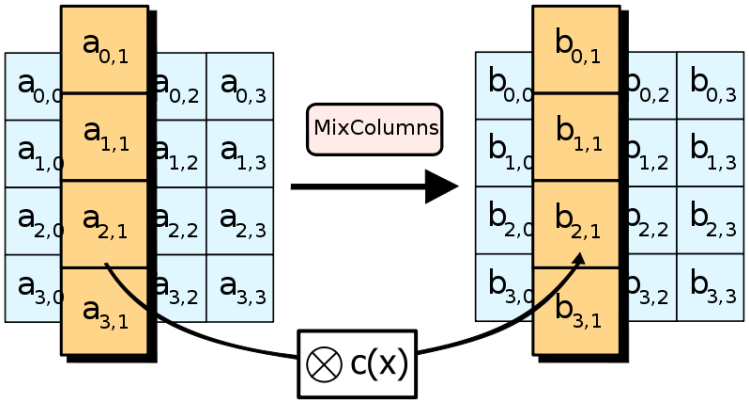
Σχήμα 5.7 – Σχηματική αναπαράσταση του SubBytes [25]

2) ShiftRows: Βήμα μεταφοράς των bytes στον πίνακα, όπου οι τρεις τελευταίες σειρές του πίνακα μετατοπίζονται κυκλικά σε ένα ορισμένο αριθμό βημάτων.



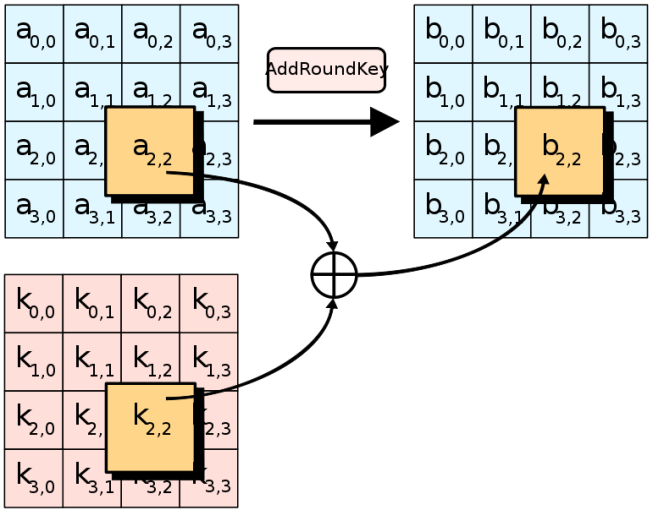
Σχήμα 5.8 – Σχηματική αναπαράσταση του ShiftRows [26]

3) MixColumns: Στο βήμα αυτό γίνεται σύνθεση των 4 στηλών του πίνακα, συνδυάζοντας τα 4 bytes κάθε στήλης με μια σταθερή συνάρτηση.



Σχήμα 5.9 – Σχηματική αναπαράσταση του MixColumns [27]

4) AddRoundKey: Στο βήμα αυτό γίνεται και η προσθήκη του round key στον πίνακα. Κάθε round key έχει το ίδιο μέγεθος με τον πίνακα και έτσι κάθε byte του κλειδιού συνδυάζεται με δυαδικό XOR με το αντίστοιχο byte του πίνακα state.



Σχήμα 5.10 – Σχηματική αναπαράσταση του AddRoundKey [28]

4. Final Round: Το στάδιο έχει τις ίδιες λειτουργίες με το προηγούμενο στάδιο με μόνη διαφορά το ότι δεν εκτελείται το MixColumns. Έχει δηλαδή τα εξής 3 βήματα:

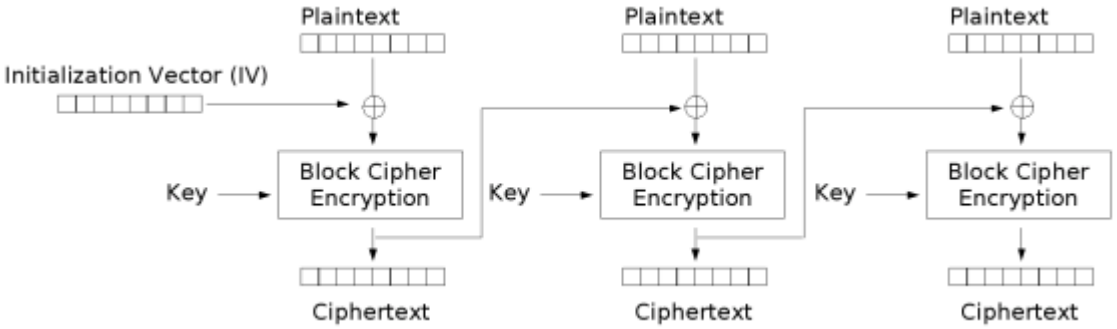
    1) SubBytes

    2) ShiftRows

    3) AddRoundKey

Στην εφαρμογή μας, για να επιτευχθεί η κρυπτογράφηση των συμβολοσειρών με τον AES, έγινε χρήση του πακέτου javax.crypto της Java και της κλάσης Cipher, η οποία υλοποιεί τον αλγόριθμο αυτό. Στην εφαρμογή μας, το μέγεθος κλειδιού που χρησιμοποιεί ο AES είναι μεγέθους 128 bits και για επιπλέον ασφάλεια γίνεται χρήση της πρακτικής του Cipher Block Chaining (CBC) [29]. Το CBC, με τη χρήση ενός initialization vector, κωδικοποιεί το ακρυπτογράφητο μπλοκ δεδομένων πριν αυτό δοθεί στον αλγόριθμο του AES. Για αυτό το λόγο, στην εφαρμογή μας γίνεται χρήση και ενός initialization vector μεγέθους 16 bytes.

Το CBC λειτουργεί με τον εξής τρόπο: Το initialization vector γίνεται XOR με το πρώτο από τα 128-bit μπλοκς του απλού κειμένου και το αποτέλεσμα δίνεται στον AES για κρυπτογράφηση. Το αμέσως επόμενο μπλοκ απλού κειμένου γίνεται XOR με το αμέσως προηγούμενο κρυπτογραφημένο μπλοκ. Η διαδικασία συνεχίζεται με τον ίδιο τρόπο για κάθε μπλοκ απλού κειμένου. Πιο κάτω στο Σχήμα 5.11 φαίνεται η σχηματική αναπαράσταση του CBC:



Σχήμα 5.11 – Σχηματική αναπαράσταση του Cypher Block Chaining [35]

Ακολούθως, για την κωδικοποίηση των ακεραίων χρησιμοποιήθηκε διαφορετικός αλγόριθμος από τον AES ώστε να επιτύχουμε την κωδικοποίηση των ακεραίων σε ακεραίους. Αυτό μας είναι χρήσιμο λόγω του ότι στη βάση δεδομένων μας η κρυπτογραφημένη τιμή θα αποθηκευτεί σε πεδίο τύπου integer. Ο αλγόριθμος που χρησιμοποιήθηκε είναι ο Feistel [30] και δέχεται σαν είσοδο ακέραιο τύπου long integer. Ο ακέραιος χωρίζεται στη μέση, σε δύο ακεραίους μεγέθους 32 bytes, και για κάθε μισό ακολουθείται ο ακόλουθος αλγόριθμος:

