ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΠΟΛΥΜΕΣΩΝ

Πτυχιακή Εργασία

ΘΕΜΑ

"Εφαρμογή Εσωτερικής Διαχείρησης Κρατήσεων σε Ξενοδοχεία"

Ακαδημαϊκό Έτος : 2006 - 2007

Σπουδαστές :

Σαγιαδινός Ευστάθιος
Καρούζος Ιωάννης

Επιβλέπων Καθηγητής :

Παχουλάκης Ιωάννης

Ιούνιος 2006

Πίνακας Περιεχομένων

Πίνακας Σχημάτων - Εικόνων 3

Ευχαριστίες 4

Κεφάλαιο 1 5

1. ΕΙΣΑΓΩΓΗ 5

Κεφάλαιο 2 7

2. Τεχνολογίες Υλοποίησης 7

2.1 Γιατί Visual Basic .NET; 7

2.1.1 Object – oriented προγραμματισμός 10

2.1.2 ADO .NET 14

2.1.3 XML 19

2.2 Βάσεις Δεδομένων και SQL Server 2000 (MSDE) 22

Κεφάλαιο 3 24

3. Εφαρμογή – Hotels Manager 24

3.1 Περιγραφή της Βάσης Δεδομένων 24

3.1.1 Πίνακες της Βάσης Δεδομένων 24

3.1.2 Διάγραμμα βάσης δεδομένων – Συσχετίσεις πινάκων 27

3.2 Αρχιτεκτονική Εφαρμογής 29

3.2.1 Φόρμες και αλληλεπίδραση μεταξύ τους και με τη βάση δεδομένων 29

3.2.1.1 Main Form 29

3.2.1.2 Booking Procedure 30

3.2.1.3 Add Customer Form 36

3.2.1.4 Edit Customer Form 38

3.2.1.5 Find and Delete Booking 38

3.2.1.6 View Day 39

3.2.1.7 Custom View 41

3.2.1.8 Graphical View 43

3.2.1.9 Arriving Today – Check In Procedure 45

3.2.1.10 Departing Today – Check Out Procedure 46

3.2.1.11 Rooms Maintainance (Rooms To Do) 47

3.2.1.12 Log Form 48

3.2.1.13 Calendar Form 49

Κεφάλαιο 4 50

4. Ανακεφαλαίωση και Μελλοντικές Προσθήκες Βελτιώσεις 50

Βιβλιογραφία – Πηγές (Reference) 52

Πίνακας Σχημάτων - Εικόνων

ΣΧΗΜΑ 1 - DATAADAPTERS AND DATASETS 18
ΣΧΗΜΑ 2 – ΕΠΙΚΟΙΝΩΝΙΑ SERVER ΜΕ ΧΡΗΣΤΕΣ (LAN – INTERNET) 22
ΣΧΗΜΑ 3 – ΣΥΝΟΛΟ ΦΟΡΜΩΝ ΚΑΙ ΑΛΛΗΛΕΠΙΔΡΑΣΗ ΜΕΤΑΞΥ ΤΟΥΣ 29
ΣΧΗΜΑ 4 – ΔΟΜΗ ΤΟΥ ΑΡΧΕΙΟΥ GRAPHICALDATA.XML ΣΕ ΓΡΑΦΙΚΗ ΜΟΡΦΗ 44

ΕΙΚΟΝΑ 1 - ΠΙΝΑΚΑΣ CUSTOMERS 24
ΕΙΚΟΝΑ 2 - ΠΙΝΑΚΑΣ FACILITIES 25
ΕΙΚΟΝΑ 3 - ΠΙΝΑΚΑΣ HOTELFACILITIES 25
ΕΙΚΟΝΑ 4 - ΠΙΝΑΚΑΣ ROOMFACILITIES 25
ΕΙΚΟΝΑ 5 - ΠΙΝΑΚΑΣ HOTELS 26
ΕΙΚΟΝΑ 6 - ΠΙΝΑΚΑΣ ROOMS 26
ΕΙΚΟΝΑ 7 - ΠΙΝΑΚΑΣ RESERVATION 27
ΕΙΚΟΝΑ 8 - ΔΙΑΓΡΑΜΜΑ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ 28
ΕΙΚΟΝΑ 9 – ΚΥΡΙΩΣ ΦΟΡΜΑ (MAIN FORM) 30
ΕΙΚΟΝΑ 10 – ΞΕΚΙΝΑΜΕ ΤΗ ΔΙΑΔΙΚΑΣΙΑ ΤΟΥ BOOKING 31
ΕΙΚΟΝΑ 11 – BOOKFORM1 (SELECT DATES AND FACILITIES) 32
ΕΙΚΟΝΑ 12 – ΕΠΙΛΟΓΗ ΔΩΜΑΤΙΟΥ ΑΠΟ DATAGRID 33
ΕΙΚΟΝΑ 13 - GRAPHICAL VIEW 33
ΕΙΚΟΝΑ 14 – ΦΟΡΜΑ SELECT CUSTOMER 34
ΕΙΚΟΝΑ 15 – ΦΟΡΜΑ BOOK CONFIRM 35
ΕΙΚΟΝΑ 16 – FINAL BOOK FORM 36
ΕΙΚΟΝΑ 17 – ADD CUSTOMER FORM (NEW CUSTOMER) 37
ΕΙΚΟΝΑ 18 – ΑΝΟΙΓΟΥΜΕ ΤΗ FIND BOOKING FORM 38
ΕΙΚΟΝΑ 19 – FIND BOOKINGS FORM 38
ΕΙΚΟΝΑ 20 – ΕΠΙΛΕΓΟΥΜΕ ΑΠΟ ROOMS ΤΟ TODAY 39
ΕΙΚΟΝΑ 21 – VIEW DAY FORM 39
ΕΙΚΟΝΑ 22 – ΕΠΙΛΕΓΟΥΜΕ ΑΠΟ ROOMS ΤΟ CUSTOM DATE 41
ΕΙΚΟΝΑ 23 – CUSTOM VIEW FORM 1 41
ΕΙΚΟΝΑ 24 – CUSTOM VIEW FORM 2 42
ΕΙΚΟΝΑ 25 – ARRIVING TODAY ΓΙΑ CHECK IN 46
ΕΙΚΟΝΑ 26 – ΕΠΙΒΕΒΑΙΩΣΗ ΓΙΑ ΤΟ CHECK IN 46
ΕΙΚΟΝΑ 27 – DEPARTING TODAY ΓΙΑ CHECK OUT 46
ΕΙΚΟΝΑ 28 – ΕΠΙΛΕΓΟΥΜΕ ΑΠΟ ROOMS ΤΟ ROOMS MAINTAINANCE 47
ΕΙΚΟΝΑ 29 – ROOMSTODO FORM (ROOMS MAINTAINANCE) 47
ΕΙΚΟΝΑ 30 – LOG FORM 48
ΕΙΚΟΝΑ 31 – TIME CALENDAR FORM 49

Ευχαριστίες

Ευχαριστούμε ιδιαίτερα τους γονείς μας και τα αδέρφια μας για την οποιασδήποτε μορφής υποστήριξη μας παρείχαν όλα αυτά τα χρόνια της φοίτησής μας και για την υπομονή που επέδειξαν. Ευχαριστούμε θερμά όσους συνέβαλαν στη επίτευξη της παρούσας πτυχιακής εργασίας με την υλική ή ψυχολογική τους βοήθεια, από τους οποίους δε θα μπορούσαν να αποτελέσουν εξαίρεση οι φίλοι και το οικογενειακό μας περιβάλλον.

Ιδιαίτερες ευχαριστίες στον καθηγητή – εισηγητή μας Κύριο Παχουλάκη Ιωάννη για την καθοδήγηση, βοήθεια και υποστήριξή του καθώς και για την υπομονή του σχετικά με το χρόνο που χρειάστηκε προκειμένου να φέρουμε επιτυχώς εις πέρας την υλοποίηση της πτυχιακής εργασίας μας.

Τέλος θα θέλαμε να ευχαριστήσουμε το Τ.Ε.Ι. Ηρακλείου που μας φιλοξένησε όλα αυτά τα χρόνια σπουδών μας και συγκεκριμένα τη σχολή «Εφαρμοσμένης Πληροφορικής και Πολυμέσων», από όπου μας δόθηκε η δυνατότητα να αντλήσουμε απαραίτητες γνώσεις, οι οποίες θα μας φανούν χρήσιμες στην επαγγελματική μας πορεία.

