ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΜΜΕ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Σχεδιασμός και ανάπτυξη διαδικτυακής εφαρμογής Διενέργειας και Διαχείρισης κρατήσεων σε ξενοδοχειακές μονάδες»

ΠΑΣΧΑΛΗ ΠΑΝΑΓΙΩΤΑ

ΕΠΟΠΤΕΥΩΝ ΚΑΘΗΓΗΤΗΣ: ΦΕΡΕΝΤΙΝΟΣ ΒΗΣΣΑΡΙΩΝ

ΠΥΡΓΟΣ, 2018

ΠΙΣΤΟΠΟΙΗΣΗ

Πιστοποιείται ότι η πτυχιακή εργασία με θέμα:

«Σχεδιασμός και ανάπτυξη διαδικτυακής εφαρμογής διενέργειας και Διαχείρισης κρατήσεων σε ξενοδοχειακές μονάδες»

του φοιτητή του Τμήματος ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΜΜΕ

ΠΑΣΧΑΛΗ ΠΑΝΑΓΙΩΤΑ

Α.Μ.: 1625

παρουσιάστηκε δημόσια και εξετάσθηκε στο Τμήμα ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΜΜΕ στις

___12____ / ___09____ / ___2018____

Ο ΕΠΙΒΛΕΠΩΝ

ΒΗΣΣΑΡΙΩΝ ΦΕΡΕΝΤΙΝΟΣ

Ο ΠΡΟΕΔΡΟΣ ΤΟΥ ΤΜΗΜΑΤΟΣ

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΜΗ ΛΟΓΟΚΛΟΠΗΣ

Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Ακόμα δηλώνω ότι αυτή η γραπτή εργασία προετοιμάστηκε από εμένα προσωπικά και αποκλειστικά και ειδικά για την συγκεκριμένη πτυχιακή εργασία και ότι θα αναλάβω πλήρως τις συνέπειες εάν η εργασία αυτή αποδειχθεί ότι δεν μου ανήκει.

ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΠΟΥΔΑΣΤΗ 1	**ΑΜ**	**ΥΠΟΓΡΑΦΗ**
ΠΑΣΧΑΛΗ ΠΑΝΑΓΙΩΤΑ	1625	

ΕΥΧΑΡΙΣΤΙΕΣ

Θα θελα να ευχαριστήσω θερμά τον καθηγητή μου κ. Φερεντίνο κυρίως για την εμπιστοσύνη που μου έδειξε, και την υπομονή που έκανε κατά τη διάρκεια υλοποίησης της πτυχιακής εργασίας. Όπως επίσης και για την πολύτιμη βοήθεια και καθοδήγηση της, για την επίλυση διάφορων θεμάτων.

Θα ήθελα επίσης να απευθύνω τις ευχαριστίες μου στους γονείς μου, οι οποίοι στήριξαν τις σπουδές μου με διάφορους τρόπους, φροντίζοντας για την καλύτερη δυνατή μόρφωση μου.

ΠΡΟΛΟΓΟΣ

Η Ελλάδα βασίζει την οικονομία της κατά ένα μεγάλο βαθμό στον τουρισμό. Το γεγονός αυτό έχει απασχολήσει αρκετούς ξενοδόχους για το πώς θα γίνει ευκολότερη η διαχείρηση των ξενοδοχείων τους, καθώς και το πώς θα ευχαριστήσουν τις ανάγκες των πελατών τους.

Η παρούσα εργασία έχει ως στόχο την δημιουργία ενός εύκολου προς τον χρήστη συστήματος για τις διακοπές του. Επίσης έχει ως στόχο την διευκόλυνση της διαχείρισης μιας ξενοδοχειακής μονάδας εξολοκλήρου από τον ξενοδόχο.

Σκοπός της εργασίας είναι η κατανόηση των εργαλείων που υπάρχουν για την ανάπτυξη διαδικτυακών εφαρμογών καθώς και η δημιουργία μιας εφαρμογής εξολοκλήρου βασισμένο σε κώδικα και όχι σε κάποια σουίτα δημιουργίας διαδικτυακών εφαρμογών (CMS).

ΠΕΡΙΛΗΨΗ

Στην εργασία αυτή, παρουσιάζεται μια εφαρμογή για την διαχείρηση ξενοδοχειακών μονάδων. Η εφαρμογή αυτή υλοποιήθηκε με χρήση PHP, HTML, CSS, JavaScript και jQuery όσον αφορά των κώδικα υλοποίησής του και με MySql όσον αφορά την βάση δεδομένων.

Στην εφαρμογή υπάρχουν τρία είδη χρηστών. Πρώτο είδος είναι ο διαχειριστής ο οποίος έχει την δυνατότητα να απαγορεύει την είσοδο κάποιου ξενοδόχου ή να την επαναφέρει, να διαγράφει προσωρινά ή να επαναφέρει ένα ξενοδοχείο στην αναζήτηση καθώς και να δέχεται κάποιο αίτημα ενός χρήστη να γίνει ξενοδόχος.

Δεύτερο είδος είναι ο ξενοδόχος, ο οποίος μπορεί να αλλάζει τις πληροφορίες του ξενοδοχείου του όπως είναι οι φωτογραφίες του ή κάποια περίληψη που έχει προσθέσει γι' αυτό, να δημιουργεί περισσότερα από ένα ξενοδοχεία και να τα επεξεργάζεται καθώς και να διαγράφει και να εισάγει δωμάτια με τις παροχές τους.

Τέλος υπάρχει και ο απλός χρήστης, ο οποίος μπορεί να ψάξει ένα ξενοδοχείο ή μια περιοχή και να βρει το κατάλληλο γι' αυτόν ξενοδοχείο, φιλτράροντας την επιλογή του όπως αυτός θέλει. Για να κάνει μια κράτηση θα πρέπει να έχει πρώτα κάνει είσοδο στην εφαρμογή με τον κωδικό που έχει ή να δημιουργήσει καινούργιο λογαριασμό.

ABSTRACT

This thesis is describing a web application about managing hotels. This application is made with PHP, HTML, CSS, JavaScript και jQuery for programming it and MySql for holding the data in a database.

In this application there are three kinds of users. To begin with there is an admin user, who can ban the entrance of a hotel manager to the application or enable his access. Also he can hide temporally a hotel from the search engine or restore it. Moreover he can accept a request of a user to be a hotel manager.

The second kind of users is the hotel manager, who can change his hotel's information like its photos or a description that he had already written for it. Also he can create and manage more than one hotel. In addition he can add or remove rooms from his hotels.

Finally there is the third kind of users, which is the common user, who can search for a hotel or a place and filter the hotels with some filters that are made by the application. In order to book a room, he has to enter the application with his account.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Διαδικτυακή εφαρμογή, php,jquery,javascript,mysql,ξενοδοχεία

Περιεχόμενα

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

Ευρετήριο κωδίκων

ΕΙΣΑΓΩΓΗ

Η πτυχιακή αυτή είναι μια προσπάθεια δημιουργίας μιας διαδικτυακής εφαρμογής για την διαχείρηση ξενοδοχιακών μονάδων. Για την ανάπτυξη της εφαρμογής αυτής χρησημοποιήθηκαν η markup γλώσσα HTML 5 καθώς και η styling «γλώσσα» CSS3. Για το προγραμματιστικό κομμάτι χρησιμοποιήθηκε η γλώσσα προγραμματισμού PHP, όσον αφορά την πληροφορίες από το server, και όπου δεν χρειαζόταν να λάβει πληροφορίες χρησημοποιήθηκε η JavaScript με την βιβλιοθήκη της jQuery ώστε να μην χρειάζεται ο server να χρησημοποιήσει παραπάνω επεξεργαστηκή ισχύ από ότι επιβάλλετε.

Μερικές φορές, χρησιμοποιήθηκε η τεχνολογία AJAX για την επικοινωνία με τον server, με σκοπό να μην ανανεώνεται συνέχεια η εκάστοτε ιστοσελίδα που βρίσκεται ο χρήστης και με αυτό τον τρόπο τον κουράσει.

Κατά την διάρκειά μου στο Τ.Ε.Ι. ήρθα σε επαφή με μερικές από τις γλώσσες αυτές. Στην εργασία αυτή χρησιμοποίησα τις θεωρητικές γνώσεις που είχα λάβει προσπαθώντας να τις αξιοποιήσω στην εφαρμογή αυτή. Η αξιοποίησή τους όμως ήταν πιο δύσκολη. Ξεκίνησα να βλέπω διάφορα βίντεο με τον τρόπο λειτουργίας των εργαλείων που είχα καθώς και με τον τρόπο που μπορώ να τα χρησιμοποιήσω προς όφελος την εφαρμογής. Ξεκίνησα με μικρά κομμάτια κώδικα κάνοντας κάποιες ασκήσεις που έβρισκα στο διαδίκτυο και συνέχιζα στα μεγαλύτερα.

Αφού είχα καταφέρει να πάρω τις κατάλληλες γνώσεις στο κομμάτι του προγραμματισμού, και έχοντας διαβάσει πολύ για αυτές στο διαδίκτυο, ξεκίνησα την εφαρμογή μου, σχεδιάζοντας πως θέλω να εμφανίζεται η εφαρμογή μου στον χρήστη και μετά στην υλοποίηση με κώδικα. Σκέφτηκα τις κατηγορίες των χρηστών και τι μπορούν να κάνουν αυτές για στην εφαρμογή και τέλος πως αυτά συνδυάζονται με την βάση δεδομένων που είχα στο μυαλό μου.

Ξεκίνησα κατασκευάζοντας το σκελετό της βάσης δεδομένων καθώς και δημιουργώντας τα κατάλληλα πεδία που εκείνη την στιγμή θεωρούσα απαραίτητα. Στην συνέχεια έβαλα κάποια ψευδό-δεδομένα στην βάση για να κατασκευάσω την αρχική σελίδα που θα μπαίνει ο χρήστης. Στην αρχική σελίδα μπορεί κάποιος να γράψει την περιοχή που θέλει ή ένα ξενοδοχείο και να του εμφανίσει αντίστοιχα τα ξενοδοχεία της περιοχής - τα οποία μπορεί και να φιλτράρει ανάλογα με το τι ψάχνει – ή πληροφορίες για το ξενοδοχείο που επέλεξε.

Στην συνέχεια έφτιαξα τις σελίδες εγγραφής και σύνδεσης για την εφαρμογή, και χρησιμοποίησα την κρυπτογραφική μέθοδο salt and pepper με κρυπτογράφηση SHA-256. Ο λόγος είναι ότι σε μια εφαρμογή δεν πρέπει να υπάρχει σε κείμενο ο κωδικός κάποιου χρήστη και με το salt and pepper προσθέτεις μία ακόμα ασφάλεια στην εφαρμογή σου.

Έπειτα ξεκίνησα την σύνταξη της σελίδας του προφίλ για κάθε χρήστη και την σελίδα αλλαγής κωδικού. Μόλις τελείωσα με αυτό ξεκίνησα την σελίδα του διαχειριστή και τέλος τις σελίδες που αφορούν τον ξενοδόχο.

1 Παρόμοιες εφαρμογές

Στην εποχή της ραγδαίας ανάπτυξης της τεχνολογίας, υπάρχουν αρκετές εφαρμογές οι οποίες υλοποιούν τα ίδια πράγματα με την εφαρμογή αυτή. Τα παλαιότερα χρόνια, η διαδικασία της κράτησης ενός δωματίου ήταν πολύ διαφορετική. Για να κάνεις κράτηση ενός δωματίου, έπρεπε να επικοινωνήσεις μέσω τηλεφώνου στο ξενοδοχείο και να γίνει έλεγχος για την ύπαρξη ή όχι ελεύθερου δωματίου. Στην περίπτωση δε που δεν υπάρχει διαθέσιμο δωμάτιο, τότε έπρεπε να καλέσεις σε άλλο ξενοδοχείο, και αυτό έπρεπε να το κάνεις για όλα τα ξενοδοχεία που υπήρχαν σε ένα μέρος. Στην συνέχεια έπρεπε να κάνεις έρευνα για τις τιμές σε αυτά που είχαν ελεύθερα δωμάτια. Όταν λοιπόν έβρισκες αυτό που ήθελες τότε έπρεπε να πάρεις τηλέφωνο ξανά για να το κλείσεις. Πάλι όμως μπορεί να μην το έβρισκες μιας και μπορεί κάποιος άλλος να το είχε κλείσει όσο έψαχνες να βρεις την κατάλληλη λύση.

Με την ανάπτυξη της τεχνολογίας και των διαδικτυακών εφαρμογών, η λύση βρίσκεται στο χέρι του καθενός από εμάς. Ο λόγος είναι οι διαδικτυακές εφαρμογές οι οποίες διαχειρίζονται τις κρατήσεις ξενοδοχείων ακόμα και αν ο χρήστης χρησιμοποιήσει το κινητό τηλέφωνό του. Στην συνέχεια θα αναλυθούν οι τρείς πιο γνωστές εφαρμογές του διαδικτυακού χώρου οι οποίες έχουν κάνει την διαδικασία αυτή πιο εύκολη και άμεση από ποτέ.

1.1 Booking.com

Booking.com

Ξεκινώντας μια από τις μεγαλύτερες είναι το booking.com. Είναι μια εφαρμογή η οποία παρέχει μια ολοκληρωμένη σουίτα σε έναν ξενοδόχο ώστε να διαχειριστεί το κατάλυμά του. Η εφαρμογή αυτή ανήκει στην εταιρία Booking.com, η οποία ιδρύθηκε το 1996 από μια ολλανδική εταιρία στο Άμστερνταμ[1]. Αυτή την στιγμή η εταιρία έχει επεκταθεί σε ολόκληρο τον κόσμο και απασχολεί πλέον πάνω από 17.000 εργαζόμενους σε 198 γραφεία και σε 70 χώρες σε όλο τον κόσμο. Ο ιστοχώρος και οι εφαρμογές για κινητές συσκευές της Booking.com διατίθενται σε πάνω από 40 γλώσσες, προσφέρουν 28.908.875 καταχωρίσεις συνολικά, και καλύπτουν πάνω από 138.765 προορισμούς σε 229 χώρες και επικράτειες σε όλο τον κόσμο.

1.2 Agoda.com

agoda

Μια ακόμα εφαρμογή είναι αυτή της Agoda. Η Agoda είναι μία από τις ταχύτερα αναπτυσσόμενες πλατφόρμες online ξενοδοχειακών κρατήσεων. Η νεοφυής αυτή

επιχείρηση, που ιδρύθηκε το 2005[2], επεκτάθηκε σύντομα στην Ασία και, το 2007, αποκτήθηκε από την Booking Holdings Inc., τη μεγαλύτερη υπηρεσία διαδικτυακής πώλησης δωματίων. Η Agoda έχει την έδρα της στη Σιγκαπούρη, με 53 γραφεία σε μεγάλες πόλεις σε 30 χώρες του κόσμου και πάνω από 3.700 υπαλλήλους συνολικά. Παρέχει ένα δίκτυο από περισσότερα από 2 εκατομμύρια καταλύματα, μεταξύ των οποίων διαμερίσματα, βίλες, σπίτια και ξενοδοχεία, τα οποία στηρίζουν πάνω από 15 εκατομμύρια κριτικές από αληθινούς ταξιδιώτες. Η Agoda υπήρξε πρωτοπόρος στα μη ξενοδοχειακά καταλύματα, προσφέροντας βίλες και σπίτια σε όλες τις περιοχές από το ξεκίνημά της.

1.3 Bookgreece.com

Τέλος αξίζει να αναφερθεί και η εφαρμογή του BookGreece.com. Ο Bookgreece.com είναι ένα από τα μεγαλύτερα και πιο εξειδικευμένα Ταξιδιωτικά Πρακτορεία που προσφέρει ταξιδιωτικές υπηρεσίες μέσω του διαδικτύου στην Ελλάδα [3]. Έχει

- Ένα μεγάλο εύρος με πάνω από 5.000 διαφορετικά ξενοδοχεία σε όλη την Ελλάδα, όπως στην Αθήνα, στην Βόρεια Ελλάδα και σε πολλά Ελληνικά Νησιά όπως την Κρήτη.
- 100% Ασφαλής Διαδικασία Κράτησης.
- Πάνω από 75 χρόνια εμπειρίας στην Τουριστική Βιομηχανία.