Αρχικά, ο ακέραιος αριθμός χωρίζεται σε δύο μισά, κάθε ένα μεγέθους 16 bytes. Τα δύο αυτά μισά (το αριστερό μισό και το δεξιό μισό του ακεραίου) χρησιμοποιούνται εναλλάξ σε κάθε γύρο κωδικοποίησης ώστε στο τέλος να προκύψει ο κωδικοποιημένος ακέραιος. Όσον αφορά τα βήματα που εκτελούνται σε κάθε γύρο κωδικοποίησης, έχουμε τα εξής:

1. Το μισό που αντιστοιχεί στο γύρο κωδικοποίησης γίνεται XOR με το round key που επίσης αντιστοιχεί στο γύρο κωδικοποίησης. Για κάθε γύρο κωδικοποίησης υπάρχει ένα ξεχωριστό round key.
2. Ο αριθμός που υπολογίστηκε στο Βήμα 1 υψώνεται στο τετράγωνο.
3. Το αριστερό μισό του αριθμού που υπολογίστηκε στο Βήμα 2 γίνεται XOR με το δεξιό μέρος.
4. Τα δύο μισά του αρχικού αριθμού, τα οποία έτυχαν και αλλαγής, ενώνονται ξανά και έχουμε τον κωδικοποιημένο ακέραιο αριθμό.

Για την αποκωδικοποίηση του αριθμού ακολουθείται ο ίδιος αλγόριθμος με είσοδο τον κρυπτογραφημένο αριθμό.

Τέλος, για την κωδικοποίηση των πεδίων ημερομηνίας υιοθετήθηκε η πρακτική της αφαίρεσης και πρόσθεσης ημερών στην αρχική ημερομηνία ώστε η ημερομηνία που θα αποθηκευτεί στη βάση δεδομένων να μην αντιστοιχεί στην πραγματική ημερομηνία.

5.5 Υλοποίηση συστήματος καταγραφής ενεργειών χρήστη (audit trail)

Το κομμάτι αυτό είναι υπεύθυνο να κρατά πληροφορίες που αφορούν τους χρήστες του συστήματος και τις ενέργειες που αυτοί έκαναν όσο ήταν συνδεδεμένοι σε αυτό. Τέτοιες πληροφορίες είναι το όνομα του χρήστη (username) που ενήργησε, η ημερομηνία και ώρα που πραγματοποιήθηκε η ενέργεια, η συσκευή από την οποία συνδέθηκε και άλλες πληροφορίες που αφορούν κυρίως την ασφάλεια του συστήματος. Περισσότερα για τις πληροφορίες αυτές αναφέρονται στη συνέχεια του υποκεφαλαίου αυτού.

Ως βάση του συστήματος αυτού χρησιμοποιήθηκε η εφαρμογή OpenATNA [31], η οποία έχει αναπτυχθεί από το Open Health Tools [31] ειδικά για την καταγραφή των κινήσεων των χρηστών σε ένα σύστημα. Το OpenATNA είναι εφαρμογή, η οποία δέχεται μηνύματα συγκεκριμένης μορφής και επεξεργαζόμενη τα πεδία που περιέχονται στα μηνύματα αυτά εξάγει πληροφορίες τις οποίες αποθηκεύει στη βάση δεδομένων. Οι πληροφορίες αυτές, σε μετέπειτα στάδιο, χρησιμοποιούνται ώστε να δημιουργηθούν αναφορές και έτσι να γίνει μια παρακολούθηση του συστήματος και της ασφάλειας που υπάρχει σε αυτό.

```
<AuditMessage>
    <EventIdentification EventActionCode="C" EventDateTime="2017-04-19 19:09:07" EventOutcomeIndicator="0">
        <EventID
            code="110100" codeSystemName="ICA" displayName="Application Activity" originalText="(optional)">
        </EventID>
        <EventTypeCode
            code="110120" codeSystemName="ICA" displayName="Insert in Cardiology" originalText="(optional)">
        </EventTypeCode>
    </EventIdentification>
    <ActiveParticipant
        UserID="doctor1" AlternativeUserID="doctor1@email.com" UserName="(Optional)"
        UserIsRequestor="True"
        NetworkAccessPointID="127.0.0.1" NetworkAccessPointTypeCode="2">
        <RoleIDCode
            code="110150" codeSystemName="ICA" displayName="Insert in Cardiology" originalText="(optional)">
        </RoleIDCode>
    </ActiveParticipant>
    <AuditSourceIdentification AuditEnterpriseSiteID="(optional)" AuditSourceID="127.0.0.1@THEOCHARIS">
        <AuditSourceTypeCode
            code="110170" codeSystemName="ICA" displayName="Application ID" originalText="(optional)">
        </AuditSourceTypeCode>
    </AuditSourceIdentification>
</AuditMessage>
```

Σχήμα 5.12 – Μήνυμα σε μορφή όπως ορίζεται από το RFC 3881 [32]

Όσον αφορά τα μηνύματα που δέχεται το OpenATNA, χρειάζεται να είναι στη μορφή που ορίζει το RFC 3881 [32]. Ένα παράδειγμα έγκυρου μηνύματος φαίνεται στο Σχήμα 5.12.

Το RFC 3881 [32] ορίζει πέντε τύπους, κάθε ένας από τους οποίους περιέχει τα δικά του πεδία. Οι πέντε αυτοί τύποι μαζί με τα πεδία που περιέχουν είναι οι εξής:

1. Event Identification: προσδιορίζει το όνομα, τον τύπο της ενέργειας, την ώρα και τη διάθεση του ελεγχόμενου συμβάντος. Περιέχει τα εξής πεδία:
    α. Event ID: Αναγνωριστικό για ένα συγκεκριμένο ελεγχόμενο συμβάν.
    β. Event Action Code: Ένα γράμμα, το οποίο δείχνει τον τύπο ενέργειας που εκτελέστηκε. C: Create, R: Read, U: Update, D: Delete, E: Execute
    γ. Event Date/Time: Ημερομηνία και ώρα που έγινε το συμβάν.
    δ. Event Outcome Indicator: Δείκτης που δείχνει κατά πόσο η ενέργεια ήταν επιτυχής ή όχι.
    ε. Event Type Code: Αναγνωριστικό για την κατηγορία του συμβάντος.
2. Active Participant Identification: Προσδιορίζει έναν χρήστη με σκοπό την τεκμηρίωση της ευθύνης για το ελεγχόμενο συμβάν. Ένας χρήστης μπορεί να είναι ένα άτομο ή μια συσκευή υλικού ή μια διαδικασία λογισμικού για γεγονότα που δεν έχουν ξεκινήσει από ένα άτομο. Προαιρετικά, μπορεί να καθοριστεί και η θέση πρόσβασης του χρήστη στο δίκτυο. Τα πεδία που περιέχει είναι τα εξής:
    α. User ID: Μοναδικό αναγνωριστικό για το χρήστη που λαμβάνει μέρος στο συμβάν.
    β. Alternative User ID: Εναλλακτικό αναγνωριστικό για το χρήστη που λαμβάνει μέρος στο συμβάν.
    γ. User Name: Το όνομα του χρήστη σε μορφή κατανοητή από τον άνθρωπο.
    δ. User is Requestor: Δείκτης για το αν ο χρήστης είναι ή δεν είναι ο αιτών ή ο εκκινητής του συμβάντος που ελέγχεται.
    ε. Role ID Code: Καθορισμός του ρόλου που παίζει ο χρήστης κατά την εκτέλεση του συμβάντος.