Κεφάλαιο 1

1. ΕΙΣΑΓΩΓΗ

Το αντικείμενο της πτυχιακής εργασίας είναι ο σχεδιασμός και η υλοποίηση μιας εφαρμογής αυτοματοποίησης των εσωτερικών διεργασιών κρατήσεων και πληρωμής σε ένα ξενοδοχείο από το προσωπικό του ξενοδοχείου και θα αναπτύσσεται σε συνέχεια της πτυχιακής με τίτλο «Εφαρμογή Υποστήριξης Κρατήσεων σε Ξενοδοχεία από πελάτες μέσω Διαδικτύου» (Σεπτέμβρης 2004).

Η εφαρμογή υλοποιήθηκε σε τεχνολογία .NET, με βάση δεδομένων σε SQL Server (MSDE) και επιτρέπει την πρόσβαση προσωπικού του ξενοδοχείου για την εσωτερική πια διαχείριση δωματίων, λοιπών χώρων καθώς και δεδομένων των κρατήσεων των πελατών.

Το μοντέλο που χρησιμοποιείται είναι Client-Server. Ένας Server κρατά τη βάση δεδομένων και επιτρέπει σε πολλούς χρήστες (clients) την πρόσβαση για ανάγνωση, μετατροπή ή εγγραφή δεδομένων. Έτσι, μπορούμε στην ίδια βάση να συνδέουμε διάφορους χρήστες απο διαφορετικά σημεία του ξενοδοχείου όπως η Reception, ο διευθυντής και λοιποί. Παράλληλα η ίδια βάση εξυπηρετεί και το online σύστημα. Οπότε αν γίνει κράτηση online, η πληροφορία είναι διαθέσιμη σε πραγματικό χρόνο στους χρήστες της εφαρμογής μας.

Έχει μεταξύ άλλων τις εξής δυνατότητες:

- Πραγματοποίηση κρατήσεων για οποιοδήποτε χρονικό διάστημα και με διαφορετικούς τρόπους.
- Μηχανή εύρεσης δωματίων βάση διαθεσιμότητας και facilities
- Δυνατότητα εύρεσης, προσθήκης και ανανέωσης στοιχείων πελατών
- Πλήρες Log για τις κρατήσεις (πραγματοποίηση ή διαγραφή) με αυτόματη προβολή του Log της ημέρας
- Πρόνοια για χειροκίνητη λειτουργία Check-in και Check-out για να υπάρχει έλεγχος

- Ευκολη πρόσβαση στις πληροφορίες των τρέχουσων (σημερινών) check-ins και check-outs
- Δυνατότητα προσθήκης to-do/Notes/Εκκρεμοτήτων για τα δωμάτια και λειτουργία σχετικής πληροφόρησης απο διάφορα σημεία του προγράμματος.
- Client – Server αρχιτεκτονική
- Ψευδοτρισδιάσταση απεικόνιση των δωματίων
- Γραφικές φόρμες για την πληροφόρηση της κατάστασης των δωματίων και άλλων χαρακτηριστικών που σχετίζονται με αυτά για την τρέχουσα ημέρα ή για οποιοδήποτε χρονικό διάστημα.

Σε αυτή την εργασία θα γίνει μια παρουσιαση των τεχνολογιών που χρησιμοποιήθηκαν, όπως η Visual Basic .Net, ο SQL Server, η XML κλπ. Επίσης θα περιγραφεί η βάση δεδομένων που χρησιμοποιήθηκε και θα εξηγηθεί η λειτουργία της εφαρμογής με παράθεση αλλά και επεξήγηση των διαφόρων φορμών, της αρχιτεκτονικής και της φιλοσοφίας πίσω απο τον κώδικα.

Κεφάλαιο 2

2. Τεχνολογίες Υλοποίησης

2.1 Γιατί Visual Basic .NET;

Η Visual Basic .NET παρέχει χαρακτηριστικά που είναι πολύ σημαντικά για τους προγραμματιστές, όπως αντικειμενοστραφή προγραμματισμό, ακολουθίες χαρακτήρων, γραφικά, συστατικά γραφικού περιβάλλοντος χρήστη (GUI), χειρισμό εξαιρέσεων, πολυνηματισμό, πολυμέσα (ήχο, εικόνες, video), επεξεργασία αρχείων προκαθορισμένες δομές δεδομένων, επεξεργασία βάσεων δεδομένων, δικτύωση βασισμένη στο Internet, client/server και κατανεμημένη πληροφορική. [Ref. 2]

Η Visual Basic ακολούθησε εντυπωσιακή εξέλιξη την τελευταία δεκαετία, από την έκδοση 1.0 στη σημερινή έκδοση .NET ή 7.0, όπως είναι η εσωτερική αρίθμηση του προϊόντος. Οι νέες εντολές, οι βελτιώσεις στον τρόπο ανίχνευσης των λαθών αλλά και όσα εργαλεία πρστέθηκαν μέχρι τη σημερινή έκδοση, δεν άλλαξαν τον αντικειμενικό σκοπό, που είναι η εύκολη και γρήγορη ανάπτυξη εφαρμογών.

Έτσι, η Visual Basic, μέσα από το Visual Studio, παραμένει ένα εύκολο και ιδιαίτερα γρήγορο εργαλείο για την ανάπτυξη αυτόνομων εφαρμογών αλλά και υπηρεσιών στο Web βασιζόμενες στο .NET framework. [Ref. 1]

Το .NET πλαίσιο (framework) εισάγει ένα απολύτως νέο πρότυπο για τον προγραμματισμό και την επέκταση των εφαρμογών. Η .NET είναι ένα όραμα της Microsoft «λογισμικό ως υπηρεσία», ένα περιβάλλον ανάπτυξης στο οποίο μπορούμε να χτίσουμε, να δημιουργήσουμε, και να επεκτείνουμε τις εφαρμογές μας και την επόμενη γενεά components.

Ωστόσο, η νέα έκδοση Visual Basic .NET φέρνει αρκετές αλλαγές στην ίδια τη γλώσσα, στο περιβάλλον προγραμματισμού και κατ΄ επέκταση στη μεταφορά μιας υπάρχουσας εφαρμογής. Οι βελτιώσεις στη γλώσσα περιλαμβάνουν νέες εντολές και σχήματα που φέρνουν πιο κοντά τη VB στις υπόλοιπες αντικειμενοστραφείς γλώσσες, όπως η C++ και η Java. Από τις πιο σημαντικές βελτιώσεις σε αυτό τον τομέα είναι η δυνατότητα της κληρονομικότητας (inheritance), της δημιουργίας κλάσεων (class) αλλά και της ενσωμάτωσης πιο αυστηρών κανόνων για την

αντίδραση σε λάθη (error trapping). Ο προγραμματιστής έχει στη διάθεσή του την εντολή "inherits", που επιτρέπει τη μεταφορά όλων των ιδιοτήτων, των μεθόδων και των events ενός αντικειμένου ή μιας κλάσης. Υπάρχει επίσης και το function overloading, που προσφέρει τη δυνατότητα να χρησιμοποιούμε τις ίδιες λειτουργίες με διαφορετικές παραμέτρους. [Ref. 1]

Το μεγαλύτερο μέρος του Win32 API είναι τώρα προσιτό μέσω ενός πολύ απλού προτύπου αντικειμένου. Τα περισσότερα από τα χαρακτηριστικά γνωρίσματα και τις λειτουργίες C++ προστέθηκαν στη Visual Basic.

Ο κώδικας που γράφεται η .NET δεν συντάσσεται άμεσα στον εκτελέσιμο, αντ' αυτού η .NET χρησιμοποιεί δύο βήματα για να συντάξει τον κώδικα. Κατ' αρχάς, ο κώδικας μεταφράζεται σε μια ενδιάμεση γλώσσα αποκαλούμενη Microsoft Intermediate Language (MSIL). Δεύτερον, ο μεταφρασμένος κώδικας θα επαναμεταφραστεί με τη Common Language Runtime (CLR), η οποίος μετατρέπει τον κώδικα σε κώδικα μηχανής. Η βασική ιδέα αυτών των δύο στάδιων ήταν να γίνει η κωδικοποιημένη γλώσσα ανεξάρτητη από τη πλατφόρμα (όπως συμβαίνει με τη Java).