Το Bookgreece είναι μέρος της Aeolos Cyprus Travel Limited. Η Aeolos Cyprus Travel Limited ανήκει στην Φραγκούδη & Στεφάνου Λτδ . Η Φραγκούδη & Στεφάνου είναι ένας διαφοροποιημένος, ευέλικτος και ανταγωνιστικος Όμιλος Εταιρειών με πάνω από 1500 εργαζόμενους πλήρους απασχόλησης σε Κύπρο και εξωτερικό και συμμετέχουν σε ένα ευρύ φάσμα δραστηριοτήτων στους τομείς της ασφάλειας, Ταξιδιού & Τουρισμού, Ξενοδοχείων, Τηλεπικοινωνιών, Πληροφορικής & τεχνολογίας, Κατασκευών, και Εμπορίου. Το TUI Group είναι παγκοσμίως η κορυφαία εταιρεία ταξιδίων αναψυχής και λειτουργεί σε περισσότερες από 180 χώρες κσι με περισσότερους από 30 εκατομμύρια πελάτες σε 27 βασικές αγορές.

2 Σχεδιασμός εφαρμογής

Για τον σχεδιασμό μιας εφαρμογής, υπάρχουν πολλές τεχνικές μέσω των οποίων μπορεί να γίνει πιο σωστά η αναπαράσταση της με γραφικό, και όχι μόνο, τρόπο. Σε μία έρευνα που έγινε σχετικά με τα αποτελέσματα της χρήσης αυτών των τεχνολογιών στην διαδικασία της ανάπτυξης της εφαρμογής, έδειξε ότι από τις εταιρίες αυτές, το 85% ανέφερε ότι αυξήθηκε η παραγωγικότητα, το 10% είπε ότι έμεινε στα ίδια επίπεδα, ενώ μόνο το 5% είπε ότι μειώθηκε η παραγωγικότητα [4]. Για τον λόγο αυτό στην εφαρμογή χρησιμοποιήθηκαν τα διαγράμματα ονοτήτων, τα διαγράμματα περιπτώσεων χρήσης, τα διαγράμματα μετάβασης καταστάσεων και τα διαράμματα ροής.

2.1 Βάση Δεδομένων

Μία βάση δεδομένων είναι μια ολοκληρωμένη συλλογή από συσχετιζόμενα δεδομένα. [7]. Σε μία βάση δεδομένων μπορεί να γίνουν αναζητήσεις (queries), ενημερώσεις (updates) και συναλλαγές (transactions). Μια ειδική κατηγορία συναλλαγής είναι οι συναλλαγές αναζητήσεων (read-only ή query transactions) οι οποίες κάνουν μόνο αναζήτηση χωρίς κάποια ενημέρωση.

Στην εφαρμογή χρησιμοποιείται το σχεσιακό μοντέλο δεδομένων. Το μεγάλο πλεονέκτημα του μοντέλου αυτού είναι ότι μπορεί να περιγράφει με μαθηματικό τρόπο, με την βοήθεια της θεωρίας συνόλων ή της κατηγορηματικής λογικής. Οι βασικοί στόχοι του μοντέλου αυτού είναι:

- Η υποστήριξη της ανεξαρτησίας δεδομένων, έτσι ώστε αλλαγές στην οργάνωση της βάσης δεδομένων να μην απαιτούν αλλαγές στην εφαρμογή
- Η αποφυγή του πλεονασμού, δηλαδή της αποθήκευσης των ίδιων δεδομένων πολλές φορές σε διαφορετικές περιοχές της βάσης δεδομένων
- Η διατήρηση της ακεραιότητας και της συνέπειας των δεδομένων [19]

Η βάση δεδομένων της εφαρμογής απότελείται από τους εξής πίνακες:

- users
- hotels
- rooms
- towns
- paroxes_xenodoxiou
- paroxes_dwmatiou
- closed_rooms
- kratiseis
- user_ratings
- manager_requests

Ο πίνακας users περιέχει πληροφορίες που αφορούν τους χρήστες, είτε αυτοί είναι απλοί χρήστες, είτε ξενοδόχοι είτε διαχειριστές. Αποτελείτε από τα εξής πεδία:

1. *id*: είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id του κάθε χρήστη.
2. *fname*: είναι το όνομα του χρήστη
3. *lname*: είναι το επώνυμο του χρήστη
4. *email*: είναι το email του χρήστη
5. *username*: αντιστοιχεί στο όνομα χρήστη
6. *pwd*: είναι το πεδίο που αποθηκεύει το SHA-256 hash του κωδικού που δίνει ο χρήστης στην εφαρμογή, αφού έχει προστεθεί ένα τυχαίο `salt`.
7. *salt*: είναι ένα τυχαίο string το οποίο προστίθεται στον κωδικό που δίνεται κατά την εγγραφή νέου χρήστη και έπειτα γίνεται το SHA-256 hash του νέου string.
8. *type*: είναι ο τύπος του χρήστη, και πιο αναλυτικά:
    a. 1: αντιστοιχεί στον απλό χρήστη
    b. 2: αντιστοιχεί στον ξενοδόχο
    c. 3: αντιστοιχεί στον διαχειριστή
9. *status*: είναι ένα πεδίο το οποίο παίρνει μηδέν και ένα ανάλογα αν ο χρήστης είναι απενεργοποιημένος από τον διαχειριστή ή όχι αντίστοιχα.

Ο πίνακας hotels περιέχει πληροφορίες για τα ξενοδοχεία και αποτελείτε από τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id του κάθε ξενοδοχείου.
2. *name:* αποτελεί το όνομα του ξενοδοχείου
3. *description:* περιέχει μια περιγραφή για το ξενοδοχείο
4. *creator_id:* είναι ένα ξένο κλειδί που συνδέεται με το id του ξενοδόχου που το έχει δημιουργήσει
5. *main_photo:* είναι το path που πρέπει να ακολουθήσει η εφαρμογή για να βρει την βασική φωτογραφία του ξενοδοχείου
6. *other_photos:* είναι τα paths που πρέπει να ακολουθήσει η εφαρμογή για να βρει τις υπόλοιπες φωτογραφείς που έχει το ξενοδοχείο. Τα paths χωρίζονται με κόμμα μεταξύ τους
7. *status:* είναι ένα πεδίο το οποίο παίρνει μηδέν και ένα ανάλογα αν το ξενοδοχείο είναι απενεργοποιημένο από τον διαχειριστή ή όχι αντίστοιχα.
8. *paroxes:*περιέχει τα id των παροχών που έχει το ξενοδοχείο τα οποία βρίσκονται στον πίνακα paroxes_xenodoxiou χωρισμένες με κόμμα
9. *town:* αντιστοιχεί στο μοναδικό id της πόλης στην οποία το ξενοδοχείο είναι δηλωμένο από τον ξενοδόχο κατά την δημιουργία του και είναι ξένο κλειδί που συνδέεται με τα id του πίνακα towns
10. *stars:* είναι ένα νούμερο από το μηδέν μέχρι το πέντε που δηλώνει τα αστέρια του ξενοδοχείου
11. *lat:* Αποτελεί το γεωγραφικό πλάτος στο οποίο βρίσκεται το ξενοδοχείο
12. *lng:* Αποτελεί το γεωγραφικό μήκος στο οποίο βρίσκεται το ξενοδοχείο

Ο πίνακας rooms αποτελεί τον πίνακα στον οποίο αποθηκεύονται τα δωμάτια και γίνεται αντιστοιχία με το ξενοδοχείο στο οποίο ανήκουν. Αποτελείται από τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id του κάθε δωματίου.
2. *type*: αντιστοιχεί στον τύπο του δωματίου και πιο συγκεκριμένα παίρνει τις εξής τιμές:
    a. 1: μονόκλινο
    b. 2: δίκλινο
    c. 3: οικογενειακό
3. *price*: Είναι η τιμή του δωματίου
4. *discount*: Είναι η πιθανή έκπτωση που μπορεί ένα δωμάτιο να έχει
5. *hotel_id*: Αντιστοιχεί στο μοναδικό id του ξενοδοχείου στο οποίο το δωμάτιο ανήκει και αποτελεί ξένο κλειδί που συνδέεται με το id του πίνακα hotels
6. *paroxes_dwmatiou*: περιέχει τα id των παροχών που έχει το ξενοδοχείο τα οποία βρίσκονται στον πίνακα paroxes_dwmatiou χωρισμένες με κόμμα

Ο πίνακας towns αποτελεί τον πίνακα που περιέχει πληροφορίες για τις πόλεις τις οποίες έχει η εφαρμογή μας και περιέχει τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id της κάθε πόλης
2. *name:* Αποτελεί το όνομα της πόλης
3. *nomos:* Αποτελεί τον νομό της πόλης
4. *poli_perifereia:* Αναφέρεται σε ένα νούμερο το οποίο δηλώνει αν είναι πόλη και παίρνει την τιμή ένα και αν είναι περιφέρεια παίρνει την τιμή 2. Το πεδίο αυτό προστέθηκε διότι έπρεπε με κάποιον τρόπο να χωριστούν οι δήμοι τις Αττικής μιας και σε διαφορετικούς δήμους μπορεί να έχουμε ίδιες οδούς.

Στην συνέχεια ο πίνακας paroxes_xenodoxiou αποτελεί τον πίνακα που αποθηκεύονται οι παροχές του ξενοδοχείου. Αποτελείται από τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id της κάθε παροχής του ξενοδοχείου
2. *name:* είναι το όνομα τα παροχής που πιθανόν να έχει ένα ξενοδοχείο.

Παρόμοιος είναι και ο πίνακας paroxes_dwmatiou ο οποίος αναφέρεται στις παροχές που μπορεί να έχει το δωμάτιο ενός ξενοδοχείου και αποτελείτε από τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id της κάθε παροχής του δωματίου
2. *name:* είναι το όνομα τα παροχής που πιθανόν να έχει ένα ξενοδοχείο.

Ο πίνακας closed_rooms είναι ο πίνακας στον οποίο αποθηκεύεται η πληροφορία για το ποιο δωμάτιο είναι κλεισμένο και σε πια περίοδο. Αποτελείτε από τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id του κάθε κλεισμένου δωμάτιου
2. *room_id:* είναι το id του δωματίου το οποίο είναι κλεισμένο για μια χρονική περίοδο. Αποτελεί ξένο κλειδί που συνδέεται με τον πίνακα rooms και το πεδίο id
3. *from_date:* είναι η ημερομηνία από την οποία είναι κλεισμένο το δωμάτιο
4. *to_date:* είναι η ημερομηνία μέχρι την οποία το δωμάτιο είναι κλεισμένο

Ο πίνακας kratiseis είναι ένας πίνακας παρόμοιος με τον προηγούμενο περιέχει όμως περισσότερες πληροφορίες και χρησιμοποιείται όχι μόνο για να γίνει έλεγχος αν ένα δωμάτιο είναι κλειστό για μια χρονική περίοδο αλλά για να μπορέσει ο χρήστης να δει την κράτησή του στο ιστορικό κρατήσεων. Αποτελείται από τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id της κάθε κράτησης
2. *from_date:* είναι η ημερομηνία από την οποία είναι κλεισμένο το δωμάτιο
3. *to_date:* είναι η ημερομηνία μέχρι την οποία το δωμάτιο είναι κλεισμένο
4. *hotel_id:* Είναι το id του ξενοδοχείου στο οποίο ανήκει το δωμάτιο που έγινε η κράτηση και αποτελεί ξένο κλειδί που συνδέεται με τον πίνακα hotels και το πεδίο id
5. *user_id:* Είναι το id του χρήστη ο οποίος έκανε την κράτηση και αποτελεί ξένο κλειδί που συνδέεται με τον πίνακα users και το πεδίο id
6. *room_id:* Είναι το id του δωματίου στο οποίο έγινε η κράτηση και αποτελεί ξένο κλειδί που συνδέεται με τον πίνακα rooms και το πεδίο id

Ο πίνακας user_rating αποτελεί τον πίνακα στον οποίο αποθηκεύονται οι πληροφορίες που αφορούν την βαθμολογία που έχουν δώσει οι χρήστες στο ξενοδοχείο. Ο πίνακας αυτός έχει τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id της κάθε βαθμολογίας

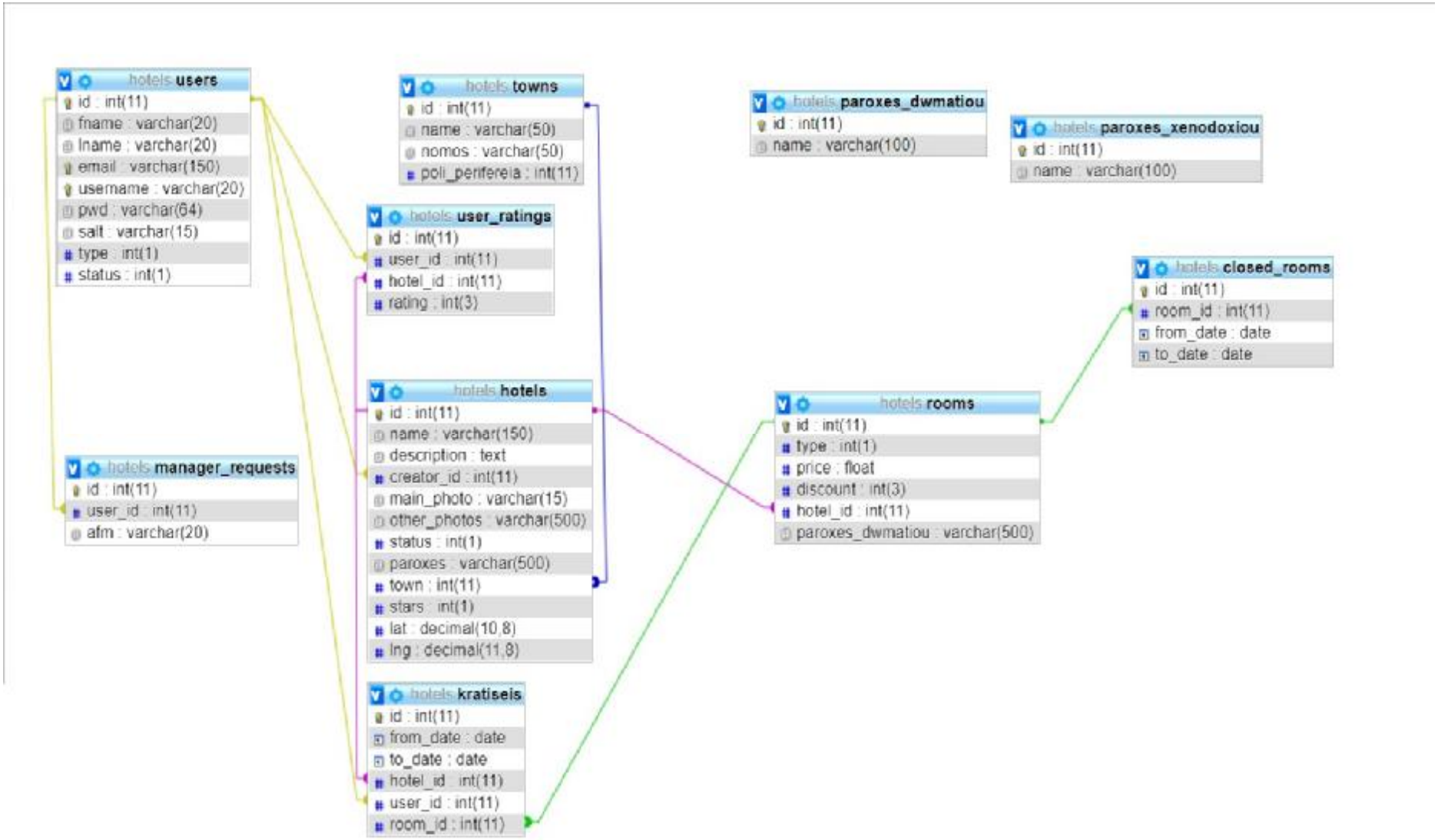
2. *user_id:* Είναι το id του χρήστη ο οποίος έκανε την βαθμολογία και αποτελεί ξένο κλειδί που συνδέεται με τον πίνακα users και το πεδίο id
3. *hotel_id:* Είναι το id του ξενοδοχείου στο οποίο έγινε η κριτική και αποτελεί ξένο κλειδί που συνδέεται με τον πίνακα hotels και το πεδίο id
4. *rating:* Αποτελεί την επί τοις εκατό βαθμολογία που έχει δώσει ο χρήστης

Τέλος ο πίνακας manager_requests περιέχει τα άτομα-χρήστες που έχουν κάνει αίτηση για να γίνουν ξενοδόχοι στην εφαρμογή. Ο πίνακας αυτός αποτελείται από τα εξής πεδία:

1. *id:* είναι το πρωτεύον κλειδί και είναι ένας αύξων αριθμός που αντιστοιχεί στο μοναδικό id του κάθε αιτήματος
2. *user_id:* Είναι το id του χρήστη ο οποίος έκανε την αίτηση και αποτελεί ξένο κλειδί που συνδέεται με τον πίνακα users και το πεδίο id
3. *afm:* Είναι το Α.Φ.Μ. του ξενοδοχείου και ζητείται από τον χρήστη για να γίνει πιστοποίηση των στοιχείων ότι ο χρήστης είναι όντος ιδιοκτήτης του ξενοδοχείου.