3. Network Access Point Identification: Περιέχεται μέσα στο Active Participant Identification (είναι 1:1 αντιστοιχία) και προσδιορίζει τη λογική θέση του χρήστη στο δίκτυο. Τα πεδία που περιέχει είναι τα εξής:

    α. Network Access Point Type Code: Αναγνωριστικό για τον τύπο του σημείου πρόσβασης στο δίκτυο από όπου ξεκίνησε το συμβάν ελέγχου. Υπάρχουν τρεις πιθανές τιμές που μπορεί να πάρει: 1: Όνομα μηχανής συμπεριλαμβανομένου του DNS, 2: IP Address, 3: Αριθμός τηλεφώνου.

    β. Network Access Point ID: Αναγνωριστικό για το σημείο πρόσβασης της συσκευής του χρήστη στο δίκτυο. Αυτό μπορεί να είναι ένα αναγνωριστικό συσκευής, μια διεύθυνση IP ή κάποιο άλλο αναγνωριστικό που σχετίζεται με μια συσκευή.

4. Audit Source Identification: Ο τύπος αυτός απαιτείται κυρίως για συστήματα εφαρμογών και διεργασιών. Μιας και οι εφαρμογές πολλαπλών βαθμίδων με κατανεμημένα ή σύνθετα στοιχεία καθιστούν διφορούμενη την αναγνώριση της πηγής, αυτή η συλλογή πεδίων μπορεί να επαναληφθεί για κάθε εφαρμογή ή διαδικασία που συμμετέχει ενεργά στην εκδήλωση. Περιέχει τα εξής πεδία:

    α. Audit Enterprise Site ID: Λογική θέση της πηγής (από όπου ξεκίνησε το συμβάν) εντός του δικτύου υγείας. Μπορεί να είναι ένα νοσοκομείο ή τοποθεσία ενός άλλου παρόχου.

    β. Audit Source ID: Αναγνωριστικό της πηγής από την οποία προέρχεται το συμβάν.

    γ. Audit Source Type Code: Κωδικός που καθορίζει τον τύπο της πηγής από την οποία προέκυψε το συμβάν.

5. Participant Object Identification: Τα δεδομένα του στοιχείου αυτού έχουν σαν στόχο να βοηθήσουν στη διαδικασία ελέγχου, υποδεικνύοντας συγκεκριμένες περιπτώσεις δεδομένων ή αντικειμένων τα οποία έτυχαν πρόσβασης. Είναι ένα προαιρετικό στοιχείο και τα δεδομένα του απαιτούνται μόνο εάν τα δεδομένα από το Event Identification, το Active Participant Identification και το Audit Source Identification δεν επαρκούν για την τεκμηρίωση του συμβάντος. Περιέχει τα εξής πεδία:

α. Participant Object Type Code: Κωδικός για τον τύπο αντικειμένου που συμμετέχει στο συμβάν και το οποίο ελέγχεται.

β. Participant Object Type Code Role: Κωδικός που αντιπροσωπεύει τον ρόλο της λειτουργικής εφαρμογής του συμμετέχοντος αντικειμένου που ελέγχεται.

γ. Participant Object Data Life Cycle: Αναγνωριστικό για το στάδιο κύκλου ζωής δεδομένων για το αντικείμενο του συμμετέχοντα. Αυτό μπορεί να χρησιμοποιηθεί για την παροχή μιας διαδρομής ελέγχου για δεδομένα, με την πάροδο του χρόνου, καθώς περνά μέσα από το σύστημα.

δ. Participant Object ID Type Code: Περιγράφει το αναγνωριστικό που περιέχεται στο Participant Object ID.

ε. Participant Object Sensitivity: Δηλώνει ευαισθησία καθορισμένη από την πολιτική για το Participant Object ID, όπως VIP, κατάσταση HIV, κατάσταση ψυχικής υγείας ή παρόμοια θέματα.

στ. Participant Object ID: Προσδιορίζει μια συγκεκριμένη εμφάνιση του αντικειμένου του συμμετέχοντα.

ζ. Participant Object Name: Ένας συγκεκριμένος περιγραφικός δείκτης του Participant Object ID που καταγράφεται. Παράδειγμα είναι όνομα κάποιου ατόμου.

η. Participant Object Query: Το πραγματικό query για ένα Participant Object.

θ. Participant Object Detail: Λεπτομέρειες σχετικά με το αντικείμενο που χρησιμοποιήθηκε ή που έτυχε πρόσβασης.

Η λειτουργία του audit trail έχει υλοποιηθεί για το login και logout του γιατρού από το σύστημα όπως, επίσης, και για τις τρεις βασικές λειτουργείες του συστήματος, δηλαδή για την εισαγωγή, αναζήτηση και ενημέρωση των πληροφοριών ενός ασθενή. Όταν ένας γιατρός θελήσει, για παράδειγμα, να προσθέσει έναν ασθενή στο σύστημα, τότε μαζί με τις πληροφορίες του ασθενή το σύστημα κρατά την ώρα που έγινε η εισαγωγή, το username του γιατρού, το κατά πόσο η εισαγωγή του ασθενή στη βάση έγινε με επιτυχία

ή αν υπήρξε κάποιο σφάλμα, όπως επίσης, και την ενέργεια που έγινε από τον γιατρό (στην περίπτωση αυτή είναι η εισαγωγή ασθενή στο σύστημα). Επιπλέον, κρατείται η τοποθεσία από την οποία ο γιατρός συνδέθηκε στο σύστημα (URL ή όνομα της συσκευής, π.χ. doctor1@DoctorPC) και η IP διεύθυνση της συσκευής.

Όλες οι πιο πάνω πληροφορίες οργανώνονται ώστε να σχηματίσουν το μήνυμα (βλ. Σχήμα 5.12), όπως ορίζεται από το RFC 3881 [32]. Ακολούθως, το μήνυμα αποστέλνεται μέσω socket στην εφαρμογή του OpanATNA, η οποία είναι υπεύθυνη να αναλύσει το μήνυμα και να αποθηκεύσει τις πληροφορίες στη βάση δεδομένων.