Η CLR είναι ο πυρήνας του .NET Framework. Όλος ο κώδικας που γράφεται στις .NET γλώσσες εκτελείται μέσω των CLR. Η CLR είναι παρόμοια με προηγούμενους χρόνους εκτέλεσης όπως ο Visual Basic runtime. Ο Visual Basic κώδικας εκτελείται μέσω του Visual Basic runtime, ο οποίος μεταφράζει τη VB γλώσσα στις χαμηλού επιπέδου κλήσεις Windows API.

Η CLR είναι ένα πιό ενεργό στοιχείο των εφαρμογών από ότι είναι ο VB runtime. Στην πραγματικότητα, η CLR παίρνει έναν τέτοιο ενεργό ρόλο στην εκτέλεση του κώδικα, έτσι ο κώδικας που γράφεται για τη CLR αναφέρεται ως διαχειρίζον κώδικας. Αυτό συμβαίνει επειδή, εκτός από την εκτέλεση του κώδικα, η CLR παρέχει υπηρεσίες. Παραδείγματος χάριν, η CLR φροντίζει όλη τη διαχείριση μνήμης και τη συλλογή άχρηστων στοιχείων (που επαναχρησιμοποιούν τη μνήμη που καταλαμβάνεται από τα αντικείμενα που δεν είναι πλέον σε λειτουργία) για τις .NET εφαρμογές. [Ref. 8]

Οι εντολές On Error και Resume Next έχουν αντικατασταθεί με πιο σταθερές και συνεπείς λύσεις, όπως οι εντολές Try, Catch και Finally, επιτρέποντας τη συγγραφή πιο καθαρού κώδικα.

Ένα ακόμα ενδιαφέρον χαρακτηριστικό στις φόρμες της Visual Basic .NET είναι η μεγαλύτερη ακρίβεια. Χαρακτηριστικά όπως το Menu Designer, Control Anchoring,

Control Docking και μια σειρά από νέα στοιχεία ελέγχου επιτρέπουν να αναπτύξει κανείς πιο καλαίσθητες φόρμες. Σε αυτό έρχεται να προστεθεί και η υποστήριξη του GDI+, του διαδόχου του GDI (Graphic Device Interface) που συναντούσαμε στις προηγούμενες εκδόσεις των Windows, το οποίο προσφέρει πιο πλούσια χαρακτηριστικά. Έτσι, μπορεί κανείς εύκολα να σχεδιάσει οθόνες που εμφανίζονται σε διαφανή και πολυεπίπεδα παράθυρα, χαρακτηριστικό που επιτρέπουν τα Windows 2000 και τα Windows XP. [Ref. 1]

Άλλο ένα ενδιαφέρον στοιχείο της Visual Basic .NET είναι η πλήρης υποστήριξη του Unicode, ακόμα και στις φόρμες.

Με την ολοκλήρωση εντολών IntelliSense και τον αυτόματο έλεγχο σφαλμάτων σύνταξης, η Visual Basic .NET ενημερώνει τους προγραμματιστές όταν είναι λανθασμένος ο κώδικας και παρέχει άμεση εμβάθυνση σε ιεραρχίες κλάσεων και API. Χρησιμοποιώντας την Εξερεύνηση λύσεων (Solution Explorer), οι προγραμματιστές μπορούν εύκολα να επαναχρησιμοποιήσουν έναν κώδικα σε διάφορα έργα, ακόμη και να δημιουργήσουν λύσεις σε πολλές γλώσσες, οι οποίες ανταποκρίνονται πιο αποτελεσματικά στις ανάγκες της επιχείρησής τους. Και, χάρη στο πλήρως επεκτάσιμο IDE, οι προγραμματιστές μπορούν να απολαμβάνουν τα οφέλη ενός εντυπωσιακού συμπληρωματικού προγράμματος άλλου κατασκευαστή και μιας κοινότητας προμηθευτών στοιχείων (components) που τους βοηθά να προσαρμόσουν και να επεκτείνουν περαιτέρω το περιβάλλον ώστε να ανταποκρίνεται στις ανάγκες τους.

Με τους οδηγούς εφαρμογών, τα πρότυπα έργων και το δείγμα πηγαίου κώδικα, οι προγραμματιστές μπορούν να δημιουργούν γρήγορα εφαρμογές Windows, Web και συσκευών με ελάχιστη αρχική επένδυση. Η δυναμική Βοήθεια και το Microsoft Developer Network (MSDN) παρέχουν βοήθεια η οποία βασίζεται στην τρέχουσα εργασία και γλώσσα προγραμματισμού, διασφαλίζοντας ότι οι προγραμματιστές θα έχουν πάντα στη διάθεσή τους πληροφορίες για την πλατφόρμα .NET ή τη γλώσσα της επιλογής τους.

2.1.1 Object – oriented προγραμματισμός

Γενικές Αρχές

Το **Object Model** που χρησιμοποιείται στη VB είναι παρόμοιο με άλλες αντικειμενοστρεφείς γλώσσες (Object Oriented Languages). Η αντικειμενοστρέφεια είναι νεώτερη από το Δομημένο Προγραμματισμό, αλλά είναι πλέον πολύ σημαντική για τους προγραμματιστές σήμερα. Ο Object Oriented προγραμματισμός παρέχει μια τυποποιημένα γλώσσα και ένα πλαίσιο που είναι σημαντικό για να κάνει την εργασία τμημάτων λογισμικού (objects) ενωμένα μαζί σε ένα σύστημα. Οι Object Oriented έννοιες που μαθαίνονται για τη VB ισχύουν για τις περισσότερες γλώσσες που ένας προγραμματιστής είναι πιθανό να αντιμετωπίσει σήμερα. [Ref. 7,10]

Μια γλώσσα βασισμένη σε αντικείμενα έχει τις περισσότερες από τις ιδιότητες μιας αντικειμενοστρεφούς γλώσσας, αλλά μπορεί να στερείται μερικών. Παραδείγματος χάριν η Visual Basic δεν έχει κληρονομικότητα, ενώ μια γλώσσα βασισμένη σε πρωτότυπα στηρίζεται σε αυτά αντί στις κλάσεις για να δημιουργήσει τα αντικείμενα.

Η αντικειμενοστρέφεια (Object Orientation) έχει αλλάξει τον τρόπο που οι βιομηχανίες λογισμικού προσεγγίζουν την ανάπτυξη λογισμικού, και ιδιαίτερα, πώς οι τεχνικοί προσεγγίζουν την ανάλυση, το σχεδιασμό, την επαναχρησιμοποίηση κώδικα, και την υλοποίηση. [Ref. 12]

Ο Object Oriented προγραμματισμός είναι μια προσέγγιση (όχι ένα συγκεκριμένο εργαλείο) που οργανώνεται γύρω από τα αντικείμενα παρά από τις εφαρμογές, τα δεδομένα ή τη λογική.

Objects (Αντικείμενα)

Ο Object Oriented προγραμματισμός οργανώνεται γύρω από τα αντικείμενα, το interface τους, το πώς τα αντικείμενα επικοινωνούν το ένα με το άλλο, την κατάστασή τους σε ένα δεδομένο instance κ.λπ. Βλέποντας την ανάλυση και το σχεδιασμό του λογισμικού ως αντικείμενα είναι πολύ σημαντικό επειδή και οι άνθρωποι σκέφτονται σε σχέση με τα αντικείμενα. Όλα γύρω μας είναι αντικείμενα. Για παράδειγμα, σκεφτόμαστε ένα αυτοκίνητο, ένα πορτοφόλι, ένα τραπέζι και ακόμη τους ανθρώπους ως αντικείμενα. [Ref. 12]

Πολλά αντικείμενα αποτελούνται από άλλα αντικείμενα. Ένα πακέτο από έξι κουτάκια αναψυκτικό αποτελείται από έξι δοχεία αναψυκτικό και αποτελούν ένα

τύπο εμπορευματοκιβωτίου. Ένα αυτοκίνητο αποτελείται από μηχανή, τέσσερις ρόδες, τις πόρτες, κ.λπ. Ένα αντικείμενο είναι ένα instance μιας κλάσης. [Ref. 12]

Classes (Κλάσεις)

Μια κλάση είναι ένα σύνολο παρόμοιων ενεργειών και ιδιοτήτων, και ένα αντικείμενο είναι ένα instance (μια περίπτωση) μιας κλάσης. Μπορούμε να σκεφτούμε μια κλάση ως σχεδιασμό ενός αντικειμένου που θα δημιουργηθεί στον κώδικά μας. Και οι κλάσεις και τα αντικείμενα έχουν ταυτότητα. Μια κλάση καθορίζει τις ιδιότητες και τις μεθόδους, ενώ ένα αντικείμενο τις χρησιμοποιεί.