Το διάγραμμα της βάσης δεδομένων μαζί με τις συσχετίσεις τους φαίνεται στο επόμενο σχήμα

2.1.1 Διάγραμμα Οντοτήτων συσχετίσεων

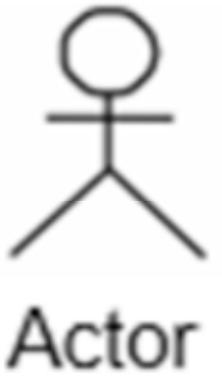


Εικόνα 2.1 Διάγραμμα οντοτήτων συσχετίσεων της βάσης δεδομένων

2.2 Διαγράμματα Περιπτώσεων Χρήσης

2.2.1 Η θεωρία πίσω από τα διαγράμματα περίπτωσής χρήσης

Τα διαγράμματα περιπτώσεων χρήσης αναπαρηστούν περιπτώσεις χρήσης (use cases) του πληροφοριακού συστήματος, ενεργοποιητές (actors) και σχέσεις μεταηύ περιπτώσεων χρήσης και ενεργοποιητών[15][16]. Κάθε περίπτωση χρήσης αντιστοιχίζεται σε έναν και μόνο έναν συγκεκριμένο τρόπο χρήσης του συστήματος. Σχετίζεται άμεσα με την λειτουργικότητα του συστήματος, το οποίο ενεργοποιείται για να ανταποκριθεί σε έναν εξωτερικό ενεργοποιητή (π.χ. έναν χρήστη). Ο ενεργοποιητής συμβολίζεται με ένα ανθρωπάκι όπως αυτό φαίνεται στην εικόνα 2.2.

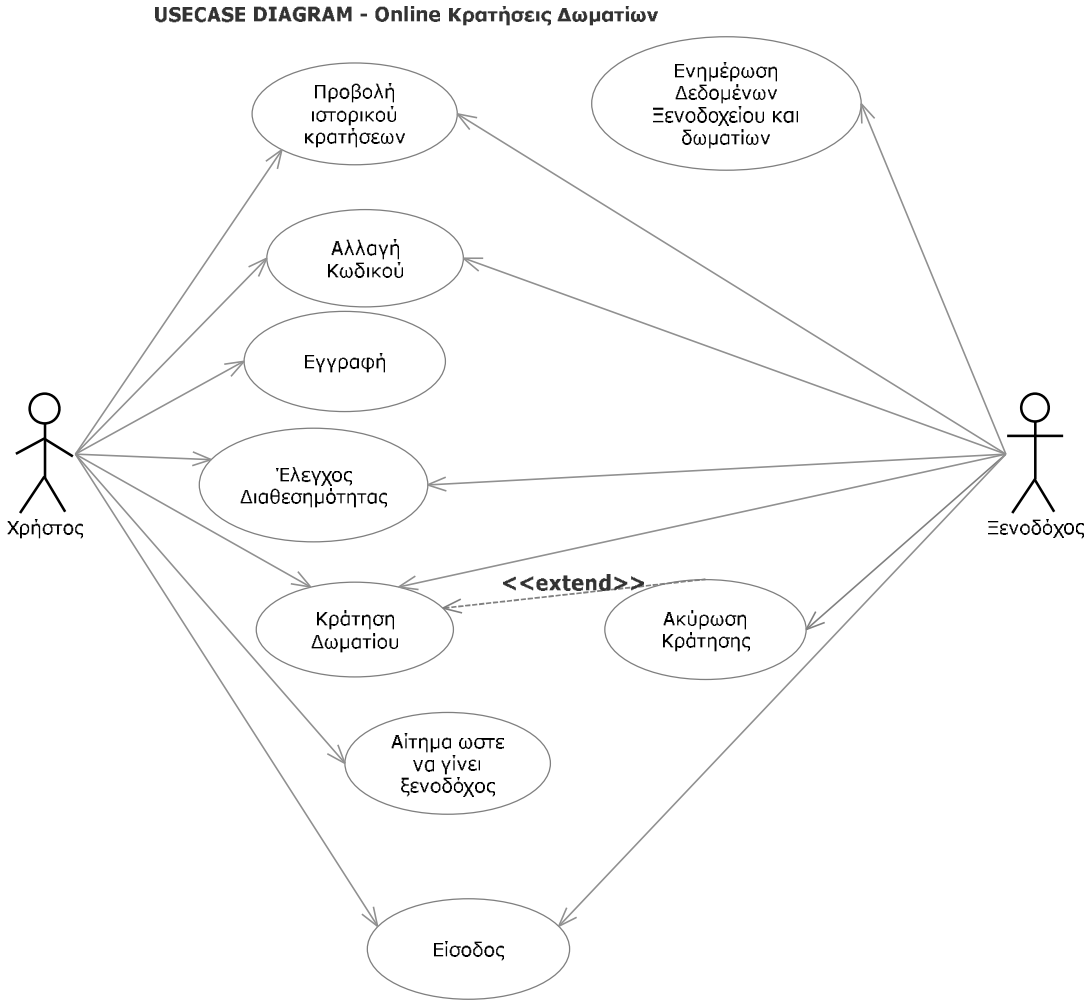


Εικόνα 2.2 Αναπαράσταση ενεργοποιητή στα διαγράμματα περίπτωσης χρήσης

Οι ενεργοποιητές αντιπροσωπεύουν τον ρόλο ενός χρήστη ή μιας οντότητας του συστήματος. Προσδιορίζονται παρατηρώντας τους άμεσους χρήστες του συστήματος. Η συμμετοχή του ενεργοποιητή συμβολίζεται με μια γραμμή μεταξύ του ενεργοποιητή και της περίπτωσης χρήσης και ονομάζεται «σχέση επικοινωνεί». Στην περίπτωση που στο τελείωμα της γραμμής υπάρχει βελάκι, τότε η σχέση ονομάζεται «σχέση χρησιμοποιεί». Η «σχέση επικοινωνεί» είναι η μοναδική σχέση που μπορεί να υπάρξει μεταξύ ενεργοποιητών και περιπτώσεων χρήσης, ενώ η «σχέση χρησιμοποιεί» ορίζεται μεταξύ δύο περιπτώσεων χρήσης και δηλώνει ότι ένα στιγμιότυπο της πηγής συμπεριλαμβάνει τη συμπεριφορά του στόχου (Εικόνα 2.3).

Εικόνα 2.3 Σχέση Χρησιμοποιεί

2.2.2 Διάγραμμα κράτησης δωματίου

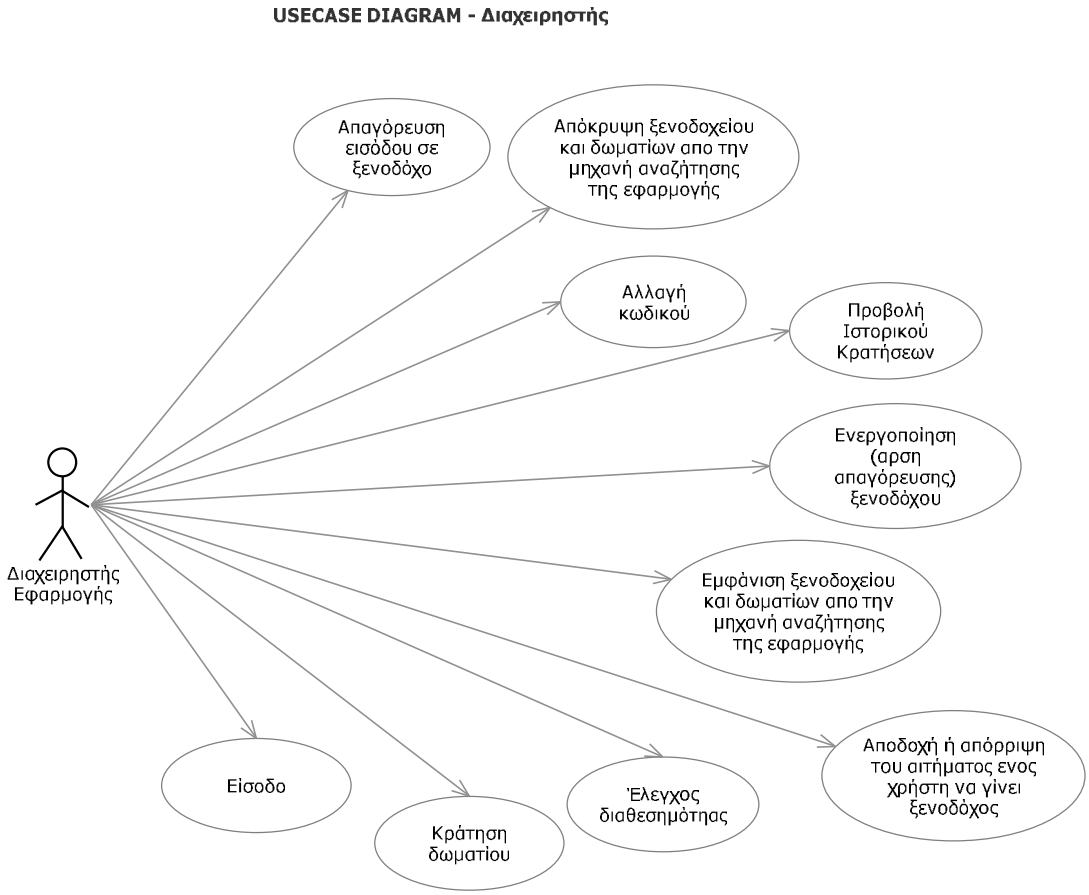


Εικόνα 2.4 Διάγραμμα περίπτωσης χρήσης "Κράτηση Δωματίου"

Στο διάγραμμα αυτό, εμφανίζονται οι διαφορετικές περιπτώσεις που μπορεί ένας χρήστης να χρησιμοποιήσει το σύστημα αυτό [15][16]. Όπως φαίνεται, ένας απλός χρήστης μπορεί να κάνει εγγραφή, να αλλάξει τον κωδικό του, να κάνει έλεγχο διαθεσιμότητας του δωματίου για μια συγκεκριμένη χρονική περίοδο καθώς και να δει το ιστορικό των κρατήσεών του. Επιπλέον, μπορεί να κάνει αίτηση στον διαχειριστή του συστήματος να γίνει ξενοδόχος.

Στο διάγραμμα αυτό φαίνονται επίσης και οι διαφορετικές περιπτώσεις χρήσης του συστήματος από ένα ξενοδόχο. Ο ξενοδόχος μπορεί να κάνει και αυτός έλεγχο διαθεσιμότητας ενός δωματίου για μια συγκεκριμένη χρονική περίοδο και να κάνει κράτηση δωματίου. Ως επί το πλείστον, μπορεί να κάνει ενημέρωση δεδομένων του ξενοδοχείου του και των δωματίων που έχει στην διάθεσή του και να κάνει κάποια ακύρωση ενός δωματίου.

2.2.3 Διάγραμμα διαχειριστή



Εικόνα 2.5 Διάγραμμα περίπτωσης χρήσης Διαχειριστή

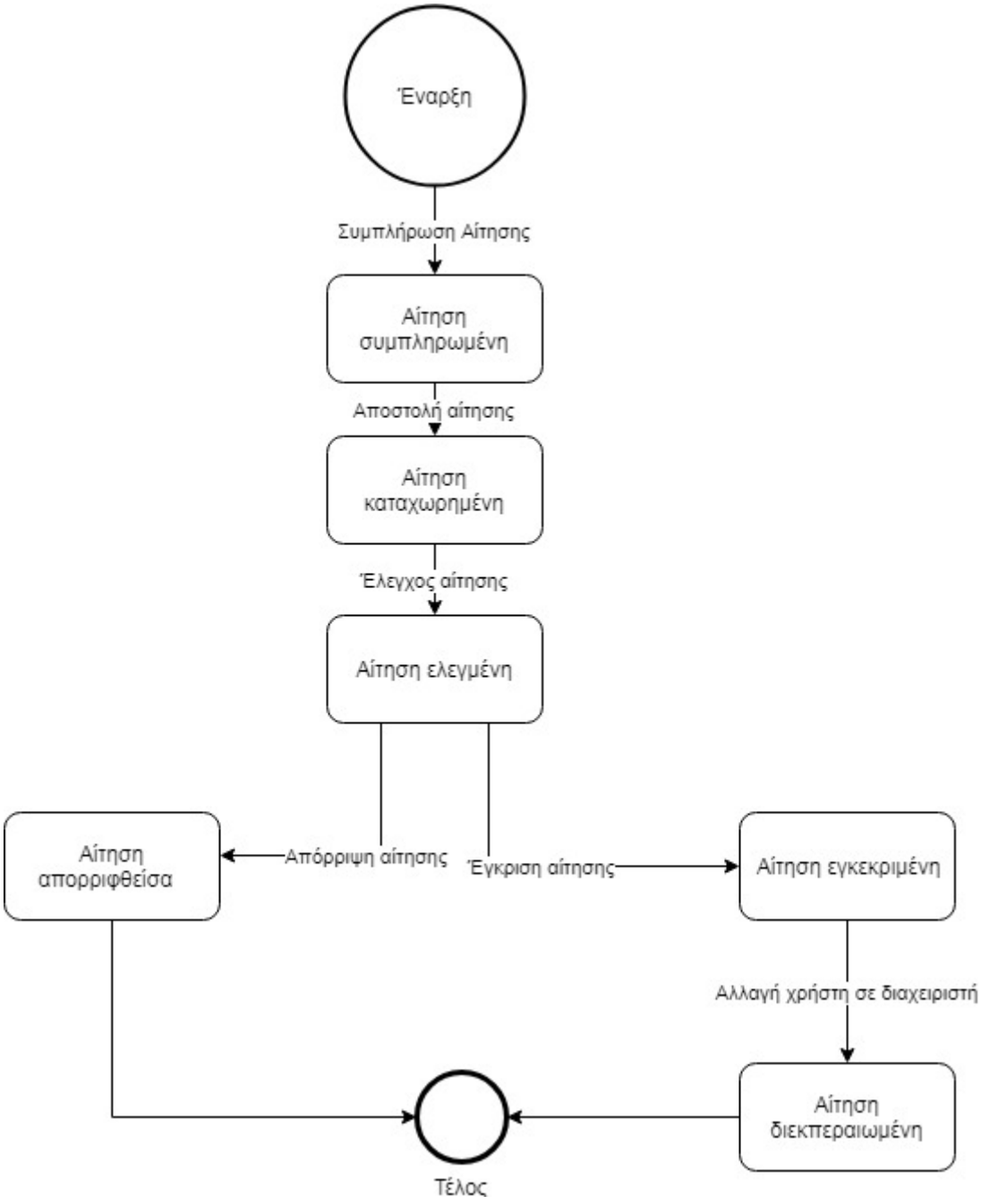
Στο διάγραμμα αυτό, εμφανίζεται η χρήση του συστήματος από τον διαχειριστή του[15][16]. Ένας διαχειριστής μπορεί να εισέλθει στο σύστημα με τον κωδικό του και να απαγορεύσει την είσοδο ενός ξενοδόχου στο σύστημα ή να ενεργοποιήσει την είσοδό του, σε περίπτωση που την έχει ήδη απαγορεύσει. Επιπλέον μπορεί να αποκρύψει/εμφανίσει ένα ξενοδοχείο στο σύστημα αναζήτησης της εφαρμογής. Επίσης όταν ένας χρήστης έχει κάνει αίτημα για να γίνει ξενοδόχος, τότε ο διαχειριστής εισέρχεται στην εφαρμογή και αποδέχεται ή απορρίπτει το αίτημα αυτό. Τέλος ένας διαχειριστής μπορεί να κάνει και μερικές από τις λειτουργίες του απλού χρήστη όπως να αλλάξει τον κωδικό του, να δει το ιστορικό των κρατήσεων του ή να δει τα διαθέσιμα δωμάτια για μια χρονική περίοδο και να κάνει κράτηση κάποιου από αυτά.