Εν συνεχεία, σε περίπτωση που ο διαχειριστής του συστήματος θελήσει να δει τις ενέργειες που έγιναν στο σύστημα, επικοινωνεί με την εφαρμογή του OpenATNA ζητώντας τα πεδία που θέλει να δει και η εφαρμογή, μέσα από τη βάση δεδομένων, του επιστρέφει όλες τις διαθέσιμες πληροφορίες.

5.6 Επιπλέον ασφάλεια που παρέχεται από το σύστημα

Εκτός από τις τρεις προαναφερθείσες υλοποιήσεις ασφάλειας του συστήματος, έγινε χρήση και μερικών άλλων πρακτικών ώστε να αποφευχθούν τόσο εσωτερικές όσο και εξωτερικές επιθέσεις προς το σύστημα. Πιο συγκεκριμένα, το σύστημα υλοποιεί prepared statements [14] για την αποφυγή επιθέσεων SQL Injection, όπως επίσης, κάνει χρήση CSRF token για κάθε χρήστη, ώστε να αποφευχθούν επιθέσεις Cross-Site Request Forgery προς το σύστημα.

Κεφάλαιο 6

Αξιολόγηση και Αξιοπιστία Εφαρμογής

6.1 Εισαγωγή

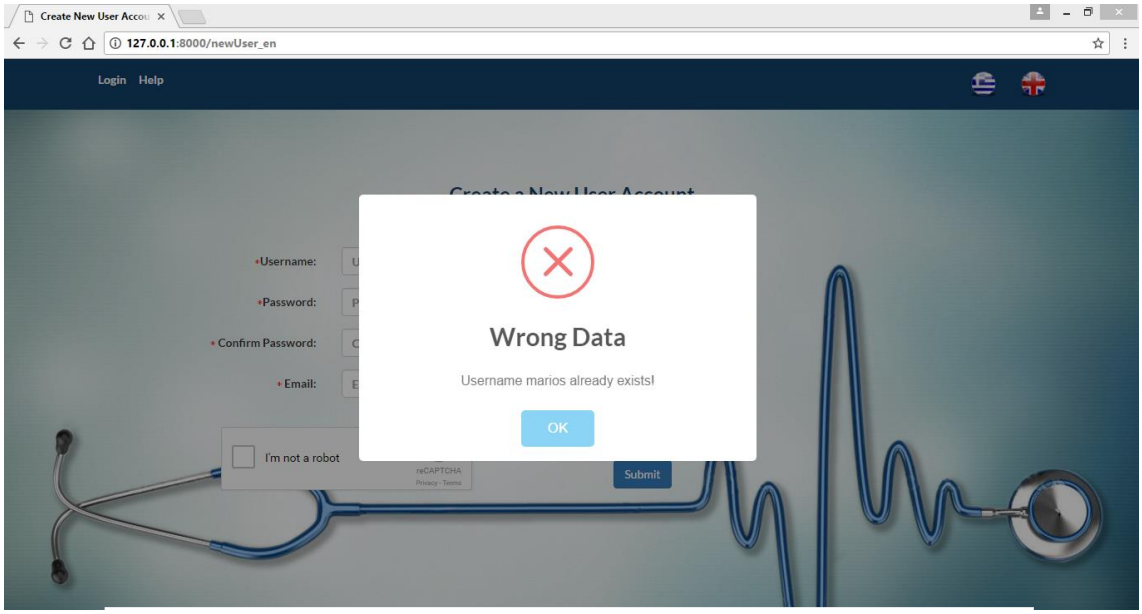
Φτάνοντας στο προτελευταίο στάδιο του κύκλου ανάπτυξης της εφαρμογής, το οποίο είναι η εξέταση και ο έλεγχός της, καλούμαστε να αξιολογήσουμε την εφαρμογή και να δούμε αν όσα υλοποιήθηκαν πληρούν τους στόχους που θέσαμε στην αρχή του κύκλου αυτού. Σημαντικό ρόλο στην εργασία αυτή έχει, βεβαίως, και η αξιοπιστία της εφαρμογής, μιας και το θέμα της ασχολείται με την ασφάλεια ευαίσθητων δεδομένων. Η αξιολόγηση έγινε με βάση τις πιθανές εισόδους δεδομένων του χρήστη στο σύστημα ώστε, με αυτόν τον τρόπο, να δούμε αν τα διάφορα κομμάτια που υλοποιήθηκαν λειτουργούν σωστά και παράγουν τα αναμενόμενα αποτελέσματα.

6.2 Έλεγχος αξιοπιστίας σελίδας νέου χρήστη

Σε αυτό το κομμάτι έγινε έλεγχος της νέας σελίδας που δημιουργήθηκε και το κατά πόσο ο νέος χρήστης μπορεί με κάποιο τρόπο να "ξεγελάσει" το σύστημα ώστε να έχει πρόσβαση στα δεδομένα χωρίς να πρέπει. Πιο συγκεκριμένα έγιναν οι εξής ελέγχοι:

1. <u>Έλεγχος για εισαγωγή ορθών και επιτρεπτών δεδομένων σε όλα τα πεδία:</u> Σε αυτή την περίπτωση ο νέος χρήστης δημιουργείται στο σύστημα. Παρ' όλα αυτά, αν προσπαθήσει την ίδια χρονική στιγμή να εισέλθει στο σύστημα, πριν ακόμα εγκριθεί από το διαχειριστή, τότε δε θα του επιτραπεί η είσοδος.

2. Έλεγχος για εισαγωγή username που υπάρχει ήδη στο σύστημα: Σε αυτή την περίπτωση το σύστημα θα ενημερώσει τον χρήστη ότι το username που ζητά είναι ήδη σε χρήση από το σύστημα και θα τον επαναφέρει στην ίδια σελίδα ώστε να επιλέξει ένα διαφορετικό username. Με αυτό τον τρόπο το σύστημα αποτρέπει τη ύπαρξη διπλών usernames. Παράδειγμα μηνύματος για ήδη υπάρχον username φαίνεται στο Σχήμα 6.1, όπου το ήδη υπάρχον username είναι το "marios".



Σχήμα 6.1 – Μήνυμα λάθους για ήδη υπάρχον username

3. Έλεγχος για εισαγωγή μη επιτρεπτών δεδομένων εισόδου στα πεδία της σελίδας: Σε αυτό τον έλεγχο έγινε εισαγωγή δεδομένων στα πεδία, τα οποία δεν υποστηρίζονται από το σύστημα. Κάποια παραδείγματα είναι τα εξής:

    α. Εισαγωγή αριθμού στο πεδίο του username.