Μια κλάση καθορίζει τα στοιχεία των δεδομένων και τη μορφή των δεδομένων που συνδυάζονται για να απεικονίσουν το τι ένα αντικείμενο «γνωρίζει». Παραδείγματος χάριν, μπορούμε να δημιουργήσουμε ένα αντικείμενο αποκαλούμενο 'Oscar' από μια κλάση που ονομάζεται Dog. Η κλάση Dog καθορίζει τι πρόκειται να είναι το αντικείμενο 'Oscar', και όλα τα «dog-related» μηνύματα που ένα αντικείμενο Oscar μπορεί να ενεργήσει επάνω τους. Όλες οι αντικειμενοστραφείς γλώσσες έχουν κάποιες έννοιες. Καλούν μερικές φορές ένα «εργοστάσιο», να κατασκευάσει τα instances ενός αντικειμένου από τον καθορισμό μιας κλάσης. Μπορούμε να κάνουμε περισσότερα από ένα αντικείμενα αυτής της κλάση, και να τα καλέσουμε Spot, Fido, Rover, κ.λπ. Η κλάση Dog καθορίζει τα μηνύματα που τα αντικείμενα Dog καταλαβαίνουν, όπως «bark», «fetch», και «roll-over».

Οι κλάσεις και τα αντικείμενα έχουν χαρακτηριστικά. Παραδείγματα χαρακτηριστικών τους για ένα άτομο είναι η ηλικία, το ύψος, το βάρος, και δείκτης νοημοσύνης. Αυτά τα χαρακτηριστικά αναφέρονται ως χαρακτηριστικά κλάσης και ιδιότητες αντικειμένου. Τα χαρακτηριστικά της κλάσης και οι ιδιότητες του αντικειμένου είναι επίσης γνωστές ως member fields. Επειδή ένα αντικείμενο έχει τη δυνατότητα να θέσει τις ιδιότητες, τα αντικείμενα λέγονται επίσης ότι έχουν την κυριότητα. Οι τρέχουσες τιμές των χαρακτηριστικών ενός αντικειμένου είναι η επικρατούσα κατάστασή τους. [Ref. 10,12]

Properties (Ιδιότητες)

Τα **Properties** (ιδιότητες) λένε για το αντικείμενο, πληροφορίες όπως είναι το όνομά του, η θέση του στον υπολογιστή, εάν είναι ορατό, ενεργό, ή το χρώμα του. Οι ιδιότητες είναι όπως τα επίθετα που περιγράφουν τα αντικείμενα (ουσιαστικά). Οι

ιδιότητες στην VB.net δηλώνονται με τη χρησιμοποίηση του ονόματος αντικειμένου, μιας περιόδου (.), και την επιθυμητή ιδιότητα. Η VB δήλωση cmdPushMe.Visible = False θα καθιστούσε το cmdPushMe αόρατο σε μια φόρμα. Η σημαντικότερη ιδιότητα κάθε αντικειμένου είναι γενικά το όνομά της. Το όνομα είναι αυτό που δένει το αντικείμενο σε ένα script και το πώς ένα αντικείμενο δηλώνεται από οποιοδήποτε άλλο αντικείμενο. Η κωδικοποίηση της κλάσης πρέπει να λαμβάνεται υπόψιν κατά την ονομασία των αντικειμένων. [Ref. 11]

Methods (Μέθοδοι)

Οι κλάσεις και τα αντικείμενα έχουν ενέργειες. Αυτές οι ενέργειες αναφέρονται ως λειτουργίες κλάσης και **μέθοδοι** αντικειμένου. Τις αναφέρουμε γενικά ως μεθόδους (**methods**). Μέθοδος είναι το πώς ο κώδικας μπορεί να χρησιμοποιήσει ένα αντικείμενο κάποιας κλάσης. Οι μέθοδοι μπορούν να διαιρεθούν σε queries (ερωτήσεις) επιστρέφοντας την επικρατούσα κατάσταση και σε commands (εντολές) που την αλλάζουν (υπορουτίνα). Μερικές φορές η πρόσβαση στα δεδομένα ενός αντικειμένου είναι περιορισμένη στις μεθόδους της κλάσης του.

Μια μέθοδος είναι ουσιαστικά η εφαρμογή μιας υπηρεσίας αντικειμένου η οποία είναι απλά η δράση που ένα μήνυμα μεταφέρει. Είναι ο κώδικας, ο οποίος εκτελείται όταν το μήνυμα στέλνεται σε ένα συγκεκριμένο αντικείμενο. Ορίσματα παρέχονται συχνά ως τμήμα ενός μηνύματος. Στον Object Oriented προγραμματισμό στέλνουμε ένα μήνυμα από ένα αντικείμενο σε ένα άλλο. Οι μέθοδοι και οι ιδιότητες είναι κλήσεις μηνυμάτων. Οι παράμετροι μιας μεθόδου είναι το περιεχόμενο των μηνυμάτων. [Ref. 10]

Events (Γεγονότα)

Τα γεγονότα (**Events**) είναι το πως τα αντικείμενα πρέπει να ανταποκρίνονται. Μπορούν να είναι ερεθίσματα για μεθόδους αντικειμένων και αναφέρονται ως Object.Events. Παραδείγματος χάριν cmdPushMe.Clicked αναφέρεται στο γεγονός όταν πατάει ο χρήστης το command button που ονομάζεται cmdPushMe. [Ref. 11]

Inheritance (Κληρονομικότητα)

Η αρχή μιας κλάσης καθιστά πιθανό να καθοριστούν οι υποκλάσεις που μοιράζουν μερικά ή όλα τα κύρια χαρακτηριστικά της κλάσης. Αυτό καλείται κληρονομικότητα

(**inheritance**). Η κληρονομικότητα μας επιτρέπει επίσης να επαναχρησιμοποιήσουμε τον κώδικα αποτελεσματικότερα. ένας μηχανισμός για τις υποκλάσεις και παρέχει έναν τρόπο να καθοριστεί μια υποκλάση ως εξειδίκευση ή υποκατηγορία ή επέκταση μιας γενικότερης κλάσης. Μια υποκλάση κληρονομεί όλα τα μέλη (members) των superclass (υπερκλάσεών) της, αλλά μπορεί να επεκτείνει τη «συμπεριφορά» τους και να προσθέσει τα νέα μέλη. [Ref. 10]

Encapsulation

Το **encapsulation** (ενθυλάκωση) είναι το κρύψιμο των στοιχείων και του κώδικα και καλείται συχνά ως «black box» προσέγγιση, δεδομένου ότι οι χρήστες μιας κλάσης δεν μπορούν να δουν μέσα στην κλάση (μπορούν να δουν μόνο το public interface της). Εξασφαλίζει επίσης ότι ο κώδικας έξωτερικά από μια κλάση βλέπει μόνο τις λειτουργικές λεπτομέρειες εκείνης της κλάσης, αλλά όχι τις λεπτομέρειες της υλοποίησης. Το Encapsulation επιτυγχάνεται διευκρινίζοντας ποιές κλάσεις μπορούν να χρησιμοποιήσουν τα μέλη ενός αντικειμένου. Το αποτέλεσμα είναι ότι κάθε αντικείμενο εκθέτει σε οποιαδήποτε κλάση ένα συγκεκριμένο interface. Τα μέλη καθορίζονται συχνά ως public, protected και private. Αυτό καθορίζεται από το εάν είναι διαθέσιμα σε όλες τις κλάσεις, τις υποκλάσεις ή μόνο την καθορισμένη κλάση. Μερικές γλώσσες επεκτείνονται περαιτέρω: Η Java χρησιμοποιεί protected λέξη κλειδί για να περιορίσει την πρόσβαση. Η C# και η VB.NET υποκαθιστούν μερικά μέλη σε κλάσεις χρησιμοποιώντας τις λέξεις κλειδιά ως internal (C#) ή ως Friend (VB.NET). Τα αντικείμενα αλληλεπιδρούν το ένα με το άλλο μέσω μηνυμάτων. Το μόνο πράγμα που ένα αντικείμενο γνωρίζει για ένα άλλο είναι το interface του αντικειμένου. Τα δεδομένα και η λογική κάθε αντικειμένου είναι κρυμμένα από άλλα αντικείμενα. Αυτό επιτρέπει στον αναλυτή να χωρίσει την υλοποίηση ενός αντικειμένου από το interface του. Εφ' όσον το interface παραμένει ίδιο, οποιεσδήποτε αλλαγές στην εσωτερική υλοποίηση είναι προφανείς στο χρήστη. [Ref. 10,12]