2.3 Διαγράμματα μετάβασης καταστάσεων

Τα διαγράμματα μετάβασης καταστάσεων ή αλλιώς διαγράμματα κατάστασης μηχανής, μοντελοποιούν τη συμπεριφορά ενός συστήματος με τη μορφή μεταβάσεων μεταξύ πεπερασμένου αριθμού καταστάσεων[15][16].

Δίνουν έμφαση στη ροή ελέγχου από μία κατάσταση σε μία άλλη. Περιέχουν καταστάσεις και μεταβάσεις ενώ περιγράφουν και τα γεγονότα τα οποία οδήγησαν στην μετάβαση αυτή.

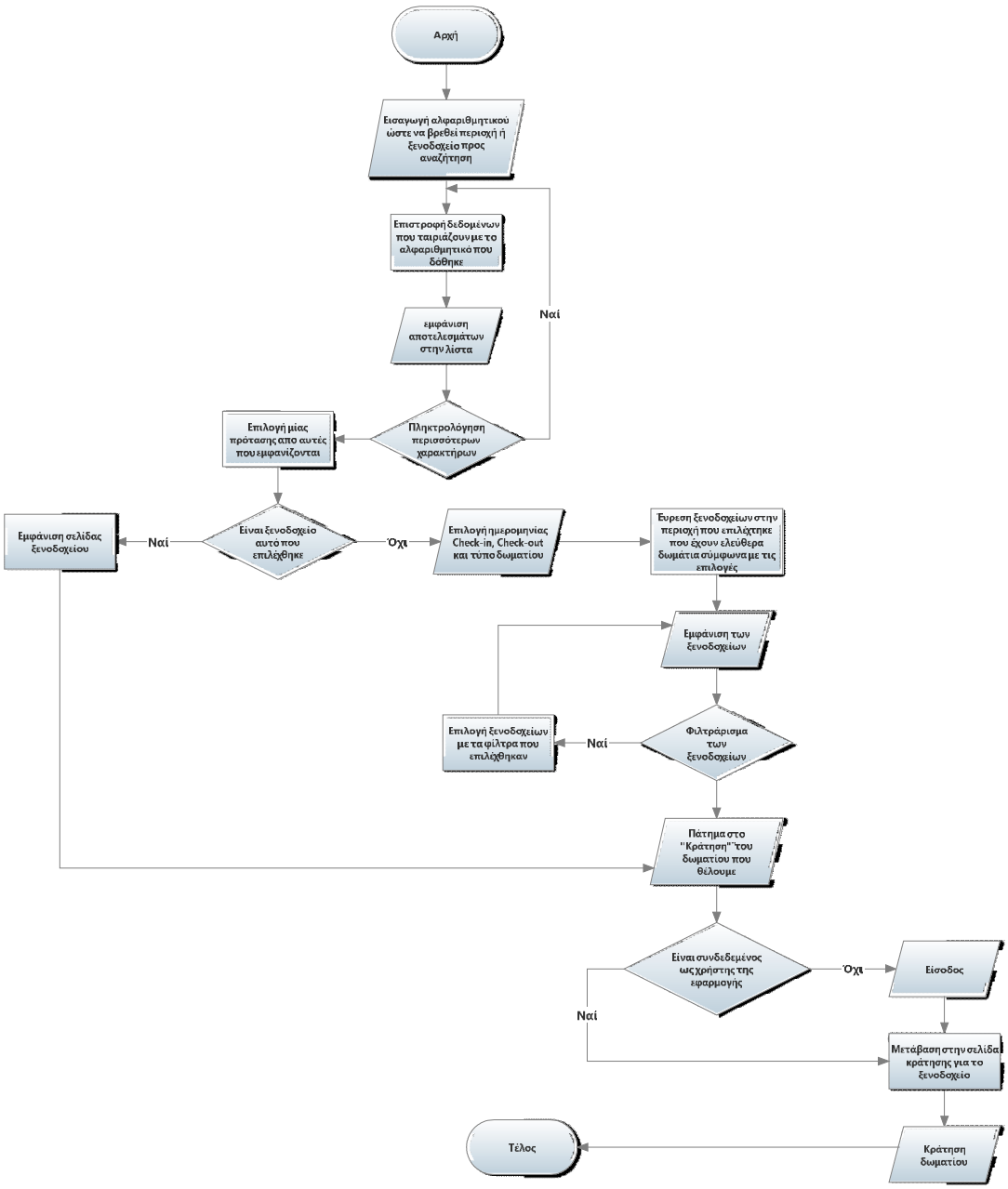
Στην εφαρμογή, αξίζει να σημειωθεί το διάγραμμα μετάβασης καταστάσεων για την κατάσταση του αιτήματος ενός χρήστη ώστε να γίνει ξενοδόχος. Στην εικόνα παρουσιάζεται το διάγραμμα αυτό.



Εικόνα 2.6 Διάγραμμα μετάβασης κατάστασης αιτήματος έγκρισης ως ξενοδόχος

2.4 Διαγράμματα ροής

Στο σημείο αυτό, παρουσιάζονται τα διαγράμματα ροής. Τα διαγράμματα αυτού του τύπου αποτυπώνουν την ακολουθιακή ροή διεργασιών [15][16]. Πρόκειται για απλά διαγράμματα, τα οποία χρησιμοποιούνται στη μοντελοποίηση διαδικασιών, όταν δεν απαιτούνται πολλές πληροφορίες γι' αυτές. Στην ενότητα αυτή θα αναλυθούν τα βασικά σημεία της εφαρμογής.

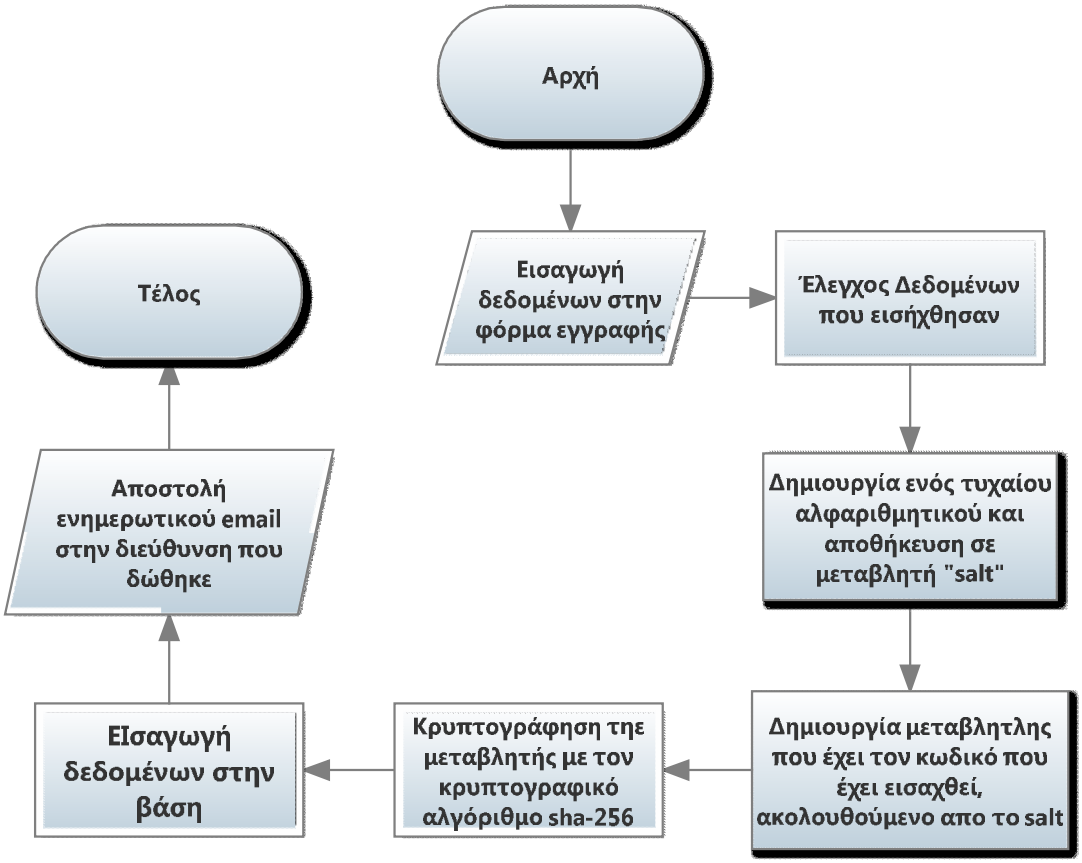


Εικόνα 2.7 Διάγραμμα ροής για κράτηση θέσης

Στην εικόνα 2.7 φαίνεται η διαδικασία η οποία πρέπει να γίνει ώστε να γίνει η κράτηση ενός δωματίου από έναν χρήστη.

Ένα ακόμα διάγραμμα ροής που πρέπει να αναλυθεί είναι ο τρόπος με τον οποίο γίνεται η αποθήκευση δεδομένων στην βάση σχετικά με τον χρήστη. Αυτό που θέλει προσοχή είναι η διαδικασία με την οποία κρυπτογραφείται ο κωδικός του

χρήστη ώστε να μην υπάρξει πρόβλημα σε περίπτωση επίθεσης κάποιου στον server.



Εικόνα 2.8 Διάγραμμα ροής εγγραφής χρήστη

Στην εικόνα 2.8 φαίνεται η διαδικασία με την οποία γίνεται το πέρασμα των δεδομένων στην βάση.

3 Υλοποίηση Εφαρμογής

3.1 Τεχνολογίες που χρησιμοποιήθηκαν

3.1.1 HTML (Hyper Text Markup Language)

Ο όρος HTML είναι το ακρονύμιο των λέξεων (Hyper Text Markup Language) το οποίο μεταφράζεται σε Γλώσσα Σήμανσης Υπαρκειμένου [6][10][15][16]. Δεν είναι γλώσσα προγραμματισμού, αλλά όπως αναφέρει και το όνομά της είναι μια γλώσσα σήμανσης. Η χρήση της ξεκίνησε για τις διαδικτυακές εφαρμογές και αφορούσε την εμφάνιση δεδομένων και πληροφοριών στις ιστοσελίδες.

Τα στοιχεία της HTML είναι οι ετικέτες (tags), οι οποίες περιβάλλονται μέσα στα σύμβολα του μικρότερου και του μεγαλύτερου. Για παράδειγμα η βασικές ετικέτες που πρέπει να υπάρχουν σε κάθε ιστοσελίδα είναι η html, η head και η body, οι οποίες πρέπει να γραφούν με τα σύμβολα του μικρότερου και του μεγαλύτερου. Επομένως οι προηγούμενες ετικέτες εμφανίζονται με πλήρη αντιστοιχία στον κώδικα ως <html>,<head>,<body>. Ο περιηγητής όταν θα κληθεί να διαβάσει τον κώδικα, δεν θα εμφανίσει τις ετικέτες αλλά θα ερμηνεύσει την ετικέτα και θα εμφανίσει το περιεχόμενό της.

Στο σημείο αυτό θα πρέπει να αναφέρουμε ότι όλες οι ετικέτες θα πρέπει να ανοίγουν και να κλείνουν. Για παράδειγμα η ετικέτα html πρέπει να ανοίγει στην αρχή του κειμένου και να κλείνει στο τέλος του. Οι περισσότερες ετικέτες ανοίγουν με όπως η html ετικέτα με τα σύμβολα του μικρότερου και μεγαλύτερου και κλείνουν παρεμβάλλοντας μεταξύ του μικρότερου και του ονόματος της ετικέτας το σύμβολο ‘/’. Για παράδειγμα η ετικέτα html ανοίγει γράφοντας <html> και κλείνει γράφοντας </html>. Μερικές ετικέτες μπορούν να ανοίξουν και να κλείσουν στην δήλωση του ανοίγματος. Μια από αυτές είναι η ετικέτα κενής γραμμής ή αλλιώς br. Η ετικέτα αυτή για να κλείσει αρκεί να παρεμβάλεις του σύμβολο ‘/’ μεταξύ του ονόματος της ετικέτας και του συμβόλου του μεγαλύτερου (
).

Η βασική δομή ενός html αρχείου είναι

```
<html>
    <head>

    </head>

    <body>

    </body>

</html>
```

Κώδικας 3.1 Βασική δομή html αρχείου

Μέσα στο head tags μπορούν να ενσωματωθούν εξωτερικά αρχεία είτε που αφορούν το styling της ιστοσελίδας τα οποία είναι αρχεία css που θα δούμε παρακάτω, είτε αρχεία scripting τα οποία περιέχουν κάποια script με τις συναρτήσεις που θα χρησιμοποιηθούν στην ιστοσελίδα. Στο body tag γράφεται ο κώδικας ο οποίος περιέχει τα στοιχεία αυτά που εμφανίζονται στον περιηγητή.

Τέλος ο κώδικας html μπορεί να αποθηκευτεί σε αρχεία τα οποία έχουν κατάληξη html ή php. Η διαφορά είναι ότι τα αρχεία php θα τρέξουν τον κώδικα php ο οποίος βρίσκεται εντός του αρχείου ενώ τα αρχεία html δεν θα τρέξουν κώδικα php. Στην εφαρμογή αυτή τα αρχεία όλα είναι αποθηκευμένα με κατάληξη php.

3.1.2 Cascading Style Sheets (CSS)

Το ακρωνύμιο css προέρχεται από τις λέξεις Cascading Style Sheets που σημαίνει διαδοχικά φύλλα στυλ [6][10][12][15]. Είναι μια γλώσσα «Φύλλων Στυλ» και ρόλος της είναι η διαμόρφωση του περιεχομένου μιας ιστοσελίδας. Ένα παράδειγμα χρήσης της γλώσσας αυτής είναι το χρώμα των γραμμάτων που υπάρχουν σε μια ετικέτα. Με άλλα λόγια η css εφαρμόζεται στις ετικέτες της html δημιουργόντας πρότυπα για την εμφάνισή τους. Το αποτέλεσμα χρήσης των φύλλων στυλ είναι μια πιο εμφανίσιμη και φιλική προς τον χρήστη ιστοσελίδα.