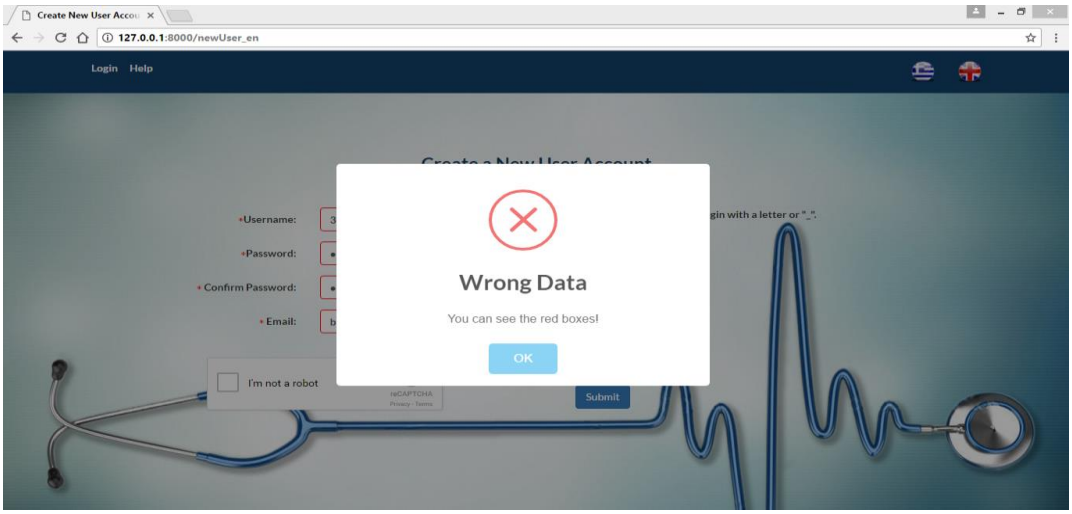
    β. Εισαγωγή κωδικού μήκους μικρότερου από 6 χαρακτήρες.

    γ. Εισαγωγή διαφορετικών κωδικών στα πεδία του password και confirmation password.

    δ. Εισαγωγή δεδομένων στο πεδίο του Email το οποίο δεν ήταν της μορφής email@provider.domain1(.domain2) (π.χ. doctor1@gmail.com).

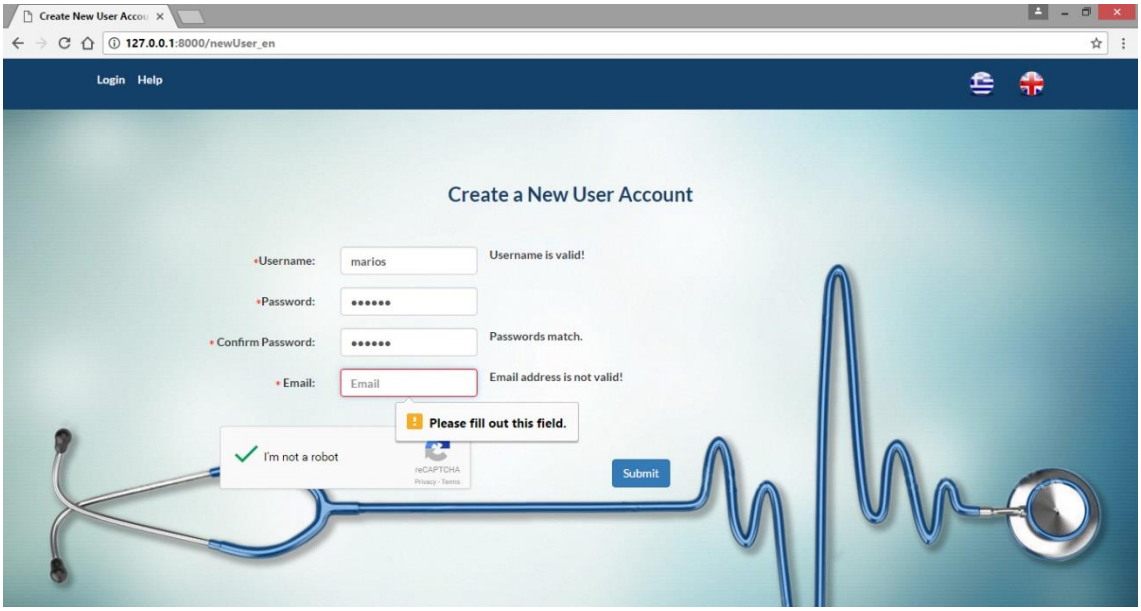
Για όλους τους πιο πάνω ελέγχους η σελίδα ενημερώνει τον χρήστη για το ότι τα δεδομένα που προσπαθεί να εισαγάγει δεν είναι επιτρεπτά. Δίπλα από κάθε πεδίο

εμφανίζεται το αντίστοιχο μήνυμα ώστε να καθοδηγήσει τον χρήστη να διορθώσει την είσοδό του. Σε περίπτωση που ο χρήστης επιλέξει να καταχωρήσει τη φόρμα με τα πεδία συμπληρωμένα λάθος, τότε λαμβάνει μήνυμα λάθους από τη σελίδα. Το μήνυμα της σελίδας φαίνεται στο Σχήμα 6.2.



Σχήμα 6.2 – Μήνυμα λάθους για λάθος συμπληρωμένα πεδία

Επιπλέον, αν κάποιο από τα πεδία δεν έχει συμπληρωθεί καθόλου, τότε η σελίδα θα ειδοποιήσει τον χρήστη να το συμπληρώσει προτού προχωρήσει στην καταχώρηση της φόρμας. Παράδειγμα τέτοιου μηνύματος φαίνεται στο Σχήμα 6.3, όπου παραμένει κενό το πεδίο του Email.



Σχήμα 6.3 – Ειδοποίηση ότι κάποιο πεδίο έμεινε κενό

4. Έλεγχος για δεδομένα που μπορεί να προκαλέσουν SQL Injection: Σε αυτό τον έλεγχο έγινε δοκιμή κατά πόσο το σύστημα θα αποδεχτεί usernames της μορφής «hi' or 1=1--». Usernames της προαναφερθείσας μορφής δε γίνονται αποδεχτά μιας και το πεδίο Username της ιστοσελίδας δέχεται συμβολοσειρές οι οποίες να ξεκινούν με γράμμα ή '_' και να συνεχίζουν με γράμμα, αριθμό ή '_'. Επομένως, συμβολοσειρές που περιέχουν μονά εισαγωγικά και κενά (white spaces) δε γίνονται αποδεχτά.

6.3 Έλεγχος αξιοπιστίας κρυπτογραφημένων δεδομένων

Σε αυτό το υποκεφάλαιο αναφέρεται ο έλεγχος που έγινε στο κομμάτι της υλοποίησης της κρυπτογράφησης. Έγινε ένας αριθμός από ελέγχους στην εφαρμογή με διάφορα δεδομένα εισόδου, για να εξεταστεί το κατά πόσο τα δεδομένα κρυπτογραφούνται σωστά στη βάση και όταν ανακτώνται και γίνεται η αποκρυπτογράφηση, τότε είναι σωστά. Για αυτό το λόγο, οι αλγόριθμοι κρυπτογράφησης εφαρμόστηκαν σε όλα τα πεδία της βάσης, ασχέτως αν αυτά ήταν ευαίσθητα προσωπικά δεδομένα ή όχι.