Polymorphism (Πολυμορφισμός)

Ένα άλλο όφελος που προκύπτει από το διαχωρισμό της υλοποίησης από τη συμπεριφορά είναι ο πολυμορφισμός (**polymorphism**). Ο πολυμορφισμός είναι συμπεριφορά που ποικίλλει ανάλογα με την κλάση στην οποία η συμπεριφορά

επιδρά, δηλαδή δύο ή περισσότερες κλάσεις μπορούν να αντιδράσουν διαφορετικά στο ίδιο μήνυμα. Ο πολυμορφισμός επιτρέπει σε δύο ή περισσότερα αντικείμενα για να αποκριθούν στο ίδιο μήνυμα. Με άλλα λόγια, ο πολυμορφισμός επιτρέπει σε οποιαδήποτε απόγονη κλάση για να επαναπροσδιορίσει οποιαδήποτε μέθοδο που κληρονομείται από την κλάση γονέων του. Η επίδραση είναι ότι ο πολυμορφισμός επιτρέπει σε ένα σταλμένο αντικείμενο να επικοινωνήσει με τα διαφορετικά αντικείμενα κατά σύμφωνο τρόπο χωρίς ανησυχία για το πόσες διαφορετικές υλοποιήσεις ενός μηνύματος υπάρχουν. [Ref. 10,12]

2.1.2 ADO .NET

Η ADO.NET είναι μια εξέλιξη του ADO μοντέλου πρόσβασης δεδομένων, που εξετάζει άμεσα τις απαιτήσεις χρηστών για τις εξελικτικές εφαρμογές. Η ADO.NET χρησιμοποιεί μερικά ADO αντικείμενα, όπως τα αντικείμενα σύνδεσης (**Connection Objects**) και εντολών (**Command Objects**), και εισάγει επίσης νέα αντικείμενα. Τα βασικά νέα αντικείμενα ADO.NET περιλαμβάνουν το **DataSet**, το **DataReader**, και το **DataAdapter**. [Ref. 6]

Η σημαντική διάκριση μεταξύ αυτού του εξελιγμένου σταδίου της ADO.NET και των προηγούμενων αρχιτεκτονικών στοιχείων είναι ότι υπάρχει ένα αντικείμενο -- το **DataSet** -- το οποίο είναι ξεχωριστό και ευδιάκριτο από οποιοδήποτε αποθηκευμένο δεδομένο. Λόγω αυτού, το **DataSet** λειτουργεί ως αυτόνομη οντότητα. Μπορείτε να σκεφτείτε το **DataSet** ως ένα μόνιμα αποσυνδεδεμένο recordset που δεν ξέρει τίποτα για την πηγή ή τον προορισμό των στοιχείων που περιέχει. Μέσα σε ένα **DataSet**, όπως σε μια βάση δεδομένων, υπάρχουν πίνακες, στήλες, σχέσεις, περιορισμοί, απόψεις, και ούτω καθ'εξής. [Ref. 6]

Ένα **DataAdapter** είναι το αντικείμενο που συνδέεται με τη βάση δεδομένων για να γεμίσει το **DataSet**. Κατόπιν, συνδέεται πίσω στη βάση δεδομένων για να ενημερώσει τα δεδομένα εκεί, βασισμένα στις διαδικασίες που πραγματοποιήθηκαν ενώ το **DataSet** κρατούσε τα δεδομένα. Στο παρελθόν, η επεξεργασία δεδομένων βασιζόταν κυρίως στη σύνδεση (connection). Τώρα, σε μια προσπάθεια να γίνουν οι

πολυ-στρωματικές εφαρμογές αποδοτικότερες, η επεξεργασία δεδομένων μετατρέπεται σε μια προσέγγιση που βασίζεται στα μηνύματα τα οποία κυρίως περιστρέφονται γύρω από τα σημαντικότερα κομμάτια της πληροφορίας. Στο κέντρο αυτής της προσέγγισης βρίσκεται το **DataAdapter**, το οποίο παρέχει μια γέφυρα για να ανακτηθούν και να αποθηκευθούν τα δεδομένα μεταξύ ενός **DataSet** και του πηγής των αποθηκευμένων δεδομένων (Database). Αυτό το ολοκληρώνει με τη βοήθεια αιτημάτων σε κατάλληλες εντολές SQL που γίνονται στα αποθηκευμένα δεδομένα. [Ref. 6]

Το **DataSet** αντικείμενο βασισμένο στην XML εξασφαλίζει ένα συνεπές πρότυπο προγραμματισμού που λειτουργεί με όλα τα πρότυπα αποθήκευσης δεδομένων: επίπεδο, σχεσιακό και ιεραρχικό. Το κάνει αυτό μη έχοντας καμία «γνώση» της πηγής των δεδομένων του. Χωρίς καμία σημασία ποιά είναι η πηγή των δεδομένων στο **DataSet**, χειρίζεται μέσω του ίδιου συνόλου του τυποποιημένου APIs που εκτίθεται μέσω του **DataSet** και των δευτερευόντων αντικειμένων του.

Ενώ το **DataSet** δεν έχει καμία γνώση της πηγής δεδομένων του, ο διαχειρίζον provider (managed provider) έχει λεπτομερείς και συγκεκριμένες πληροφορίες. Ο ρόλος του διαχειρίζον provider είναι να συνδέσει, να γεμίσει, και να διατηρήσει το **DataSet** προς και από τις «αποθήκες» δεδομένων. Οι OLE DB και SQL Server .NET Data Providers (System.Data.OleDb και System.Data.SqlClient) που είναι μέρος του .Net Framework παρέχουν τέσσερα βασικά αντικείμενα: την Εντολή (**Command**), την Σύνδεση (**Connection**), τον **DataReader** και τον **DataAdapter**. [Ref. 6]

Παρακάτω βλέπουμε μερικά αντικείμενα που έχουν εξελιχθεί, και μερικά νέα. Αυτά τα αντικείμενα είναι:

- **Connections** - Για τη σύνδεση και τη διαχείριση εκτελέσεων και συναλλαγών σε μια βάση δεδομένων.
- **Commands** - Για την έκδοση εντολών SQL σε μια βάση δεδομένων.
- **DataReaders** - Για την ανάγνωση μίας read-only – forward-only ροής των δεδομένων (data records) από μια SQL Server πηγή .
- **DataSets** - Για την αποθήκευση και τον προγραμματισμό σε flat data, δεδομένα XML και τα σχεσιακά δεδομένα.

- **DataAdapters -** Για την ώθηση των δεδομένων σε ένα DataSet, και τη προσαρμογή των δεδομένων σε μια βάση δεδομένων. [Ref. 6]

Connections (Συνδέσεις)

Οι συνδέσεις χρησιμοποιούνται για «να μιλήσουν» στις βάσεις δεδομένων, και αναπαριστούνται από τον provider συγκεκριμένες κατηγορίες όπως είναι η SQL Connection. Οι εντολές πάνω από τις συνδέσεις και τα εφαρμοσμένα αποτελέσματα επιστρέφονται υπό μορφή μιας ροής που μπορούν να διαβαστούν από ένα DataReader αντικείμενο, ή να ωθηθούν σε ένα DataSet αντικείμενο. [Ref. 6]

Commands (Εντολές)

Οι εντολές περιέχουν την πληροφορία που υποβάλλεται σε μια βάση δεδομένων, και αντιπροσωπεύονται από provider με συγκεκριμένες κατηγορίες όπως SQL Command. Μια εντολή μπορεί να είναι μια κλήση αποθηκευμένης διαδικασίας (stored procedure), μια ενημερωμένη δήλωση, ή μια δήλωση που επιστρέφει τα αποτελέσματα. Μπορούμε επίσης να χρησιμοποιήσουμε τις παραμέτρους εισαγωγής και εξαγωγής, και να επιστρέψουμε τις τιμές ως τμήμα της σύνταξης εντολής. [Ref. 6]