Η σύνταξη του αρχείου CSS ορίζεται από έναν επιλογέα και μία λίστα από ιδιότητες, οι οποίες λαμβάνουν αντίστοιχες τιμές και περιλαμβάνονται σε αγκύλες ‘{‘. Οι ιδιότητες παίρνουν τις τιμές τους με χρήση της άνω κάτω τελείας ‘:’ και ο διαχωρισμός μεταξύ ιδιοτήτων γίνεται με το ελληνικό ερωτηματικό ‘;’. Ο επιλογέας αναφέρεται σε μια συγκεκριμένη ή περισσότερες ετικέτες. Για παράδειγμα, έστω ότι θέλουμε να κάνουμε τα γράμματα της ετικέτας body να είναι μπλε και το φόντο να είναι άσπρο, χρησιμοποιούμε το όνομα του body και την ιδιότητα color με την τιμή του χρώματος που θέλουμε blue και την ιδιότητα background-color με την τιμή white. Το αποτέλεσμα είναι αυτό

```
body{
    color: blue;
    background-color: white;
}
```

Κώδικας 3.2 Βασική δομή αρχείου CSS

Υπάρχει η περίπτωση να χρειαστεί ο κώδικας να επικεντρωθεί σε μια συγκεκριμένη ετικέτα είτε σε μία ομάδα από ετικέτες. Ο τρόπος να λυθεί το πρόβλημα αυτό είναι όταν χρειαστεί να αναφερθεί σε μια συγκεκριμένη ετικέτα να χρησιμοποιηθεί το χαρακτηριστικό της ετικέτας id. Στην περίπτωση που ο κώδικας αναφέρεται σε μία ομάδα ετικετών τότε χρησιμοποιείται το χαρακτηριστικό των ετικετών class. Στην περίπτωση που αναφέρεται στο χαρακτηριστικό id τότε στο αρχείο css αναφέρεσαι σε αυτό χρησιμοποιώντας την δίεση ‘#’ πριν το όνομα του id της ετικέτας, και αυτό αποτελεί τον επιλογέα. Για παράδειγμα έστω ότι υπάρχει μια

ετικέτα p (παράγραφος) με το id της να είναι το test1 και χρειάζεται να γίνει αλλαγή στο χρώμα των γραμμάτων σε κόκκινο, τότε ο κώδικας είναι ο εξής:

```
#test1{
    color: red;
}
```

Κώδικας 3.3 CSS για επιλογέα με χρήση του id της ετικέτας

Στην περίπτωση που αναφέρεται στο χαρακτηριστικό class τότε το αρχείο css αναφέρεσαι σε αυτό χρησιμοποιώντας την τελεία '.' πριν την τιμή του χαρακτηριστικού class της ετικέτας και αυτό αποτελεί τον επιλογέα. Για παράδειγμα έστω ότι υπάρχει μια ετικέτα p με class να είναι το test1 και χρειάζεται να γίνει αλλαγή στο χρώμα του φόντου σε γκρι, τότε ο κώδικας είναι ο εξής:

```
.test1{
    background-color: red;
}
```

Κώδικας 3.4 CSS για επιλογέα με χρήση του class της ετικέτας

Ένα πιο σύνθετο παράδειγμα είναι να θέλουμε όλες τις παραγράφους ενός κειμένου να έχουν γκρι γράμματα και λευκό φόντο, τα γράμματα των παραγράφων που έχουν class ίσο με test1 να έχουν μπλε γράμματα και η παράγραφος που έχει id ίσο με test2 να έχει χρώμα φόντου γκρι και γράμματα κόκκινα, τότε ο κώδικας είναι ο εξής:

```
p{
    color: grey;
}
.test1{
    color: blue;
}
#test2{
    background-color: gray;
    color: red;
}
```

Κώδικας 3.5 Σύνθετο παράδειγμα CSS

Τέλος οι τρόποι ενσωμάτωσης του CSS στον κώδικα html είναι τρείς:

- Εξωτερικός ορισμός του κώδικα CSS με χρήση της ετικέτας link

- Εσωτερικός ορισμός του κώδικα CSS στην ετικέτα head
- Ενσωματωμένος ορισμός του κώδικα CSS σε κάθε ετικέτα

3.1.3 JavaScript

Η JavaScript αποτελεί αντικειμενοστραφή γλώσσα **σεναρίων** και όχι γλώσσα προγραμματισμού [6][9][10][11][15]. Ο τρόπος με τον οποίο συνδέεται με την γλώσσα προγραμματισμού Java είναι το γεγονός ότι και οι δύο είναι αντικειμενοστραφής και έχουν την ίδια σύνταξη. Οι βασικές αρχές σχεδιασμού της προέρχονται από τις γλώσσες Self και Scheme.

Ο κώδικας της JavaScript εκτελείται από τον περιηγητή, μέσω του ενσωματωμένου JavaScript interpreter και αποτελεί τον βασικό λόγο που η JavaScript δεν είναι γλώσσα προγραμματισμού. Επίσης το γεγονός αυτό δίνει την δυνατότητα σχεδιασμού δυναμικών ιστοσελίδων με τον κώδικα να «τρέχει» στον υπολογιστή του χρήστη και έτσι δεν «φορτώνεται» ο server με λειτουργίες που μπορούν να εκτελεστούν στην μεριά του χρήστη.

Βασική λειτουργία της JavaScript είναι ότι μπορεί να κάνει το περιεχόμενο μιας ιστοσελίδας πιο δια δραστική με τον χρήστη, αφού υπάρχει η δυνατότητα αντίδρασης σε διάφορα γεγονότα (events) όπως το πάτημα ενός click πάνω σε κάποιο σημείο της ιστοσελίδας.

Στο σημείο αυτό καλό θα ήταν να αναφερθεί ότι στην εφαρμογή έχει χρησιμοποιηθεί η βιβλιοθήκη jQuery της JavaScript, η οποία αποτελεί την πλέον γνωστή βιβλιοθήκη της. Ένα από τα μεγαλύτερα πλεονεκτήματα της jQuery είναι το γεγονός ότι υλοποιεί την τεχνολογία AJAX, που προέρχεται από τα αρχικά Asynchronous JavaScript and XML, με μεγάλη ευκολία. Η τεχνολογία αυτή χρησιμοποιείται κατά κόρον στην εφαρμογή ώστε να μην χρειαστεί να φορτώσει πάλι μια σελίδα στέλνοντας κάποια δεδομένα στην βάση αλλά να το κάνει μέσω της ήδη φορτωμένης σελίδας. Επίσης η jQuery χρησιμοποιήθηκε πολύ στην εφαρμογή διότι χειρίζεται πολύ πιο εύκολα τα γεγονότα (events).

3.1.4 Γλώσσα PHP

Η PHP ανήκει στις γλώσσες **σεναρίων** [6][8][10][11][15]. Χρησιμοποιείται για την δημιουργία ιστοσελίδων με δυναμικό περιβάλλον. Μια ιστοσελίδα είναι δυνατόν να αναπτυχθεί μόνο με χρήση της php. Αυτό δεν είναι όμως μια καλή χρήση διότι όλα τα σενάρια τρέχουν στην μεριά του server όποτε και τον επιβαρύνουν. Βασική ιδιότητα της php είναι ότι καθορίζει τη συμπεριφορά και τη λογική που ακολουθεί η εφαρμογή.

Για να γραφεί ένα σενάριο php αρκεί ο κώδικας να περιβληθεί από την ετικέτα έναρξης της php η οποία είναι “<?php” και την ετικέτα τερματισμού της php η οποία είναι “?>”. Ένα παράδειγμα είναι όταν θέλουμε να εμφανίσουμε το άθροισμα των αριθμών 5 και 3 όπου ο κώδικας για να γίνει αυτό είναι:

```php
<?php
    echo 5+3;
?>
```

Κώδικας 3.6 Βασική δομή κώδικα PHP

3.1.5 MySQL

Η MySql είναι ένα σύστημα διαχείρισης βάσεων δεδομένων ανοιχτού κώδικα[6][7][10][11][15][16]. Χρησιμποιείται από πολλές εφαρμογές και σουίτες δημιουργίας ιστοσελίδων (CMS). Βασίζεται στην γλώσσα προγραμματισμού SQL που είναι το ακρωνύμιο για τις λέξεις Structure Query Language.

Μία βάση δεδομένων ουσιαστικά είναι ένας αποθηκευτικός χώρος όπου εκεί θα αποθηκευτούν πληροφορίες που δεν πρέπει να χαθούν, όπως για παράδειγμα οι λογαριασμοί χρηστών μιας εφαρμογής ή γενικά όποιο δυναμικό στοιχείο χρειάζεται να διατηρηθεί.

Οι βάσεις δεδομένων περιέχουν πίνακες , μέσα στους οποίους είναι αποθηκευμένες οι πληροφορίες γραμμή προς γραμμή. Συνδέονται μεταξύ τους με χρήση πρωτεύοντος κλειδιού και ξένου κλειδιού. Το πρωτεύον κλειδί είναι το πεδίο του πίνακα που καθιστά κάθε εγγραφή του μοναδική. Το ξένο κλειδί θεωρείτε ένα πεδίο του πίνακα το οποίο αποτελεί πρωτεύον κλειδί σε άλλο πίνακα και τα δύο αυτά συνδέονται άμεσα.

Οι πληροφορίες που βρίσκονται στην βάση δεδομένων επιστρέφουν στην εφαρμογή με χρήση ερωτημάτων (queries). Μερικές από αυτές είναι:

- Select
- Insert
- Update
- Delete

Ένα παράδειγμα ερωτήματος είναι αυτό που παίρνει στοιχεία από ένα πίνακα. Ας υποθέσουμε ότι έχουμε ένα πίνακα με όνομα users τότε για να πάρουμε όλους του χρήστες και όλα τα στοιχεία τους αρκεί να τρέξουμε το εξής ερώτημα:

```
SELECT * FROM users
```

Κώδικας 3.7 Δομή ερωτήματος SELECT της MySQL

3.2 Εργαλεία ανάπτυξης εφαρμογής

3.2.1 XAMPP

Για την ανάπτυξη της εφαρμογής, χρειάστηκε να δημιουργηθεί ένα τοπικός server στον υπολογιστή. Για τον σκοπό αυτό χρησιμοποιήθηκε το λογισμικό xampp της apache.

Το XAMPP είναι ένα πακέτο προγραμμάτων ελεύθερου λογισμικού, λογισμικού ανοικτού κώδικα και ανεξαρτήτου πλατφόρμας το οποίο περιέχει το εξυπηρετητή ιστοσελίδων http Apache, την βάση δεδομένων MySQL και ένα διερμηνέα για κώδικα γραμμένο σε γλώσσες προγραμματισμού PHP και Perl. Η λέξη xampp είναι το ακρωνύμιο των λέξεων X (cross-platform) Apache MySQL Php Perl[17].

Το XAMPP μπορεί να χρησιμοποιηθεί τόσο τοπικά όσο και για την φιλοξενία ιστοσελίδων στο διαδίκτυο.

3.2.2 Brackets

Στην αρχή της εργασίας, προσπάθησα να γράψω τον κώδικα στην εφαρμογή notepad που παρέχεται μαζί με την διανομή των windows, πράγμα το οποίο αποδείχθηκε πολύ δύσκολο, καθώς δεν παρέχει καμία διευκόλυνση και εξειδικευμένη λειτουργεία για την συγγραφή κώδικα όπως οι διαφορετικοί χρωμματισμοί των εντολών ή των σχολίων. Έτσι αποφάσησα να αλλάξω τον κεμενογράφω και επέλεξα το brackets.

Το Brackets είναι ένας κειμενογράφος ανοιχτού κώδικα για προγραμματιστές. Έχει γραφεί σε γλώσσα scripting και συγκεκριμένα σε JavaScript [18]. Δίνει την δυνατότητα για διάφορα plugins που έχουν αναπτυχθεί από προγραμματιστές είτε της εταιρίας Adobe, που είναι και η κατασκευάστρια εταιρία, είτε από διάφορους χρήστες του προγράμματος αυτού.

Θα μπορούσαν να χρησιμοποιηθούν διάφορα προγράμματα για την ανάπτυξη της εφαρμογής όπως το notepad++ ή κάποιο πρόγραμμα που απαιτεί πληρωμή όπως το Dreamweaver της ίδιας εταιρίας.

Ο λόγος που χρησιμοποιήθηκε το brackets είναι διότι είναι δωρεάν σε αντίθεση με το Dreamweaver καθώς επίσης παρέχει προτάσεις σχετικά με το τι θα ήθελε να γράψει ο προγραμματιστής της εκάστοτε εφαρμογής, βάση της γλώσσας στην οποία γράφει.

3.3 Ανάλυση βασικών σημείων εφαρμογής (κώδικας)

3.3.1 Κοινά σημεία κώδικα στα αρχεία

Σε όλα τα αρχεία υπάρχουν κοινά κομμάτια κώδικα, τα οποία έχουν να κάνουν με ελέγχους ή με κάποιες συναρτήσεις. Για παράδειγμα στο κομμάτι κώδικα Κώδικας 3.8 φαίνεται ο τρόπος με τον οποίο γίνεται έλεγχος στα αρχεία που αφορούν τον διαχειριστή αν ο χρήστης που προσπαθεί να τα χρησημοποιήσει είναι διαχειριστής ή όχι, και στην περίπτωση που δεν είναι τον μεταφέρει στην αρχική σελίδα της εφαρμογής.

```php
<?php
session_start();
if(!isset($_SESSION['superAdmin'])){
    if(isset($_SESSION['isLogedIn'])){
        header('Location: ../index.php');
        exit;
    }
    else{
        header('Location: ../login.php');
        exit;
    }
}
?>
```

Κώδικας 3.8 Έλεγχος τύπου χρήστη σε σελίδες του διαχειριστή

Για τον έλεγχο χρησιμοποιείται μία σύνοδος (SESSION). Η σύνοδος είναι μια μεταβλητή του συστήματος, η οποία αποθηκεύεται στην μεριά του server. Θα μπορούσε να χρησιμοποιηθεί και ένα cookie αλλά επιλέχθηκε η σύνοδος γιατί προσθέτει μεγαλύτερη ασφάλεια στην εφαρμογή, μιας και ένας κακόβουλος χρήστης θα πρέπει να έχει πρόσβαση στον server, που είναι πιο δύσκολο από το να έχει πρόσβαση στον υπολογιστή κάποιου χρήστη, ώστε να δημιουργήσει κάποιο πρόβλημα στην εφαρμογή. Στο κομμάτι του κώδικα Κώδικας 3.8 γίνεται ένας έλεγχος αν υπάρχει η σύνοδος με όνομα superAdmin. Στην περίπτωση που δεν υπάρχει, ελέγχεται αν έχει γίνει σύνδεση του χρήστη στην εφαρμογή, ελέγχοντας αν υπάρχει η σύνοδος με όνομα isLogedIn. Αν υπάρχει τότε σημαίνει ότι ο χρήστης έχει κάνει είσοδο στην εφαρμογή και δεν είναι διαχειριστής οπότε θα τον μεταφέρει στην index.php. Στην περίπτωση που δεν υπάρχει τότε ο χρήστης που προσπάθησε να χρησιμοποιήσει την το συγκεκριμένο αρχείο php πρέπει να συνδεθεί πρώτα οπότε και τον μεταφέρει στην σελίδα εισόδου (login.php). Οι σύνοδοι αυτοί δημιουργούνται κατά την είσοδο του χρήστη και διαγράφονται είτε κλείνοντας τον περιηγητή είτε καλώντας την συνάρτηση της php, unset, με όρισμα την σύνοδο ώστε να την διαγράψει από τον server, όπως θα δούμε στο αρχείο logout.php.