1. Έλεγχος με δεδομένα κανονικού μεγέθους: Σε αυτόν τον έλεγχο η εφαρμογή ελέγχθηκε στο κατά πόσο αν ένας γιατρός εισάγει δεδομένα εντός των επιτρεπτών ορίων (π.χ. όνομα που να μην ξεπερνά τους 15 χαρακτήρες, αριθμό τηλεφώνου με 8 ψηφία κλπ), αν θα κρυπτογραφηθούν και αποκρυπτογραφηθούν ορθά. Μετά τον έλεγχο, όλα τα δεδομένα ήταν κρυπτογραφημένα μέσα στη βάση δεδομένων και όταν έγινε ανάκτησή τους, παρουσιάστηκαν αποκρυπτογραφημένα σωστά.

2. Έλεγχος με δεδομένα αρκετά μεγάλου μήκους: Ο έλεγχος αυτός είχε σαν στόχο να ελέγξει αν η εφαρμογή μπορεί να κρυπτογραφήσει δεδομένα μεγάλου μήκους εντός της βάσης δεδομένων. Σαν δεδομένα εισόδου δοκιμάστηκαν μεγάλες συμβολοσειρές και ακέραιοι με μεγάλο αριθμό ψηφίων. Οι ημερομηνίες δε δοκιμάστηκαν λόγω του ότι η μορφή των δεδομένων αυτών είναι σταθερή. Ο έλεγχος έδειξε ότι η εφαρμογή έχει τη δυνατότητα να κρυπτογραφήσει συμβολοσειρές με μήκος μεγαλύτερο από αυτό που θα χρειαζόταν υπό κανονικές συνθήκες το συγκεκριμένο πεδίο, όπως επίσης και ακεραίους με αριθμό ψηφίων μέχρι 8. Για παράδειγμα, στο πεδίο του ονόματος του ασθενή, είναι δυνατό να γραφτεί όνομα 30 χαρακτήρων και να αποθηκευτεί κρυπτογραφημένο στη βάση. Επίσης, όσον αφορά τους ακεραίους, ο μεγαλύτερος ακέραιος που θα χρειαστεί να εισαχθεί στο

σύστημα είναι το τηλέφωνο του ασθενή, το οποίο δε θα χρειαστεί περισσότερα από 8 ψηφία, έχοντας υπόψιν τους αριθμούς τηλεφώνων στην Κύπρο.

6.4 Έλεγχος συστήματος καταγραφής ενεργειών χρήστη

Αυτό το κομμάτι είχε σαν στόχο να ελέγξει κατά πόσο τα όσα υλοποιήθηκαν και αφορούν το κομμάτι του audit trail είναι ορθά. Στην παρούσα φάση παρουσιάστηκε ένα τεχνικό πρόβλημα, το οποίο δεν επέτρεψε την εγκατάσταση της εφαρμογής του OpenATNA στον υπολογιστή του συγγραφέως. Κατά πάσα πιθανότητα, το πρόβλημα οφειλόταν στο ότι η εφαρμογή του OpenATNA χρησιμοποιεί κάποιες πιο παλιές βιβλιοθήκες λογισμικού οι οποίες δεν είναι συμβατές με το λειτουργικό σύστημα των Windows 8.1. Δοκιμάστηκε, επίσης, και η εγκατάσταση της εφαρμογής σε λειτουργικό σύστημα Ubuntu 16.04, όπου και πάλι δεν ήταν εφικτή η εγκατάσταση. Για αυτό το λόγο, και μέσω επικοινωνίας με τον κύριο Ιωάννη Κωνσταντίνου, ερευνητή του Πανεπιστημίου μας στο εργαστήριο του e-Health, έγινε χρήση της υλοποίησης του OpenATNA Client, υλοποιημένη από την ομάδα του e-Health και η οποία έχει παραδείγματα μηνυμάτων τα οποία είναι συμβατά με την εφαρμογή του OpenATNA.

Κάνοντας χρήση αυτών των μηνυμάτων, έγινε σύγκριση με τα μηνύματα που παράγει η δική μας εφαρμογή. Πιο κάτω φαίνεται στο Σχήμα 6.4 ένα από τα συμβατά μηνύματα που χρησιμοποιήθηκαν από την ομάδα του e-Health για τον έλεγχο της εφαρμογής και στο Σχήμα 6.5 φαίνεται ένα από τα μηνύματα που δημιουργείται από την εφαρμογή μας για το συμβάν εισαγωγής ασθενή.

```
String dateStr = ISO8601Local.format (new Date());
String xml =
        "<AuditMessage>"+
        "    <EventIdentification EventOutcomeIndicator=\"0\" EventDateTime=\"" + dateStr + "\" EventActionCode=\"R\">"+
        "        <EventID codeSystemName=\"DCM\" displayName=\"Export\" code=\"110106\"/>"+
        "        <EventTypeCode codeSystemName=\"IHE Transactions\" displayName=\"Retrieve Document Set\" code=\"ITI-43\"/>"+
        "    </EventIdentification>"+
        "    <ActiveParticipant NetworkAccessPointTypeCode=\"2\" NetworkAccessPointID=\"129.162.101.164\" \
          UserIsRequestor=\"false\" UserID=\"http://www.w3.org/2005/08/addressing/anonymous\">"+
        "        <RoleIDCode codeSystemName=\"DCM\" displayName=\"Source\" code=\"110153\"/>"+
        "    </ActiveParticipant>"+
        "    <ActiveParticipant NetworkAccessPointTypeCode=\"2\" NetworkAccessPointID=\"129.162.101.116\" \
          UserIsRequestor=\"true\" UserID=\"http://localhost:8020/axis2/services/xdsrepositoryb\">"+
        "        <RoleIDCode codeSystemName=\"DCM\" displayName=\"Destination\" code=\"110152\"/>"+
        "    </ActiveParticipant>"+
        "    <AuditSourceIdentification AuditSourceID=\"MISYS\" AuditEnterpriseSiteID=\"MISYS\"/>"+
        "    <ParticipantObjectIdentification ParticipantObjectTypeCodeRole=\"3\" ParticipantObjectTypeCode=\"2\" \
          ParticipantObjectID=\"2.16.840.1.113883.3.65.2.1266421824214\">"+
        "        <ParticipantObjectIDTypeCode code=\"9\"/>"+
        "        <ParticipantObjectDetail value=\"MS4zLjYuMS40LjEuMjEzNjcuMjAxMC4xLjIuMTEyNQ==\" type=\"RepositoryUniqueId\"/>"+
        "    </ParticipantObjectIdentification>"+
        "</AuditMessage>";
```