DataReaders

Το αντικείμενο DataReader είναι κάπως συνώνυμο με ένα read only/forward only δρομέα πάνω από τα δεδομένα. Ο DataReader API υποστηρίζει επίπεδα καθώς επίσης και ιεραρχικά δεδομένα. Ένα DataReader αντικείμενο αφού εκτελέσει μια εντολή επιστρέφεται σε μία βάση δεδομένων. Η διάταξη του επιστρεφόμενου DataReader αντικειμένου είναι διαφορετική από ένα recordset. Για παράδειγμα μπορούμε να χρησιμοποιήσουμε το DataReader για να παρουσιάσουμε τα αποτελέσματα μιας λίστας αναζήτησης σε μία ιστοσελίδα. [Ref. 6]

DataSets

Το DataSet αντικείμενο είναι παρόμοιο με το αντικείμενο ADO Recordset, αλλά ισχυρότερο, και με μια άλλη σημαντική διάκριση: το DataSet είναι πάντα αποσυνδεδεμένο. Το DataSet αντικείμενο αντιπροσωπεύει μια κρύπτη των δεδομένων, με τη βάση δεδομένων - σαν δομές όπως οι πίνακες, οι στήλες, οι

σχέσεις, και οι περιορισμοί. Εντούτοις, αν και ένα DataSet μπορεί και συμπεριφέρεται σαν μια βάση δεδομένων, είναι σημαντικό να αναφερθεί ότι τα DataSet αντικείμενα δεν αλληλεπιδρούν άμεσα με τις βάσεις δεδομένων, ή άλλα αρχικά δεδομένα. Αυτό επιτρέπει σε αυτόν που κάνει ανάπτυξη να λειτουργήσει με ένα πρότυπο προγραμματισμού που είναι πάντα συνεπές, ανεξάρτητα το που τα δεδομένα ανήκουν. Τα δεδομένα που προέρχονται από μια βάση δεδομένων, ένα αρχείο XML, από κώδικα, ή την εισαγωγή απο κάποιο χρήστη μπορούν να τοποθετηθούν στα DataSet αντικείμενα. Κατόπιν, καθώς γίνονται αλλαγές στο DataSet μπορούν να ακολουθηθούν και να ελεγχθούν πρίν ενημερώνουν τα αρχικά δεδομένα.

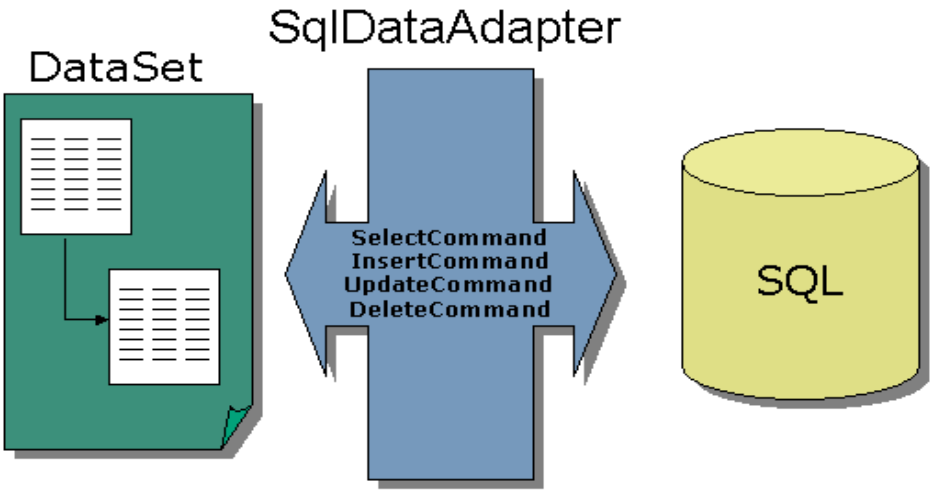
Το DataSet έχει πολλά χαρακτηριστικά XML, συμπεριλαμβανομένης της δυνατότητας να παραχθούν και να καταναλωθούν τα στοιχεία XML και τα σχήματα (schemas) XML. Τα σχήματα XML μπορούν να χρησιμοποιηθούν για να περιγράψουν τα σχήματα που ανταλλάσσονται μέσω WebServices. [Ref. 5,6]

DataAdapters (OLEDB/SQL)

Το αντικείμενο DataAdapter λειτουργεί ως γέφυρα μεταξύ του DataSet και των αρχικών δεδομένων. Χρησιμοποιώντας provider - συγκεκριμένο SqlDataAdapter (μαζί με τα σχετικά SqlCommand και τα SqlConnection του) μπορεί να αυξηθεί η γενική απόδοση όταν δουλεύουμε με Microsoft SQL Server Databases. Για άλλες OLE DB-υποστηριζόμενες βάσεις δεδομένων, θα χρησιμοποιούσαμε το OleDbDataAdapter αντικείμενο και τα σχετικά αντικείμενα OleDbCommand και OleDbConnection.

Το DataAdapter αντικείμενο χρησιμοποιεί τις εντολές για να ενημερώσει τα αρχικά δεδομένα αφότου έχουν γίνει οι αλλαγές στο DataSet. Χρησιμοποιώντας τη μέθοδο **Fill** του DataAdapter καλεί την εντολή SELECT, χρησιμοποιώντας τη μέθοδο **Update** καλεί την INSERT, UPDATE ή DELETE. Μπορούμε σαφέστατα να βάλουμε αυτές τις εντολές προκειμένου να ελεγχθούν οι δηλώσεις που χρησιμοποιούνται στο χρόνο εκτέλεσης για να επιλύσουν τις αλλαγές, συμπεριλαμβανομένης της χρήσης των αποθηκευμένων διαδικασιών (stored procedures). Για συγκεκριμένα σενάρια, ένα CommandBuilder αντικείμενο μπορεί να δημιουργεί αυτά στο χρόνο εκτέλεσης που βασίζεται σε μια ξεχωριστή δήλωση.

Εντούτοις, αυτή η δημιουργία του χρόνου εκτέλεσης απαιτεί έναν πρόσθετο κυκλικό δρόμο στον server προκειμένου να μαζευτούν τα απαραίτητα μεταδεδομένα, τόσο ρητά παράγοντας τις INSERT, UPDATE και DELETE εντολές στο χρόνο σχεδιασμού, θα οδηγήσει στην καλύτερη απόδοση του χρόνου εκτέλεσης. [Ref. 6]



Σχήμα 1 - DataAdapters and DataSets

2.1.3 XML

Κάνοντας χρήση της XML, μιας τεχνολογίας βιομηχανικού προτύπου για την περιγραφή δεδομένων, οι προγραμματιστές του Visual Studio .NET μπορούν να δημιουργήσουν εφαρμογές υψηλών επιδόσεων που βασίζονται σε δεδομένα. Οι προγραμματιστές μπορούν να χρησιμοποιήσουν ενσωματωμένα εργαλεία ADO.NET τα οποία προορίζονται για μια ποικιλία βάσεων δεδομένων, συμπεριλαμβανομένης της βάσης δεδομένων SQL Server, της βάσης δεδομένων Oracle ή οποιασδήποτε άλλης προέλευσης XML. Με την εσωτερική υποστήριξη για XML, το εργαλείο ADO.NET επιτρέπει στους προγραμματιστές να κάνουν κοινή χρήση δεδομένων σε διαφορετικές πλατφόρμες υπολογιστών. Επιπλέον, το Visual Studio .NET περιλαμβάνει το μηχανισμό Microsoft Data Engine (MSDE), μια βάση δεδομένων 100% συμβατή με τον SQL Server η οποία προσφέρει στους προγραμματιστές μια χρησιμοποιήσιμη βάση δεδομένων προγραμματισμού και υποστηρίζει τοπικά XML για μέγιστη διαλειτουργικότητα. Συνοπτικά τα οφέλη της XML είναι τα παρακάτω:

- **Απλότητα**

Οι πληροφορίες που κωδικοποιούνται σε XML είναι εύκολο να διαβαστούν και να κατανοηθούν, συν του ότι μπορούν να επεξεργαστούν εύκολα από τους υπολογιστές.

- **Openness**

Η XML είναι ένα W3C πρότυπο, που υποστηρίζεται από τους πρωτοπόρους στην αγορά βιομηχανίας λογισμικού.