Ένα δεύτερο κομμάτι που είναι ίδιο στα αρχεία είναι το κομμάτι που εμφανίζει τα περιεχόμενα ενός αρχείου php μέσα σε ένα άλλο php. Η διαδικασία αυτή χρησιμοποιείται κυρίως στην εισαγωγή των menu της εφαρμογής και σκοπό έχουν να κάνουν πιο εύκολη την διαχείριση των σελίδων που θα μπορεί να μεταβεί ένας χρήστης μόνο αλλάζοντας ένα αρχείο.

```php
<?php
    include ("header.php");
?>
```

Κώδικας 3.9 Εισαγωγή περιεχομένων αρχείου php σε άλλο αρχείο php

Στο κομμάτι του κώδικα 3.9 φαίνεται η κλήση του αρχείου header.php σε ένα άλλο αρχείο και συγκεκριμένα στο αρχείο index.php της εφαρμογής. Με την εντολή include γίνεται η εισαγωγή των περιεχομένων του αρχείου header.php, που εμφανίζεται στον Κώδικα 3.10, εντός του αρχείου index.php σαν να ήταν γραμμένα στο αρχείο αυτό.

```
<noscript>
    <meta http-equiv="refresh" content="0; url=noscript.php" /><!-- an o xthsths
den exei thn jv ton paei se mia selida pou toy leei pws na tn energopoihsei-->
</noscript>
<meta name="viewport" content="width=device-width, initial-scale=1"><!--
antilambaneitai to mikos ths o8omhs stn poia proballetai-->
<meta http-equiv="Content-Language" content="el">
<meta http-equiv='Content-Type' content='text/html; charset=utf-8' /><!-- utf8 gia
n anagrvrizei ta ellhnika -->
<link rel='stylesheet' type='text/css' href='css/styles.css' />
<link rel='stylesheet' type='text/css' href='css/animate.css' />
<link rel='stylesheet' type='text/css' href='css/font-awesome.css' /><!--
eikonidia-->
<link rel='stylesheet' type='text/css' href='css/bootstrap.min.css' />
<link rel='stylesheet' type='text/css' href='inc/pickadate.js-
master/lib/themes/classic.css' />
<link rel='stylesheet' type='text/css' href='inc/pickadate.js-
master/lib/themes/classic.date.css' />
<link rel='stylesheet' type='text/css' href='inc/custom-dropdown/css/style.css'/>
<link rel='stylesheet' type='text/css'
href='https://cdnjs.cloudflare.com/ajax/libs/rangeslider.js/2.3.0/rangeslider.css'/
>
<link rel="stylesheet" type="text/css" href="inc/sliderengine/amazingslider-
1.css"/>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
<script
src="http://netdna.bootstrapcdn.com/bootstrap/3.0.0/js/bootstrap.min.js"></script>
<script src="inc/pickadate.js-master/lib/picker.js"></script>
<script src="inc/pickadate.js-master/lib/picker.date.js"></script>
<script src="inc/pickadate.js-master/lib/legacy.js"></script>
<script src="inc/pickadate.js-master/lib/translations/el_GR.js"></script>
<script src="inc/custom-dropdown/js/modernizr.custom.79639.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/rangeslider.js/2.3.0/rangeslider.js"></
script>
<script
src="http://netdna.bootstrapcdn.com/bootstrap/3.1.1/js/bootstrap.min.js"></script>
<script src="inc/sliderengine/amazingslider.js"></script>
<script src="inc/sliderengine/initslider-1.js"></script>
```

Κώδικας 3.10 Περιεχόμενα header.php

3.3.2 Σύνδεση με την βάση

Η σύνδεση με την βάση γίνεται σε ξεχοριστό αρχείο (db_connect.php) και ο λόγος είναι για να είναι πιο εύκολο σε περίπτωση αλλαγής του server ή της τοποθεσίας του server η μετακίνηση της βάσης σε νέο server.

```
<?php
function connect(){
$host="localhost";
$usr="root"; // ta stoixeia poy exei h efarmogh wste na sunde8ei sthn bash
$pwd='';
$accounts=mysqli_connect($host,$usr,$pwd,"hotels");
mysqli_query($accounts,'set character set utf8'); //7 8 grammh xrhsimopoioyntai gia
na diabasoume kai na grapsoume ellhnika sthn bash
mysqli_query($accounts,"SET NAMES 'utf8'");
if (!$accounts) {
        $str=mysqli_connect_error();
    echo "<script> console.log($str);</script>";
    exit;
}
return $accounts;
}
function db_query($sql,$con){
        $result=mysqli_query($con,$sql);
        return $result;
}
?>
```

Κώδικας 3.11 Σύνδεση με την βάση

Στον κώδικα 3.11 υπάρχουν δύο συναρτήσεις, η connect και η db_query. Η πρώτη αποτελεί τον τρόπο με τον οποίο η εφαρμογή κάνει την σύνδεση με την βάση και η δεύτερη αποτελεί τον τρόπο με τον οποίο εκτελεί ένα ερώτημα sql στην βάση δεδομένων.

Πιο συγκεκριμένα, όσον αφορά την σύνδεση με την βάση, στον κώδικα 3.11 υπάρχουν τρείς μεταβλητές, η host που είναι η διεύθυνση του server που έχει εγκατεστημένη την βάση, η usr η οποία είναι το όνομα χρήστη για την βάση, το pwd που είναι ο κωδικός του χρήστη για την βάση δεδομένων και η accounts η οποία έχει την σύνδεση με την βάση. Στην συνέχεια εκτελούνται δύο ερωτήματα στην βάση ώστε να επιβεβαιώσουμε την χρήση κωδικοποίησης UTF-8 μεταξύ της βάσης και της εφαρμογής και να μπορούμε να έχουμε ελληνικούς χαρακτήρες στην εφαρμογή. Τέλος ελέγχουμε αν έγινε η σύνδεση και αν έγινε επιστρέφουμε την σύνδεση αυτή στην εφαρμογή, αλλιώς εκτυπώνουμε στην κονσόλα ένα μήνυμα λάθους.

Σχετικά με την δεύτερη συνάρτηση, αυτή της εκτέλεσης του ερωτήματος στην βάση δεδομένων, παίρνει σαν ορίσματα το ερώτημα sql και την σύνδεση με την βάση (η οποία έχει γίνει με την πρώτη συνάρτηση) και επιστρέφει τα αποτελέσματα του ερωτήματος στην εφαρμογή.

Στο ίδιο αρχείο υπάρχει και μια συνάρτηση η οποία παίρνει σαν δεδομένα ένα μέγεθος που δίνει ο χρήστης και δημιουργεί τυχαία αλφαριθμητικά με αυτό το μήκος. Πιο συγκεκριμένα, η συνάρτηση έχει ένα αλφαριθμητικό ($characters) με όλους τους πιθανούς χαρακτήρες που μπορεί να χρησιμοποιήσει για την δημιουργία του νέου αλφαριθμητικού. Στην συνέχεια επιλέγει τυχαίους χαρακτήρες μέσα από το

αλφαριθμητικό $characters μέχρι το μήκος του δημιουργημένου αλφαριθμητικού να είναι ίδιο με αυτό που παίρνει η συνάρτηση ως όρισμα. Η διαδικασία αυτή φαίνεται στον κώδικα 3.12.

```php
<?php
function generateRandomString($length) {// xrhsimopoiheite gia n ftiaksei h php
tyxaioys ari8mous to alati
    $characters = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
    $charactersLength = strlen($characters);
    $randomString = '';
    for ($i = 0; $i < $length; $i++) {
        $randomString .= $characters[rand(0, $charactersLength - 1)];
    }
    return $randomString;
}
?>
```

Κώδικας 3.12 Συνάρτηση δημιουργίας αλφαριθμητικού για το αλάτι (salt)

3.3.3 Αρχείο login.php

Το αρχείο login.php είναι το αρχείο που τρέχει όταν ο χρήστης θέλει να κάνει σύνδεση στην εφαρμογή. Αφού ο χρήστης βάλει τα στοιχεία του στην φόρμα σύνδεσης, τα στοιχεία αποστέλονται στην ίδια σελίδα. Έπειτα γίνεται σύνδεση με την βάση δεδομένων, ελέγχει το όνομα χρήστη που εισήγαγε ο χρήστης και ελέγχει να δεί αν υπάρχει στην βάση το username αυτό. Στην περίπτωση που δεν υπάρχει τότε εμφανίζει το αντίστοιχο μήνυμα.

Στην περίπτωση που υπάρχει το όνομα χρήστη στην βάση τότε ελέγχεται αν ο χρήστης είναι απενεργοποιημένος ή όχι (ελέγχεται αν το status από την βάση δεδομένων για τον συγκεκριμένο χρήστη είναι μηδέν ή όχι αντίστοιχα). Αν είναι απενεργοποιημένος τότε εμφανίζεται το αντίστοιχο μήνυμα.

Εάν ο χρήστης δεν είναι απενεργοποιημένος τότε η εφαρμογή παίρνει από την βάση τον κρυπτογραφημένο κωδικό που αντιστοιχεί στον χρήστη καθώς και το αλάτι (salt) κα το id του χρήστη. Δημιουργεί μία μεταβλητή στην οποία περνάει τον κωδικό ακολουθούμενο από το αλάτι και την κρυπτογραφεί με sha-256 κρυπτό-αλγόριθμο. Εάν ο κωδικός από την βάση δεν είναι ίδιος με την κρυπτογραφημένη μεταβλητή τότε σημαίνει ότι ο κωδικός που δόθηκε από τον χρήστη δεν είναι σωστός και εμφανίζεται σε αυτόν το αντίστοιχο μήνυμα. Σε αντίθετη περίπτωση ελέγχεται ο τύπος του χρήστη (απλός χρήστης, ξενοδόχος, διαχειριστής) και δημιουργείται η αντίστοιχη σύνοδος. Επίσης δημιουργούνται και μερικές σύνοδοι που χρειάζονται για την λειτουργία της εφαρμογής.

Τέλος ελέγχεται αν ο χρήστης πάτησε στο κουμπί είσοδος ή μεταφέρθηκε από άλλη σελίδα (ελέγχοντας αν υπάρχει η μεταβλητή GET με όνομα link) και αν υπάρχει τότε ο χρήστης μεταφέρεται στο link αυτό που έπρεπε να έχει πάει αλλιώς μεταφέρεται στην αρχική σελίδα. Όλα τα παραπάνω φαίνονται στον κώδικα 3.13.

```
        if(isset($_POST['Login']))
        {
                require_once ('db_connect.php');
                $con = connect();
                $usr = $_POST['username'];
                $pwd = $_POST['password'];
                $sql = "SELECT * FROM users where username ='$usr'";
                $result = db_query($sql,$con);
                $rows=mysqli_num_rows($result);
                $row = mysqli_fetch_array($result);
                if($rows ==1 ){
                        if($row['status']=='0'){
            echo "<script type='text/javascript'>
document.getElementById('error_msg').innerHTML = 'Ο Λογαριασμός σας έχει
απενεργοποιηθεί'</script>";// echo = ektupwse
           }else{
               $salt=$row['salt'
               $db_pwd=$row['pwd'];
               $db_id= $row['id'];
               $type=$row['type'];
               $test=$pwd.$salt;
               $pwd2 = hash('sha256',$test
               if ($pwd2 == $db_pwd){
                   if($type==3)
                       $_SESSION['superAdmin']=1;
                   if($type==2)
                       $_SESSION['hotel_manager']=1;
                   $_SESSION['isLogedIn']=1;
                   $_SESSION['id']=$db_id
                   $_SESSION['usr']=$usr;
                   if(isset($_GET['link'])){

                       $link=$_GET['link'];
                       $from=$_GET['from'];
                       $to=$_GET['to'];
                       $town=$_GET['town'];
                       $type=$_GET['type'];
                       $link=$link."&from=$from"."&to=$to"."&town=$town"."&type=$type";
                       echo "<script>alert('$link');</script>";
                       header ("Location: $link");
                       exit;
                   }else{
                       echo "<script>alert('down');</script>";
                       header ('Location: ./index.php');
                       exit;
                   }

               }
               else{
                   echo "<script type='text/javascript'>
document.getElementById('error_msg').innerHTML = 'Λάθος στοιχεία'</script>";
               }
           }
                  }
                  else{
                          echo "<script type='text/javascript'>
document.getElementById('error_msg').innerHTML = 'Λάθος στοιχεία'</script           }
        mysqli_close($con);         //kleinei thn sundesh me thn bash
        }
?>
```

Κώδικας 3.13 Βασική λειτουργία της σελίδας login.php

3.3.4 Σελίδα αποσύνδεσης logout.php

Στην σελίδα αυτή γίνεται η διαδικασία της αποσύνδεσης κάποιου χρήστη. Πιο συγκεκριμένα, αν κάποιος πατήσει στο κουμπί της αποσύνδεσης, τότε η εφαρμογή θα διαγράψει όλες της συνόδους που μπορεί να έχουν δημιουργηθεί από την εφαρμογή. Η διαδικασία αυτή φαίνεται στον κώδικα 3.14.

```php
<?php
session_start();
unset($_SESSION['isAdmin']);//diagrafei tis sunodous
unset($_SESSION['hotel_manager']);
unset($_SESSION['id']);
unset($_SESSION['usr']);
unset($_SESSION['isLogedIn']);
unset($_SESSION['superAdmin']);
header('Location: index.php');
exit;

?>
```

Κώδικας 3.14 Κώδικας αποσύνδεσης χρήστη από την εφαρμογή

3.3.5 Αρχική σελίδα εφαρμογής.

Στην αρχική σελίδα της εφαρμογής, κατά την διάρκεια που ο χρήστης πληκτρολογεί την προς αναζήτηση περιοχή ή ξενοδοχείο, στο αντίστοιχο πεδίο, ένα script τρέχει και επιστρέφει από την βάση ότι ταιριάζει με το αλφαριθμικτικό που έχει πληκτρολογηθεί, αφού πρώτα γίνει ένας έλεγχος αν το πλήκτρο που πατήθηκε δεν είναι το κάτω βέλος είτε το πάνω βέλος ή το enter. Στην περίπτωση που είναι το κάτω βέλος τότε μαρκάρεται το πρώτο (ή επόμενο αν έχει μαρκαριστεί ήδη κάποιο) αποτέλεσμα ενώ με το πάνω μαρκάρεται το προηγούμενο αποτέλεσμα. Στην περίπτωση του enter κάνει προσομοίωση το click του ποντικιού πάνω στο αποτέλεσμα που έχει καρκαριστεί με αποτέλεσμα να εμφανιστούν οι πληροφορίες που πρέπει να εμφανιστούν.

Σε κάθε άλλη περίπτωση, το αλφαριθμητικό αποστέλλεται στο αρχείο search.php ώστε να γίνει η αναζήτησή του στην βάση δεδομένων και επιστρέφονται στην index.php τα αποτελέσματα αυτά που το περιέχουν.

Η διαδικασία αυτή γίνεται μέσω της τεχνολογίας ajax. Πιο αναλυτικά, η ajax που χρησιμοποιείται παρουσιάζεται στον κώδικα 3.15.

```
$.ajax({
        dataType:'json',
        type:'POST',
        url:'search.php',
        data: {
            search: "search",
            query: search_term
        },
        success: function(result){
            for(var i=0;i<result.length;i++){
                if(result[i].icon==1){
                    var icon="fa fa-map-marker";
                }else if(result[i].icon==2){
                    var icon="fa fa-bed";
                }
                else{
                    var icon=3;
                }
                if (result[i].icon == 2){
                    $("#search_result_list").append("<li
id='result_"+result[i].id+"_"+result[i].icon+"_"+result[i].town+"'
class='search_result_icon_list_item hotel'><span class='search_result_icon'><i
class='"+icon+" fa-2x'></i></span><div class='search_result_more'><span
class='search_result_title'>"+result[i].name+"</span><span
class='search_result_subtitle'>"+result[i].nomos+"</span></div></li>");
                }else{
                   $("#search_result_list").append("<li
id='result_"+result[i].id+"_"+result[i].icon+"'
class='search_result_icon_list_item'><span class='search_result_icon'><i
class='"+icon+" fa-2x'></i></span><div class='search_result_more'><span
class='search_result_title'>"+result[i].name+"</span><span
class='search_result_subtitle'>"+result[i].nomos+"</span></div></li>");
                }
            }
            $("#search_results_loader").css('display','none');
        }
});
```

Κώδικας 3.15 Ajax call για την επιστροφή αποτελεσμάτων από την αναζήτηση του χρήστη

```php
<?php
if(isset($_POST['search'])){
    $search_term=$_POST['query'];
    require_once "db_connect.php";
    $con=connect();
    $sql="select * from towns where name like '$search_term%' ORDER BY name";
    $result=db_query($sql,$con);
       $rows=array();
       while($row=mysqli_fetch_array($result)){
        $row['icon']=1;
              $rows[]=$row;
       }
    $sql="select * from towns where nomos like '$search_term%' ORDER BY name";
    $result=db_query($sql,$con);
    while($row=mysqli_fetch_array($result)){
        $row['icon']=1;
              $rows[]=$row;
       }
    $sql="select * from hotels where name like '$search_term%' ORDER BY name";
    $result=db_query($sql,$con);
    while($row=mysqli_fetch_array($result)){
        $row['icon']=2;
        $sql2="select towns.name,towns.nomos from towns where
id='".$row['town']."'";
        $result2=db_query($sql2,$con);
        $row2=mysqli_fetch_array($result2);
        $row['town_name']=$row2['name'];
        $row['town_nomos']=$row2['nomos'];
              $rows[]=$row;
       }
    mysqli_close($con);
    print json_encode($rows);

}
?>
```

Κώδικας 3.16 Κώδικας search.php

Στον κώδικα 3.16 φαίνεται ο κώδικας της search.php. Ο κώδικας αυτός παίρνει από την εφαρμογή με POST μέθοδο, το τι έχει γράψει ο χρήστης και αναζητεί στην βάση στον πίνακα towns ονόματα πόλεων, ονόματα νομών καθώς και ξενοδοχεία τα οποία περιέχουν αυτό που έχει γράψει ο χρήστης και τα επιστρέφει στην εφαρμογή.