Σχήμα 6.4 – Μήνυμα συμβατό με την εφαρμογή του OpenATNA

```
<AuditMessage>
    <EventIdentification EventActionCode="C" EventDateTime="2017-05-23 12:17:01" EventOutcomeIndicator="0">
        <EventID code="110100" codeSystemName="ICA" displayName="Insertion"></EventID>
        <EventTypeCode code="110120" codeSystemName="ICA" displayName="Insert in Cardiology"></EventTypeCode>
    </EventIdentification>
    <ActiveParticipant UserID="doctor1" AlternativeUserID="d@d.com" UserIsRequestor="True"
        NetworkAccessPointID="127.0.0.1" NetworkAccessPointTypeCode="2">
        <RoleIDCode code="110150" codeSystemName="ICA" displayName="Insert in Cardiology"></RoleIDCode>
    </ActiveParticipant>
    <AuditSourceIdentification AuditSourceID="127.0.0.1@THEOCHARIS">
        <AuditSourceTypeCode code="110170" codeSystemName="DCM" displayName="Application ID"></AuditSourceTypeCode>
    </AuditSourceIdentification>
    <ParticipantObjectIdentification ParticipantObjectID="952056" ParticipantObjectTypeCode="1" ParticipantObjectTypeCodeRole="1">
        <ParticipantObjectIDTypeCode code="2"/>
        <ParticipantObjectName>PatientIdentifier</ParticipantObjectName>
        <ParticipantObjectDetail type="CardiologyCheckUp" value="VHVlIE1heSAyMyAwMDoxNzowMSBFRVNUIDIwMTc="/>
    </ParticipantObjectIdentification>
</AuditMessage>
```

Σχήμα 6.5 – Μήνυμα που παράγεται από τη δική μας εφαρμογή

Όπως φαίνεται από τα δύο πιο πάνω Σχήματα και μελετώντας τους κανόνες που ορίζονται στο RFC 3881 [32], τα μηνύματα που δημιουργούνται από την εφαρμογή μας θα είναι συμβατά με την εφαρμογή του OpenATNA. Το μήνυμα έχει ακολουθήσει τη δομή του μηνύματος που είχαμε σαν πρότυπο με μόνη διαφορά το ότι απουσιάζει το πεδίο του AuditEnterpriseSiteID από το AuditSourceIdentification. Ο λόγος είναι πως το πεδίο αυτό είναι προαιρετικό και δηλώνει την τοποθεσία από όπου γίνεται η ενέργεια (π.χ. νοσοκομείο, κλινική κλπ) και αφού οι γιατροί θα ενεργούν από συγκεκριμένη σταθερή τοποθεσία, δε θα χρειαστεί να κρατείται η τοποθεσία αυτή από το σύστημα.

Κεφάλαιο 7

Συμπεράσματα και Μελλοντική Εργασία

7.1 Συμπεράσματα

Με την ολοκλήρωση της εργασίας που μου ανατέθηκε ως μέρος της εκπόνησης της διπλωματικής μου εργασίας, έχει επιτευχθεί η αύξηση του επιπέδου ασφαλείας γύρω από την εφαρμογή ηλεκτρονικής καταγραφής ασθενών καρδιολογικού τμήματος. Αυτό επιτεύχθηκε μέσω της κρυπτογράφησης των δεδομένων που βρίσκονται αποθηκευμένα μέσα στη βάση δεδομένων, μέσα από την καταγραφή των συμβάντων που γίνονται στο σύστημα και την τήρηση audit trail, όπως επίσης και με την ασφαλή δημιουργία των χρηστών στο σύστημα κάνοντας χρήση διαφόρων τεχνικών ασφαλείας.

Οι πιο πάνω υλοποιήσεις αποτελούν κύριο κομμάτι εφαρμογών που ασχολούνται με καταγραφή πληροφοριών για ασθενείς. Γνωρίζοντας αυτό, έχει τεθεί σαν στόχος να γίνει η εγκατάστασή τους και σε άλλες εφαρμογές τέτοιου τύπου, οι οποίες στερούνται αυτού του είδους την ασφάλεια.

7.2 Μελλοντική Εργασία

Λόγω της μεγάλης έκτασης που καλύπτει ο τομέας της ασφάλειας ηλεκτρονικών εφαρμογών, αναφερόμενος κάποιος στον τομέα αυτό, εμπεριέχει μέσα στον όρο ένα μεγάλο φάσμα από ενέργειες που μπορούν να γίνουν ώστε να πραγματωθεί η ασφάλεια μας εφαρμογής. Η παρούσα εργασία ασχολήθηκε μόνο με ένα μικρό κομμάτι της ασφάλειας, επομένως υπάρχουν αρκετές εισηγήσεις για επέκταση της παρούσας υλοποίησης:

► **Ασφάλεια από επιθέσεις προερχόμενες από το Διαδίκτυο:**

Τέτοια ασφάλεια μπορεί να είναι η δημιουργία ασφαλούς καναλιού SSH μεταξύ του συστήματος και των γιατρών ώστε να αποτρέπεται η υποκλοπή των πληροφοριών που στέλνονται μεταξύ των συσκευών. Επιπλέον, σε αυτό το κομμάτι της ασφάλειας, μπορεί να προστεθεί και η εφαρμογή ασφάλειας από ιούς και διάφορα κακά λογισμικά που προσβάλλουν εφαρμογές μέσω Διαδικτύου.

► **Λειτουργία επανακρυπτογράφησης των πληροφοριών στη βάση δεδομένων:**

Λόγω του ότι οι αλγόριθμοι κρυπτογράφησης κάνουν χρήση συγκεκριμένου κλειδιού για να κωδικοποιηθούν τα δεδομένα στη βάση, μια εισήγηση θα ήταν κατά τακτά χρονικά διαστήματα να γίνεται αλλαγή του κλειδιού αυτού. Αυτό συνεπάγεται, όμως, και το ότι θα χρειαστεί τα ήδη υπάρχοντα δεδομένα στη βάση να αποκρυπτογραφηθούν με τη χρήση του παλιού κλειδιού και να κωδικοποιηθούν με το νέο.

► **Τήρηση αντιγράφων ασφαλείας**

Η εφαρμογή ενός χρονικού προγράμματος κατά το οποίο κατά τακτά χρονικά διαστήματα θα γίνεται αντιγραφή των υπαρχόντων πληροφοριών του συστήματος σε δευτερεύον μέσο αποθήκευσης. Με αυτό τον τρόπο θα υπάρχει η δυνατότητα, σε περίπτωση που χαθούν τα δεδομένα του συστήματος, να ανακτηθεί ένα μεγάλο μέρος (αν όχι ολόκληρο) από τα αντίγραφα ασφαλείας.

► **Χρήση ρόλων για τον κάθε γιατρό**

Το σύστημα μπορεί να επεκταθεί προσθέτοντας ρόλους στον κάθε γιατρό ξεχωριστά, ώστε να μην έχουν πρόσβαση όλοι οι γιατροί στα δεδομένα όλων των ασθενών αλλά μόνο σε όσα χρειάζεται να γνωρίζουν.