- **Επεκτασιμότητα**

Δεν υπάρχει κανένα δεδομένο σύνολο tags. Νέα tags μπορούν να δημιουργηθούν δεδομένου ότι απαιτούνται.

- **Αυτό-περιγραφή**

Στις παραδοσιακές βάσεις δεδομένων, τα δεδομένα απαιτούν σχήματα (schemas) που δημιουργούνται από τον administrator της βάσης. Τα αρχεία XML μπορούν να αποθηκευτούν χωρίς τέτοιους καθορισμούς, επειδή περιέχουν meta - data υπό μορφή tags.

Η XML παρέχει μια βάση για την αναγνώριση συντακτών και ερμηνεύεται στο επίπεδο των στοιχείων. Οποιοδήποτε XML tag μπορεί να καταλαμβάνει έναν απεριόριστο αριθμό ιδιοτήτων όπως ο συντάκτης ή η έκδοση.

- **Περιέχει machine-readable πληροφορίες**

Τα tags, οι ιδιότητες και η δομή στοιχείων παρέχουν πληροφορίες που μπορούν να χρησιμοποιηθούν για να ερμηνεύσουν την έννοια του περιεχομένου, που ανοίγει νέες δυνατότητες για ιδιαίτερα αποδοτικές μηχανές αναζήτησης, intelligent data mining, agents, κ.λπ. Αυτό είναι ένα σημαντικό πλεονέκτημα πέρα από την HTML ή το απλό κείμενο, όπου οι πληροφορίες περιεχομένου είναι δύσκολο ή αδύνατο να αξιολογηθούν.

- **Χωρίζει το περιεχόμενο από την παρουσίαση**

Τα XML tags περιγράφουν τη σημασία της μη-παρουσίασης. Το ρητό της HTML είναι: “I know how it looks” («Ξέρω πως φαίνεται»), ενώ το ρητό της XML είναι: "I know what it means, and you tell me how it should look." («Ξέρω τι σημαίνει, και μου λέτε πώς πρέπει να φαίνεται.») Η όψη και η αίσθηση ενός αρχείου XML μπορούν να ελεγχθούν από XSL style sheets, επιτρέποντας την όψη ενός αρχείου (ή ενός πλήρους Web site) να αλλάζει χωρίς να επηρεάζεται το περιεχόμενο του αρχείου.

- **Υποστηρίζει πολύγλωσσα αρχεία και Unicode**

Αυτό είναι σημαντικό για τη διεθνοποίηση των εφαρμογών.

- **Διευκολύνει τη σύγκριση και τη συνάθροιση των δεδομένων**

Η δομή δέντρου των αρχείων XML επιτρέπει στα αρχεία να συγκριθούν και να αθροιστεί αποτελεσματικά στοιχείο με στοιχείο.

- **Μπορεί να ενσωματώσει πολλαπλούς τύπους αρχείων**

Τα XML αρχεία μπορούν να περιέχουν οποιοδήποτε πιθανό τύπο δεδομένων – από δεδομένα πολυμέσων (εικόνα, ήχος, βίντεο) έως active components (Java applets, ActiveX).

- **Μπορεί να ενσωματώσει υφιστάμενα δεδομένα**

Η χαρτογράφηση των υπαρχουσών δομών δεδομένων όπως είναι τα συστήματα αρχείων ή οι σχεσιακές βάσεις, σε XML γίνεται πολύ απλή. Η XML υποστηρίζει πολλαπλά format δεδομενων και μπορεί να καλύψει όλες τις υπάρχουσες δομές δεδομένων.

- **Παρέχει '*one-server view*' για τα κατανεμημένα δεδομένα**

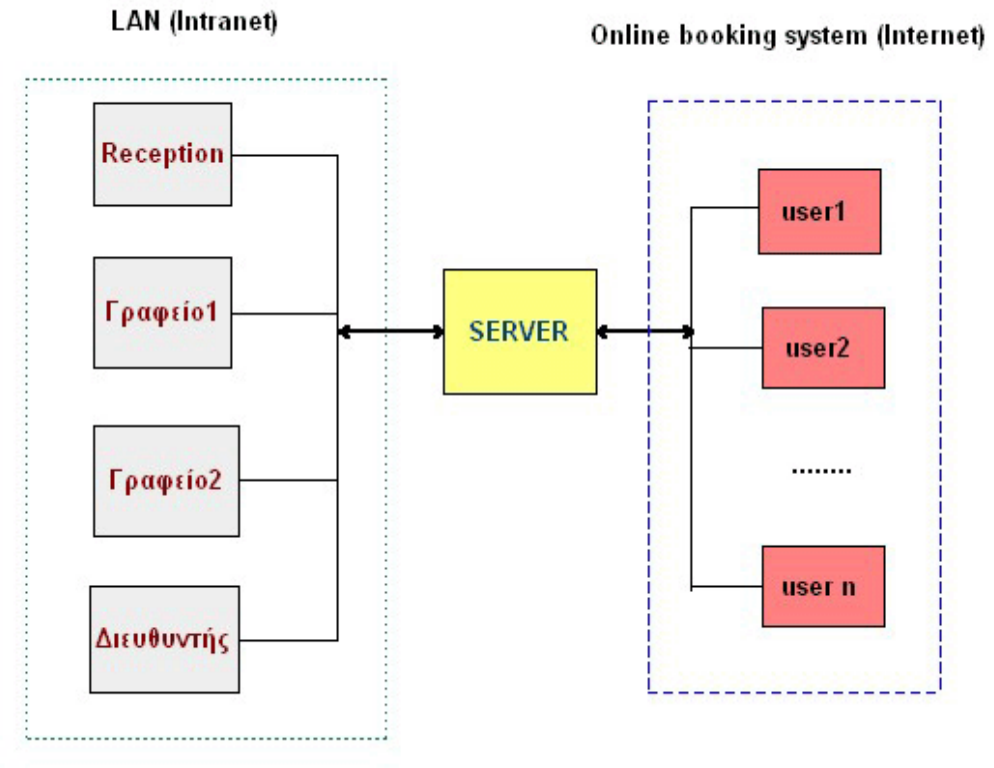
Τα XML αρχεία μπορούν να προκύψουν από τα ένθετα στοιχεία που κατανέμονται από τους πολλαπλούς remote servers. Η XML είναι αυτήν την περίοδο το

πολυπλοκότερο format για τα κατανεμημένα δεδομένα - το World Wide Web μπορεί να φανεί ως μια τεράστια XML βάση δεδομένων. [Ref. 13]

Στο δικό μας πρόγραμμα κάναμε χρήση της XML στη φόρμα ViewGraphical η οποία παίρνει τα δεδομένα για την τοποθεσία των δωματίων στο χώρο απο ένα ξεχωριστό αρχείο XML. Στο κεφάλαιο παρακάτω περιγράφεται η επιλογή μας για χρηση της XML.

2.2 Βάσεις Δεδομένων και SQL Server 2000 (MSDE)

Η ανάγκη για την χρήση μιας Βάσης Δεδομένων ήταν εξαρχής φανερή, λόγω της φύσεως του προβλήματος. Έπρεπε όλα τα δεδομένα να είναι αποθηκευμένα σε κάποιο κεντρικό σύστημα (Server) ώστε να είναι απ' ευθείας προσβάσιμα από την εφαρμογή μας και συνεπώς διαθέσιμα στους χρήστες (Users) είτε προέρχονται από την εσωτερική υποδομή (Intranet) ή από το σύστημα online booking (Internet) (Σχήμα 2).



Σχήμα 2 – Επικοινωνία Server με χρήστες (LAN – Internet)

Επιλέξαμε ως Βάση Δεδομένων SQL Server 2000 διότι πάμω σε αυτό ήταν σχεδιασμένο το σύστημα του online booking. Τα κριτήρια επιλογής της βάση ήταν να υποστηρίζει πολλούς χρήστες ταυτόχρονα, να υπάρχει η κατάλληλη τεχνογνωσία και να είναι τύπου server. Τελικά επιλέχθηκε ο SQL Server γιατί ήταν η Βάση Δεδομένων που πληρούσε τα περισσότερα κριτήρια.