Στην συνέχεια αν ο χρήστης πατήσει σε κάποιο από τα αποτελέσματα (ή πατήσει το enter) τότε θα εμφανιστούν τα ξενοδοχεία της περιοχής ή η πληροφορίες του ξενοδοχείου (αν ο χρήστης πάτησε σε ξενοδοχείο). Η διαδικασία αυτή γίνεται με την κλίση της συνάρτησης getData() που υπάρχει στο αρχείο myFanctions.js και φαίνεται στον κώδικα 3.17.

```javascript
function getData(){
    $("#results_list").empty();
    var isHotel=false;
    console.log(isHotel);
    if($("#check_out_submit").val() != ''){
        if($(".search_query").hasClass('hotel')){
            var temp=$(".search_query").attr('id');
            temp=temp.split("_");
            var id=temp[1];
```

```
            var from=$(document).find("#check_in_submit").val();
            var to=$(document).find("#check_out_submit").val();
            var type=$(document).find("#dd span").attr('id');
            var town=temp[3];
            type=type.split("_");
            type=type[1];
            var url;
            url="view_more.php";
            var form = $('<form action="' + url + '" method="get">' +
                '<input type="text" name="id" value="' + id + '" />' +
                '<input type="text" name="from" value="' + from + '" />' +
                '<input type="text" name="to" value="' + to + '" />' +
                '<input type="text" name="type" value="' + type + '" />' +
                '<input type="text" name="town" value="' + town + '" />' +
              '</form>');
            $('body').append(form);
            form.submit();
        }
        var temp=$(".search_query").attr('id');
        temp=temp.split("_");
        var id=temp[1];
        var type=temp[2];
        var from=$("#check_in_submit").val();
        var to=$("#check_out_submit").val();
        var type=$("#dd span").attr('id');
        type=type.split("_");
        type=type[1];
        $.ajax({
            dataType:'json',
            type:'POST',
            url:'display_results.php',
            data: {
                display: "display",
                type: type,
                id: id,
                from: from,
                to: to,
                type: type
            },
            success: function(result){
                //console.log(result);
                if(result.found!=0){
                    var hotels=Array();
                    for (var i=0;i<result.length;i++){
                        var hotel_id=result[i].id;
                        var name=result[i].name;
                        var img=name.replace(/ /g,"_");
                        img=img+"/"+result[i].main_photo;
                        var discount = result[i].discount;
                        var paroxes=result[i].paroxes;
                        var paroxes_dwmatiou=result[i].paroxes_dwmatiou;
                        var price=result[i].price;
                        var asteria=result[i].stars;
                        var poli=result[i].poli;
                        var nomos=result[i].nomos;
                        var rating=result[i].rating;
                        var discounted_price=price-price*(result[i].discount/100);
                        var room_id=result[i].room_id;
                        var user_id=result[i].user_id;
                        var db_rating=result[i].db_rating;
                        var user_rating=result[i].user_rating;
                        console.log('rating= '+rating+ 'db_ratong= '+db_rating);
```

```
                        var
html=createHotelHtml(rating,price,discounted_price,asteria,paroxes,paroxes_dwmatiou
,poli,nomos,img,name,discount,room_id,hotels,hotel_id,db_rating,user_rating);

                        $("#results_list").append(html);
                        //getRating(user_id,hotel_id);

                    }

                    $('#price_slider').empty();
                    var min_price=10000000;
                    var max_price=0;
                    $(document).find(".teliki_timi").each(function(){
                        var teliki_timi=$(this).html().slice(0,-1);
                        teliki_timi=parseInt(teliki_timi);
                        if(min_price>teliki_timi){
                            min_price=teliki_timi;
                        }
                        if(max_price<teliki_timi){
                            max_price=teliki_timi;
                        }
                    });
                    if (navigator.userAgent.toLowerCase().indexOf('firefox') > -1)
{
                        document.styleSheets[0].insertRule(".rangeslider:after{
content: '"+max_price+"';}",document.styleSheets[0].cssRules.length);
                        document.styleSheets[0].insertRule(".rangeslider:before{
content: '"+min_price+"';}",document.styleSheets[0].cssRules.length);
                    }
                    else{

document.styleSheets[0].addRule(".rangeslider:after",'content:
"'+max_price+'";',document.styleSheets[0].cssRules.length);

document.styleSheets[0].addRule(".rangeslider:before",'content:
"'+min_price+'";',document.styleSheets[0].cssRules.length);
                    }
                    $("#rangeslider").after(max_price);
                    $('#price_slider').append('<input id="range" type="range"
min="'+min_price+'" max="'+max_price+'" value="'+max_price+'" data-rangeslider/>');
                    $('#range').rangeslider({
                        // Deactivate the feature detection
                        polyfill: false,
                       onInit: function(position,value) {
                           $("#range_output").val("max: "+$("#range").val());
                       },
                       onSlide: function(position, value) {
                            $("#range_output").val("max: "+value);
                            $(".hotel").each(function(){
                                var
price=$(this).find(".teliki_timi").html().slice(0,-1);
                                if(value<price){
                                    $(this).css('display','none');
                                }else{
                                    $(this).css('display','list-item');
                                }
                            });
                        }
                    });
                    $("#filters").removeClass('filters_none');
                    $("#filters").addClass('filters_block');
                                $("#results").removeClass('results_none');
```

```
                    $("main").css('display','block');
                }else{
                    $("#filters").removeClass('filters_block');
                    $("#results").addClass('results_none');
                    $("#filters").addClass('filters_none');
                    $("#results_list").html("<h2> Δεν βρέθηκε ξενοδοχείο με τα
δεδομένα που δώσατε!</h2>");
                    $("main").css('display','block');
                }
            }
    })

    }
}
```

Κώδικας 3.17 Συνάρτηση getData()

Η συνάρτηση αυτή, ελέγχει αν πατήθηκε ξενοδοχείο και μεταφέρει τον χρήστη στην σελίδα πληροφοριών του ξενοδοχείου. Σε αντίθετη περίπτωση, αποθηκεύει τις πληροφορίες του check-in check-out και τον τύπο του δωματίου που έχει επιλέξει ο χρήστης και τα στέλνει με χρήση της ajax στην σελίδα display_results.php, όπου ελέγχονται τα ξενοδοχεία που έχουν διαθέσιμα δωμάτια για την περίοδο που θέλει ο χρήστης και επιστρέφονται όλα στην getData(). Στην συνέχεια η getData() τα εμφανίζει στην αρχική της εφαρμογής. Η σελίδα display_results.php φαίνεται στον κώδικα 3.18.

```
<?php
session_start();
if(isset($_POST['display'])){
    $id=$_POST['id'];
    $type=$_POST['type'];
    $from=$_POST['from'];
    $to=$_POST['to'];
    $type=$_POST['type'];
    $dates_search='';
    $period = new DatePeriod(
     new DateTime($from),
     new DateInterval('P1D'),
     new DateTime($to)
    );
    $dates=Array();
    foreach($period as $date){
        $dates[]=$date->format("Y-m-d");
    }
    for ($i=0;$i<count($dates)-1;$i++){
        $dates_search.="NOT('$dates[$i]' BETWEEN closed_rooms.from_date and
DATE_SUB(closed_rooms.to_date,INTERVAL 1 DAY)) AND ";
    }
    $last=$dates[count($dates)-1];
    $dates_search.="NOT('$last' BETWEEN closed_rooms.from_date and
DATE_SUB(closed_rooms.to_date,INTERVAL 1 DAY))";
    require_once "db_connect.php";
    $con=connect();
    if($type==1 || $type==2 || $type == 3){
        $sql="SELECT hotels.id as
id,hotels.name,hotels.status,description,creator_id,main_photo,other_photos,paroxes
,town,stars,lat,lng,rooms.type,rooms.price,rooms.discount,rooms.paroxes_dwmatiou,ro
oms.hotel_id,rooms.id as room_id,towns.nomos,towns.poli_perifereia,towns.name as
poli
```

```
from hotels
inner join towns on towns.id='$id'
inner join rooms on (rooms.hotel_id=hotels.id AND rooms.type='$type')
where hotels.town='$id' AND hotels.status!='0' ORDER BY (price-
price*discount/100)"  ;
    }
    $result=db_query($sql,$con);
    $rows=array();
    if(mysqli_num_rows($result)==0){
        $rows['found']=0;
        $rows['sql']=$sql;
    }
    else{
        $count=0;
        while($row=mysqli_fetch_array($result)){
            $hotelid=$row['id'];
            $room_id=$row['room_id'];
            $sql2="Select * from user_ratings where hotel_id='$hotelid'";
            $result2=db_query($sql2,$con);
            $sql3="select * from closed_rooms where room_id='$room_id'";
            $result3=db_query($sql3,$con);
            $flag=true;
            $dates_test=array();
            while($row3=mysqli_fetch_array($result3)){
                if(mysqli_num_rows($result3)!=0){
                    $from_date=$row3['from_date'];
                    $to_date=$row3['to_date'];
                    $dd=$from_date." ".$to_date;
                    $row['date11']=$dd;
                    for($i=0;$i<count($dates);$i++){
                        $dates_test[]=$dates[$i]."<".$to_date." &&
".$dates[$i].">=".$from_date;
                        if($dates[$i]<$to_date && $dates[$i]>=$from_date){
                            $flag=false;
                        }
                    }
                }
                $row['dates']=$dates;
                $row['flag']=$flag;
                if($flag==false){
                    break;
                }
            }
            $row['dates_test']=$dates_test;
            $rating_num=0;
            $rating_sum=0;
            $user_rating='-1';
            if(isset($_SESSION['id'])){
                $user_id=$_SESSION['id'];
                $hotel_id=$row['id'];
                $sql4="select rating from user_ratings where user_id='$user_id' and
hotel_id='$hotel_id'";
                $result4=db_query($sql4,$con);
                $user_rating='-1';
                if(mysqli_num_rows($result)==0){
                    $user_rating='-1';
                }else{
                    $row4=mysqli_fetch_array($result4);
                    $user_rating=$row4['rating'];
                }
                $sql4="select rating from user_ratings where hotel_id='$hotel_id'";
                $result4=db_query($sql4,$con);
```

```
                $sum=0;
                $count=0;
                while($row4=mysqli_fetch_array($result4)){
                    $sum+=$row4['rating'];
                    $count++;
                }
                if($count==0){
                    $db_rating=-1;
                }else{
                    $db_rating=$sum/$count;
                }
            }
            else{
                $user_id='-1';
                $hotel_id=$row['id'];
                $sql4="select rating from user_ratings where hotel_id='$hotel_id'";
                $result4=db_query($sql4,$con);
                $sum=0;
                $count=0;
                while($row4=mysqli_fetch_array($result4)){
                    $sum+=$row4['rating'];
                    $count++;
                }
                if($count==0){
                    $db_rating=-1;
                }else{
                    $db_rating=$sum/$count;
                }
                //$db_rating='-1';
            }
            if($flag==true){
                $count++;
                while($rating_row=mysqli_fetch_array($result2)){
                    $rating_num++;
                    $rating_sum+=$rating_row['rating'];
                    $row['sql']=$sql;
                }
                if($rating_num==0){
                    $row['rating']=-1;
                    $row['user_id']=$user_id;
                    $row['db_rating']=$db_rating;
                    $row['user_rating']=$user_rating;
                    $rows[]=$row;
                }else{
                    $row['rating']=$rating_sum/$rating_num;
                    $row['period']=$dates;
                    $row['sql']=$sql;
                    $row['user_id']=$user_id;
                    $row['db_rating']=$db_rating;
                    $row['user_rating']=$user_rating;
                    $rows[]=$row;
                }
            }
        }
        if($count==0){
            $rows['found']=0;
            $rows['sql']=$sql;
        }
    }
    mysqli_close($con);
    print json_encode($rows);
}
```

```
if(isset($_POST['display_hotel'])){
    $id=$_POST['id'];
    $town=$_POST['town'];
    $type=$_POST['type'];
    $from=$_POST['from'];
    $to=$_POST['to'];
    $type=$_POST['type'];
    $dates_search='';
    $period = new DatePeriod(
     new DateTime($from),
     new DateInterval('P1D'),
     new DateTime($to)
    );
    $dates=Array();
    foreach($period as $date){
        $dates[]=$date->format("Y-m-d");
    }
    for ($i=0;$i<count($dates)-1;$i++){
        $dates_search.="NOT('$dates[$i]' BETWEEN closed_rooms.from_date and
DATE_SUB(closed_rooms.to_date,INTERVAL 1 DAY)) AND ";
    }
    $last=$dates[count($dates)-1];
    $dates_search.="NOT('$last' BETWEEN closed_rooms.from_date and
DATE_SUB(closed_rooms.to_date,INTERVAL 1 DAY))";
    require_once "db_connect.php";
    $con=connect();
    if($type==1 || $type==2){
        $sql="SELECT hotels.id as
id,hotels.name,description,creator_id,main_photo,other_photos,`status`,paroxes,town
,stars,lat,lng,rooms.type,rooms.price,rooms.discount,rooms.paroxes_dwmatiou,rooms.h
otel_id,rooms.id as room_id,towns.nomos,towns.poli_perifereia,towns.name as poli
from hotels
inner join towns on towns.id='$town'
inner join rooms on (rooms.hotel_id=hotels.id AND rooms.type='$type')
where hotels.town='$town' AND hotels.id='$id' ORDER BY (price-price*discount/100)"
;
    }
    $result=db_query($sql,$con);
    $rows=array();
    if(mysqli_num_rows($result)==0){
        $rows['found']=0;
        $rows['sql']=$sql;
    }
    else{
        $count=0;
        while($row=mysqli_fetch_array($result)){
            $hotelid=$row['id'];
            $room_id=$row['room_id'];
            $sql2="Select * from user_ratings where hotel_id='$hotelid'";
            $result2=db_query($sql2,$con);
            $sql3="select * from closed_rooms where room_id='$room_id'";
            $result3=db_query($sql3,$con);
            $flag=true;
            while($row3=mysqli_fetch_array($result3)){
                if(mysqli_num_rows($result3)!=0){
                    $from_date=$row3['from_date'];
                    $to_date=$row3['to_date'];
                    $dd=$from_date." ".$to_date;
                    $row['date11']=$dd;
                    for($i=0;$i<count($dates);$i++){
                        if($dates[$i]<$to_date && $dates[$i]>=$from_date){
                            $flag=false;
```

```
                    }
                }
            }
            $row['dates']=$dates;
            $row['flag']=$flag;
            if($flag==true){
                break;
            }
        }
        $rating_num=0;
        $rating_sum=0;
        if($flag==true){
            $count++;
            while($rating_row=mysqli_fetch_array($result2)){
                $rating_num++;
                $rating_sum+=$rating_row['rating'];
                $row['sql']=$sql;
            }
            if($rating_num==0){
                $row['rating']=-1;
                $rows[]=$row;
            }else{
                $row['rating']=$rating_sum/$rating_num;
                $row['period']=$dates;
                $row['sql']=$sql;
                $rows[]=$row;
            }
        }
    }
    if($count==0){
        $rows['found']=0;
        $rows['sql']=$sql;
    }
}
    mysqli_close($con);
    print json_encode($rows);
}
if(isset($_POST['display_hotel_details'])){
    $id=$_POST['hotel_id'];
    require_once 'db_connect.php';
    $con=connect();
    $sql="select hotels.id as
id,hotels.name,hotels.description,hotels.main_photo,hotels.other_photos,hotels.paro
xes,hotels.stars,hotels.lat,hotels.lng,paroxes_xenodoxiou.id as
paroxes_id,paroxes_xenodoxiou.name as paroxes_name from hotels inner join
paroxes_xenodoxiou where hotels.id='$id'";
    $result=db_query($sql,$con);
    $rows=array();
    while($row=mysqli_fetch_array($result)){
        $rows[]=$row;
        $rows['sql']=$sql;
    }
    mysqli_close($con);
    print json_encode ($rows);
}
if(isset($_POST['insert_room'])){
    $id=$_POST['hotel_id'];
    require_once "db_connect.php";
    $con=connect();
    $sql="select * from paroxes_dwmatiou";
    $result=db_query($sql,$con);
    $rows=array();
```

```
    while($row=mysqli_fetch_array($result)){
        $rows[]=$row;
        $rows['sql']=$sql;
    }
    mysqli_close($con);
    print json_encode ($rows);
}
if(isset($_POST['edit_room'])){
    $id=$_POST['hotel_id'];
    $type=$_POST['type'];
    require_once "db_connect.php";
    $con=connect();
    $sql="select * from rooms where type='$type' and hotel_id='$id'";
    $result=db_query($sql,$con);
    $rows=array();
    while($row=mysqli_fetch_array($result)){
        $rows[]=$row;
        $rows['sql']=$sql;
    }
    mysqli_close($con);
    print json_encode ($rows);
}
if(isset($_POST['edit_room2'])){
    $id=$_POST['room_id'];
    require_once "db_connect.php";
    $con=connect();
    $sql="select rooms.id as
room_id,rooms.type,rooms.price,rooms.discount,rooms.paroxes_dwmatiou,paroxes_dwmati
ou.id as paroxes_id,paroxes_dwmatiou.name as paroxes_name from rooms inner join
paroxes_dwmatiou where rooms.id='$id'";
    $result=db_query($sql,$con);
    $rows=array();
    while($row=mysqli_fetch_array($result)){
        $rows[]=$row;
        $rows['sql']=$sql;
    }
    mysqli_close($con);
    print json_encode ($rows);
}
if(isset($_POST['get_paroxes'])){
    require_once "db_connect.php";
    $con=connect();
    $sql="select paroxes_xenodoxiou.id as paroxes_id,paroxes_xenodoxiou.name as
paroxes_name from paroxes_xenodoxiou";
    $result=db_query($sql,$con);
    $rows=array();
    while($row=mysqli_fetch_array($result)){
        $rows[]=$row;
        $rows['sql']=$sql;
    }
    mysqli_close($con);
    print json_encode ($rows);
}
if(isset($_POST['get_towns'])){
    $nomos=$_POST['nomos'];
    require_once "db_connect.php";
    $con=connect();
    $sql="select * from towns where nomos='$nomos'";
    $result=db_query($sql,$con);
    $rows=array();
    while($row=mysqli_fetch_array($result)){
        $rows[]=$row;
```

```
        $rows['sql']=$sql;
    }
    mysqli_close($con);
    print json_encode ($rows);
}
if(isset($_POST['getRating'])){
    $user_id=$_POST['user_id'];
    $hotel_id=$_POST['hotel_id'];
    require_once "db_connect.php";
    $con=connect();
    $sql="select rating from user_ratings where user_id='$user_id' AND
hotel_id='$hotel_id'";
    $result=db_query($sql,$con);
    if(mysqli_num_rows($result)>0){
        $row=mysqli_fetch_array($result);
        $rating=$row['rating'];
    }else{
        $rating='-1';
    }
    mysqli_close($con);
    echo ($rating);
}
if(isset($_POST['fillClosedRooms'])){
    require_once 'db_connect.php';
    $con=connect();
    $id=$_POST['id'];
    $sql="SELECT
kratiseis.id,kratiseis.from_date,kratiseis.to_date,kratiseis.hotel_id,user_id,krati
seis.room_id,hotels.name,users.lname,users.fname FROM kratiseis inner join hotels
on kratiseis.hotel_id=hotels.id inner join users on users.id=kratiseis.user_id
WHERE hotel_id='$id'";
    $result=db_query($sql,$con);
    $rows=array();
    while($row=mysqli_fetch_array($result)){
        $rows[]=$row;
    }
    mysqli_close($con);
    echo json_encode($rows);
}
?>
```