Λίστα Βιβλιογραφίας:

[1] «Wikipedia – Computer Security», Διαθέσιμο από:
https://en.wikipedia.org/wiki/Computer_security
[Διαβάστηκε τελευταία φορά το Δεκέμβριο του 2016]

[2] «OWASP – 2013 Top 10 List», Διαθέσιμο από:
https://www.owasp.org/index.php/Top_10_2013-Top_10
[Διαβάστηκε τελευταία φορά το Νοέμβριο του 2016]

[3] «OWASP – Testing Guide version 4», Διαθέσιμο από:
https://www.owasp.org/images/1/19/OTGv4.pdf
[Διαβάστηκε τελευταία φορά το Νοέμβριο του 2016]

[4] «Wikipedia – Attack (computing)», Διαθέσιμο από:
https://en.wikipedia.org/wiki/Attack_(computing)#Types_of_Attack
[Διαβάστηκε τελευταία φορά το Δεκέμβριο του 2016]

[5] «OWASP – Top 10 2004», Διαθέσιμο από:
https://www.owasp.org/index.php/Top_10_2004
[Διαβάστηκε τελευταία φορά το Δεκέμβριο του 2016]

[6] «OWASP – Top 10 2007», Διαθέσιμο από:
https://www.owasp.org/index.php/Top_10_2007
[Διαβάστηκε τελευταία φορά το Δεκέμβριο του 2016]

[7] «OWASP –Top 10 2010 - Main», Διαθέσιμο από:
https://www.owasp.org/index.php/Top_10_2010-Main
[Διαβάστηκε τελευταία φορά το Δεκέμβριο του 2016]

[8] «OWASP - 2013 Top 10 List», Διαθέσιμο από:
https://www.owasp.org/index.php/Top_10_2013-Top_10
[Διαβάστηκε τελευταία φορά το Νοέμβριο του 2016]

[9] «Wikipedia - Cross-site scripting», Διαθέσιμο από:
https://el.wikipedia.org/wiki/Cross-site_scripting
[Διαβάστηκε τελευταία φορά τον Ιανουάριο του 2017]

[10] «JetBrains Strikes Python Developers with PyCharm 1.0 IDE», Διαθέσιμο από: http://www.eweek.com/c/a/Application-Development/JetBrains-Strikes-Python-Developers-with-PyCharm-10-IDE-304127
[Διαβάστηκε τελευταία φορά το Φεβρουάριο του 2017]

[11] «Eclipse Tutorial», Διαθέσιμο από: https://www.tutorialspoint.com/eclipse/
[Διαβάστηκε τελευταία φορά το Φεβρουάριο του 2017]

[12] «MySQL Workbench», Διαθέσιμο από: https://dev.mysql.com/doc/workbench/en/
[Διαβάστηκε τελευταία φορά το Φεβρουάριο του 2017]

[13] «Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet», Διαθέσιμο από: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet
[Διαβάστηκε τελευταία φορά το Φεβρουάριο του 2017]

[14] «SQL Injection Prevention Cheat Sheet - Defense Option 1: Prepared Statements (with Parameterized Queries)», Διαθέσιμο από: https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet
[Διαβάστηκε τελευταία φορά το Φεβρουάριο του 2017]

[15] «Django (web framework)», Διαθέσιμο από: https://en.wikipedia.org/wiki/Django_(web_framework)
[Διαβάστηκε τελευταία φορά το Φεβρουάριο του 2017]

[16] «The 9 Most In-Demand Programming Languages of 2017», Διαθέσιμο από: http://www.codingdojo.com/blog/9-most-in-demand-programming-languages-of-2017/
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]

[17] «Top 20 Most Popular Programming Languages In 2017», Διαθέσιμο από: https://www.techworm.net/2017/03/top-20-popular-programming-languages-2017.html
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]

[18] «Apache Tomcat», Διαθέσιμο από: http://tomcat.apache.org/
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]

[19] «Redhat Hibernate», Διαθέσιμο από: http://hibernate.org/
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]

[20] «Security Audit and Access Accountability Message XML Data Definitions for Healthcare Applications», Διαθέσιμο από:
https://tools.ietf.org/html/rfc3881
[Διαβάστηκε τελευταία φορά το Μάρτιο του 2017]
[21] «Audit Trail and Node Authentication», Διαθέσιμο από:
http://wiki.ihe.net/index.php/Audit_Trail_and_Node_Authentication
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[22] «Announcing the Advanced Encryption Standard (AES)», Διαθέσιμο από:
http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[23] «Advanced Encryption Standard», Διαθέσιμο από:
https://en.wikipedia.org/wiki/Advanced_Encryption_Standard
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[24] «Substitution-permutation network», Διαθέσιμο από:
https://en.wikipedia.org/wiki/Substitution-permutation_network
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[25] «File:AES-SubBytes.svg», Διαθέσιμο από:
https://en.wikipedia.org/wiki/File:AES-SubBytes.svg
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[26] «File:AES-ShiftRows.svg», Διαθέσιμο από:
https://en.wikipedia.org/wiki/File:AES-ShiftRows.svg
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[27] «File:AES-MixColumns.svg», Διαθέσιμο από:
https://en.wikipedia.org/wiki/File:AES-MixColumns.svg
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[28] «File:AES-AddRoundKey.svg», Διαθέσιμο από:
https://en.wikipedia.org/wiki/File:AES-AddRoundKey.svg
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[29] «Block cipher mode of operation», Διαθέσιμο από:
https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]

[30] «Feistel cipher», Διαθέσιμο από:
https://en.wikipedia.org/wiki/Feistel_cipher
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[31] «IHE ATNA Auditor based on RFC-3881», Διαθέσιμο από:
https://www.projects.openhealthtools.org/sf/projects/iheprofiles/userguides/1.1.0/ATNA/Auditor.html
[Διαβάστηκε τελευταία φορά τον Απρίλιο του 2017]
[32] «Security Audit and Access Accountability Message XML Data Definitions for Healthcare Applications», Διαθέσιμο από:
https://tools.ietf.org/html/rfc3881
[Διαβάστηκε τελευταία φορά το Μάιο του 2017]
[33] Εικόνα σχετική με τη φάση της Ανάλυσης Απαιτήσεων, Διαθέσιμη από:
https://s-media-cache-ak0.pinimg.com/736x/81/73/6e/81736e85d37f5ee3351f38c914dae3c0.jpg
[Διαβάστηκε τελευταία φορά το Μάιο του 2017]
[34] Εικόνα σχετική με τη φάση της Ανάλυσης Απαιτήσεων, Διαθέσιμη από:
https://s-media-cache-ak0.pinimg.com/736x/6f/74/11/6f7411ca6484e1be1b2964c70b4ed3c0.jpg
[Διαβάστηκε τελευταία φορά το Μάιο του 2017]
[35] Εικόνα για σχηματική αναπαράσταση του Cypher Block Chaining, Διαθέσιμη από:
https://crypto.stackexchange.com/questions/225/should-i-use-ecb-or-cbc-encryption-mode-for-my-block-cipher
[Διαβάστηκε τελευταία φορά το Μάιο του 2017]