Κώδικας 3.18 Αρχείο display_results.php

Στην ουσία δημιουργεί ένα πίνακα από τις ημερομηνίες που παρεμβάλλονται μεταξύ της ημερομηνίας check-in και της ημερομηνίας check-out. Έπειτα βρίσκει τα ξενοδοχεία που έχουν διαθέσιμα δωμάτια για την περίοδο αυτή και επιστρέφει πίσω τις πληροφορίες του ξενοδοχείου (π.χ. αστέρια βαθμολογία) καθώς και των δωματίων.

Τέλος για να γίνει μια κράτηση από την αρχική σελίδα, αφού έχουν επιστραφεί όλα τα διαθέσιμα δωμάτια και ξενοδοχεία, μπορεί ο χρήστης να κάνει click στο κουμπί της κράτησης. Από εκεί θα μεταφερθεί στην σελίδα κράτησης όπου με δεδομένα που έχουν αποσταλεί με την μέθοδο GET θα μπορέσει η εφαρμογή να κάνει κράτηση στο σωστό δωμάτιο μετά από επιβεβαίωση του χρήστη.

3.3.6 Ακύρωση κράτησης

Από την στιγμή που έχει γίνει η κράτηση ενός δωματίου, για να ακυρωθεί αυτή θα πρέπει ο χρήστης να επικοινωνίσει με τον ξενοδόχο και αυτός με την σειρά του να μπεί στο σύστημα και να πάει στην ακύρβση κράτησης. Εκεί βρίσκει το εν λόγο δωμάτιο και πατάει το κουμπί ακύρωσης. Η διαδικασία αυτή χρησιμοποιεί την ajax ώστε ο διαχειρηστής να μην χρειαστεί να επαναφορτώσει την σελίδα. Η διαδικασία αυτή της ακύρωσης φαίνεται στον κώδικα 3.19.

```
$(document).on('click','.akirwsi',function(){
    var parent=$(this).parent().parent();
    var room_id=$(this).attr('id');
    var kratisi_id=parent.attr('id');
    var from_date=parent.find('.from_date').text();
    var to_date=parent.find('.to_date').text();
    $.ajax({
            url:"cancel_reservation_ajax.php",
            method: "POST",
            //dataType:"JSON",
            data:{
                cancel_res: "cancel",
                room_id: room_id,
                kratisi_id:kratisi_id,
                from_date:from_date,
                to_date:to_date
            },
            success:function(result){
                console.log(result);
                if(result=='ok'){
                    parent.remove();
                }else{
                    alert("Κάτι πήγε στραβά");
                }
            }
        });

});
```

Κώδικας 3.19 Ακύρωση κράτησης

3.3.7 Αρχείο δημιουργίας νέου ξενοδοχείου

Τέλος θα πρέπει να αναφερθεί ένα ακόμα βασικό κομμάτι κώδικα, το οποίο είναι να κάνει ένας ξενοδόχος δημιουργία του ξενοδοχείου του. Αυτό γίνεται μεσα από την σελίδα create_hotel.php. Ο ξενοδόχος, αφού συνδεθεί με τα στοιχεία του, μπορεί να πάει στην δημιουργία ξενοδοχείου. Εκεί επιλέγει τον νομό και στην συνέχεια την πόλη. Έπειτα του ανοίγονται κάποια πεδία για να βάλει τις πληροφορίες του ξενοδοχείου του. Τέλος παντώντας στην αποθήκευση τα δεδομένα αυτά δημιουργούνται. Ο λόγος που αναφέρεται η διαδικασία αυτή είναι ότι στην σελίδα αυτή, εκτώς από το να χρησιμοποιούνται απλές εντολές αναζήτησης στην βάση και εμφάνησης αποτελεσμάτων, δημιουργούνται φακέλοι για το ξενοδοχείο μέσω της εφαρμογής, όπου εκεί αποθηκεύονται οι φωτογραφίες του ξενοδοχείου. Το κομμάτι αυτό φαίνεται στον κώδικα 3.20.

```php
<?php
if(isset($_POST['insert'])){
    require_once "db_connect.php";
    $con=connect();
    $name=$_POST['name'];
    $stars=$_POST['stars'];
    $poli=$_POST['poli'];
    $description=$_POST['description2'];
    $creator=$_SESSION['id'];
    $paroxes=array();
    $images_to_del=array();
    $photo_folder_name=str_replace(" ","_",$name);
    mkdir("inc/img/uploaded_photos/".$photo_folder_name);
    foreach($_POST['paroxes_xenodoxeiou'] as $paroxi){
        $paroxes[]=$paroxi;
    }
    $telikes_paroxes=implode(",",$paroxes);
    if(isset($_POST['other_photos'])){
        foreach($_POST['other_photos'] as $photo){
            $images_to_del[]=$photo;
        }
    }
    $images_to_add=array();
    $files=array_diff(scandir("inc/img/uploaded_photos/".$photo_folder_name),
array('.', '..','main.jpg'));
    $max=0;
    foreach($files as $file) {
        $file = pathinfo($file);
        if($file['filename']>$max){
            $max=$file['filename'];
        }
    }
    $next_photo_num=$max++;
    $lat=$_POST['lat'];
    $lng=$_POST['lng'];
    if($_FILES['main_upload']['error'][0]==0){
        $main=$_FILES['main_upload']['name'];

if(!move_uploaded_file($_FILES['main_upload']['tmp_name'],"inc/img/uploaded_photos/
".$photo_folder_name."/main.jpg")){
            echo "main not uploaded";
        }
    }
    $error_upload=$_FILES['others_upload']['error'][0];
    if($error_upload == 0){
        foreach ($_FILES['others_upload']['name'] as $photo=>$name2){

move_uploaded_file($_FILES['others_upload']['tmp_name'][$photo],"inc/img/uploaded_p
hotos/".$photo_folder_name."/".$max.".jpg");
            $images_to_add[]=$max.".jpg";
            $max++;
        }
    }
    if(count($images_to_add)>0){
        $images_to_add_string=implode(",",$images_to_add);
```

```
            $other_photos=$images_to_add_string;
            $other_photos=$images_to_add_string;
    }
    $sql="insert into hotels
(name,stars,description,lat,lng,paroxes,other_photos,town,creator_id) VALUES
('$name','$stars','$description','$lat','$lng','$telikes_paroxes','$other_photos','
$poli','$creator')";
    $result=db_query($sql,$con);
    if($result){
        echo "Η εισαγωγή έγινε με επιτυχία";
    }else{
        echo "Κάτι πήγε στραβά";
    }
    mysqli_close($con);
}

?>
```

Κώδικας 3.20 Κώδικας αποθήκευσης και δημιουργίας ξενοδοχείου

4 Συμπεράσματα και μελλοντικές επεκτάσεις

Η παρούσα πτυχιακή εργασία πραγματεύτηκε το σχεδιασμό και την ανάπτυξη ενός συστήματος διαχείρησης και κράτησης δωματίων για ξενοδοχιακές μονάδες. Χωρίστηκε σε δύο φάσεις, την φάση της σχεδίασης της εφαρμογής και την φάση της υλοποίησης της.

Μέσα από την εφαρμογή δίνεται η δυνατότητα να γίνει κράτηση εύκολα και γρήγορα και πάνω από όλα σε ένα σύστημα φιλικό προς τον χρήστη. Το τελευταίο επιτυγχάνεται με την χρήσης της βιβλιοθήκης bootstrap η οποία κάνει την εφαρμογή responsive ώστε ο χρήστης να μπορεί να την χρησιμοποιήσει από όποια συσκευή θέλει, χωρίς να πρέπει να αναπτυχθεί διαφορετικό περιβάλλον για χρήση από κάποια συσκευή χειρός. Επομένως με την βιβλιοθήκη αυτή λύνεται το πρόβλημα που αναφέρθηκε.

Ένα άλλο πρόβλημα ήταν η εύρεση και η εισαγωγή των πόλεων και των χωριών της Ελλάδος στην εφαρμογή. Την λύση έδωσε μια ιστοσελίδα στο internet η yuv στην οποία μπορείς να επιλέξεις τον νομό και σου εμφανίζει τους ταχυδρομικούς κώδικες του νομού. Στην συνέχεια επιλέγοντας ένα τ.κ. σου εμφανίζει όλα τα χωριά και τις πόλεις που ανήκουν σε αυτόν (link) Η διαδικασία αυτή ήταν αρκετά χρονοβόρα, όμως ήταν και αναγκαία για την ευκολία του χρήστη.

Τέλος για την διευκόλυνση του χρήστη χρησιμοποιήθηκε η τεχνική της Ajax. Η τεχνική αυτή μας επιτρέπει την επικοινωνία της javascript με τον server, κάτι το οποίο είναι αδύνατο χωρίς αυτή. Ο λόγος που χρησιμοποιείται είναι ότι κατά την αποστολή ενός δεδομένου με χρήση μιας φόρμας σε μια σελίδα php, ο περιηγητής αναγκάζεται να μεταφερθεί στην σελίδα php που στάλθηκε. Με την χρήση της ajax δεν είναι υποχρεωτικό, μιας και τα αποτελέσματα της διαδικασίας του αρχείου μπορούν να επιστραφούν ως μεταβλητή στην javascript. Αυτό προϋποθέτει όμως τα σενάρια (scripts) στον περιηγητή να είναι ενεργοποιημένα. Για την περίπτωση που τα scripts είναι απενεργοποιημένα, έχουμε χρησιμοποιήσει την ετικέτα της html, noscript η οποία δρομολογεί τον χρήστη σε μία σελίδα με πληροφορίες σχετικά με το πώς να ενεργοποιήσει τα σενάρια στον περιηγητή του.

Ως μελλοντική επέκταση της εφαρμογής, θα μπορούσαμε να γράψουμε τις σελίδες αυτές χωρίς κάποιο javascript, ώστε οι χρήστες οι οποίοι θέλουν να έχουν απενεργοποιημένα τα σενάρια στον υπολογιστή τους, να μπορούν να χρησιμοποιήσουν την εφαρμογή χωρίς κάποιο πρόβλημα.

Βιβλιογραφία

[1] “Booking.com”, [Ιστοσελίδα], τοποθεσία: https://www.booking.com.

[2] “Agoda.com”, [Ιστοσελίδα], τοποθεσία: https://www.agoda.com.

[3] “Bookgreece.com”, [Ιστοσελίδα], τοποθεσία: https://www.bookgreece.com.

[4] Barry, Chris, and Michael Lang. "A survey of multimedia and web development techniques and methodology usage." IEEE MultiMedia 8.2 (2001): 52-60.

[5] Severance, Charles. "Javascript: Designing a language in 10 days." *Computer* 45.2 (2012): 7-8.

[6] Παναγιώτης Δ. Κεντερλής. ‘’Ανάπτυξη διαδικτυακών εφαρμογών – Θεωρία και Πράξη’’, Αθήνα (2009)

[7] R. Elmasri – S.B. Navathe, ‘’Θεμελιώδεις Αρχές Συστημάτων Βάσεων Δεδομένων’’, Τόμος Α’, 5η έκδοση αναθεωρημένη, μετάσταση – επιμέλεια: Μιχάλης Χατζόπουλος, Εκδόσεις Δίαυλος, 2007

[8] “PHP: Hypertext Preprocessor”, [Ιστοσελίδα], τοποθεσία: https://secure.php.net/

[9] “jQuery”, [Ιστοσελίδα], τοποθεσία: https://jquery.com/

[10] “W3Schools Online Web Tutorials”, [Ιστοσελίδα], τοποθεσία: http://www.w3schools.com/

[11] “StackOverflow”, [Ιστοσελίδα], τοποθεσία: http://stackoverflow.com/

[12] “Bootstrap · The world's most popular mobile-first and responsive front-end framework.”, [Ιστοσελίδα], τοποθεσία: http://getbootstrap.com/

[13] “Font Awesome, the iconic font and CSS toolkit”, [Ιστοσελίδα], τοποθεσία: http://fontawesome.io/

[14] “GitHub - PHPMailer/PHPMailer: The classic email sending library for PHP”, [Ιστοσελίδα], τοποθεσία: https://github.com/PHPMailer/PHPMailer

[15] Douligeris, C., Mitropoulos, S., Δουληγέρης, Χ., & Μητρόπουλος, Σ. (2015). Πληροφοριακά συστήματα στο διαδίκτυο.

[16] Mitakos, T., & Μητάκος, Θ. (2015). Management Information Systems

[17] “Xampp installers and Downloads for Apache Friends”, [Ιστοσελίδα], τοποθεσία: https://www.apachefriends.org.

[18] “Brackets - A modern, open source code editor that understands web design.”, [Ιστοσελίδα], τοποθεσία: http://brackets.io

[19] Ιωάννης μανωλόπουλος, Απόστολος Ν. Παπαδόπουλος, “Συστήματα Βάσεων Δεδομένων, Θεωρία και Πρακτική Εφαρμογή”,Αθήνα(2009)