Το Microsoft SQL Server είναι ένα σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων (relational database management system, RDBMS), σχεδιασμένο για εφαρμογές επεξεργασίας συναλλαγών εντός επικοινωνίας (online transaction processing, OLTP), αποθηκών δεδομένων (data warehousing), και ηλεκτρονικού εμπορίου (e-commerce) με υψηλές απαιτήσεις. [Ref. 1]

Το Microsoft SQL Server 2000 είναι σχεδιασμένο για να υποστηρίζει βάσεις δεδομένων μεγάλου όγκου και κρίσιμης σημασίας σε πολλά διαφορετικά πεδία εφαρμογών. Για την υποστήριξη αυτών των λειτουργιών, το SQL Server διαθέτει ορισμένα εργαλεία, στα οποία περιλαμβάνονται κάποια βοηθητικά προγράμματα γραμμής διαταγών όπως το bcp.exe, το οποίο αντιγράφει δεδομένα μεταξύ του SQL Server και ενός αρχείου του λειτουργικού συστήματος και τον Enterprise Manager, ένα προηγμένο παραστατικό εργαλείο για τη διαχείριση πολλών βάσεων δεδομένων και του ίδιου του SQL Server. [Ref. 1]

Πιο συγκεκριμένα για την εφαρμογή μας χρησιμοποιήθηκε η MSDE έκδοση του SQL Server 2000 βάσης δεδομένων η οποία ουσιαστικά είναι μία «κομμένη» έκδοση του εμπορικού SQL Server και διατίθεται δωρεάν από τη Microsoft. Έχει διάφορους περιορισμούς ως προς το μέγιστο μέγεθος της βάσης δεδομένων (4GB), τον αριθμό των ταυτόχρονων συνδέσεων και την απουσία κάποιου client διαχείρισης (Enterprise Manager). Ωστόσο το γεγονός ότι δίδεται δωρεάν κάνει την έκδοση MSDE ιδιαίτερα δελεαστική εφόσον είναι 100% συμβατή με SQL Server, έτσι αν παρουσιαστούν αυξημένες ανάγκες, τότε αρκεί μια απλή εγκατάσταση της εμπορικής έκδοσης για να γίνει το upscaling.

Κεφάλαιο 3

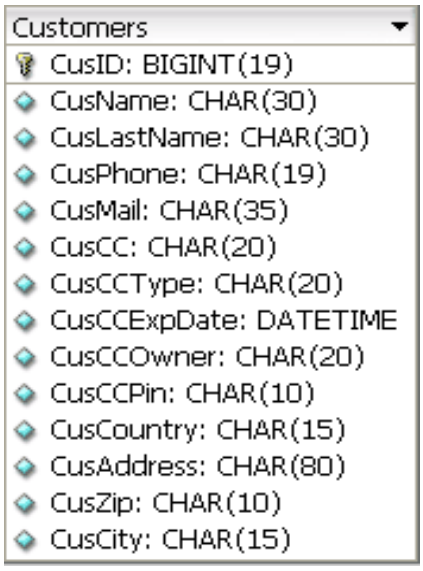
3. Εφαρμογή – Hotels Manager

3.1 Περιγραφή της Βάσης Δεδομένων

3.1.1 Πίνακες της Βάσης Δεδομένων

Η βάση δεδομένων αποτελείται από τους παρακάτω πίνακες:

- Πίνακας Customers:

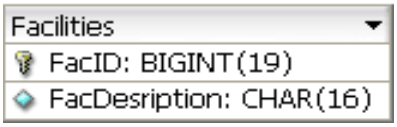


Εικόνα 1 - Πίνακας Customers

Στο παραπάνω πίνακα (Εικόνα 1) γίνεται η εισαγωγή των στοιχείων του πελάτη.

Ο πίνακας περιλαμβάνει το όνομα, το επίθετο, το τηλέφωνο, το mail, τη χώρα διαμονής, την πόλη, τον ταχυδρομικό κωδικό, τη διεύθυνση του πελάτη καθώς και τον τύπο, τον αριθμό, την ημερομηνία λήξης, τον κάτοχο και τον κωδικό ασφαλείας της πιστωτικής κάρτας του πελάτη. Παρατηρούμε πως πρωτεύων κλειδί είναι το πεδίο CusID το οποίο παίρνει αυτόματα ένα αύξοντα αριθμό από την βάση.

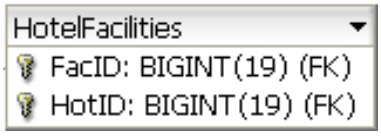
- Πίνακας Facilities:



Εικόνα 2 - Πίνακας Facilities

Ο παραπάνω πίνακας (Εικόνα 2) περιέχει τις υπηρεσίες των ξενοδοχείων και των δωματίων καθώς και το πρωτεύον κλειδί, έναν αύξοντα αριθμό που δημιουργεί η βάση.

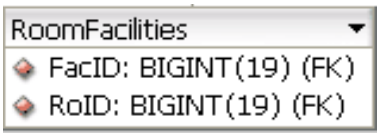
- Πίνακας HotelFacilities:



Εικόνα 3 - Πίνακας HotelFacilities

Ο παραπάνω πίνακας (Εικόνα 3) περιέχει δυο πεδία, το ID του ξενοδοχείου και το ID της υπηρεσίας (facilities). Είναι ουσιαστικά ο ενδιάμεσος πίνακας που συνδέει τα ξενοδοχεία με τα facilities που περιέχουν.

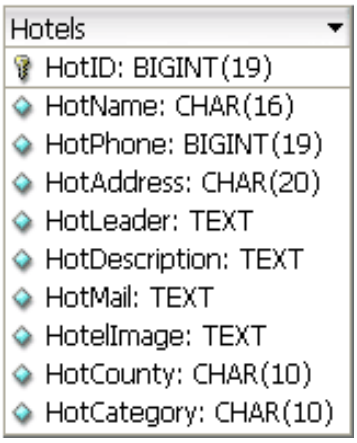
- Πίνακας RoomFacilities:



Εικόνα 4 - Πίνακας RoomFacilities

Ο παραπάνω πίνακας (Εικόνα 4) περιέχει δυο πεδία, το ID του δωματίου και το ID της υπηρεσίας (facilities). Είναι ουσιαστικά ο ενδιάμεσος πίνακας που συνδέει τα δωμάτια με τα facilities που περιέχουν.

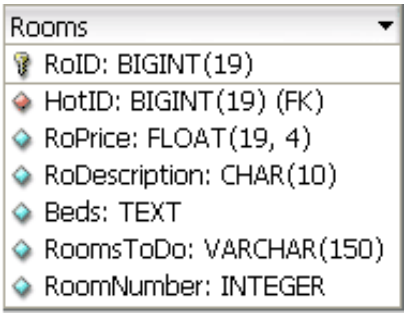
- Πίνακας Hotels:



Εικόνα 5 - Πίνακας Hotels

Ο παραπάνω πίνακας (Εικόνα 5) περιέχει τα στοιχεία των συμβαλλόμενων ξενοδοχείων με πρωτεύων κλειδί έναν αύξοντα αριθμό που δημιουργεί η βάση καθώς και όλα τα στοιχεία που πρέπει να γνωρίζουμε για αυτά. Ο συγκεκριμένος πίνακας δε μας χρειάστηκε κάπου εφόσον η εφαρμογή μας αφορά μία μόνο μονάδα ξενοδοχείου.

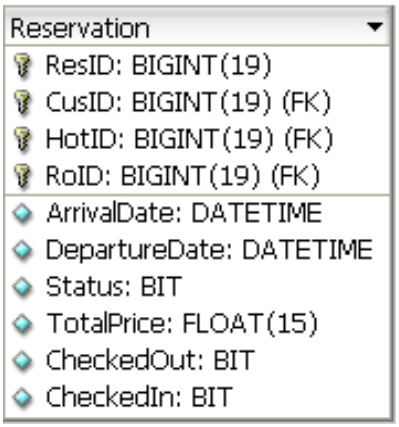
- Πίνακας Rooms:



Εικόνα 6 - Πίνακας Rooms

Ο παραπάνω πίνακας (Εικόνα 6) περιέχει τα στοιχεία των δωματίων των συμβαλλόμενων ξενοδοχείων με πρωτεύων κλειδί έναν αύξοντα αριθμό που δημιουργεί η βάση καθώς και όλα τα στοιχεία που πρέπει να γνωρίζουμε για τα δωμάτια αυτά.

- Πίνακας Reservation:



Εικόνα 7 - Πίνακας Reservation

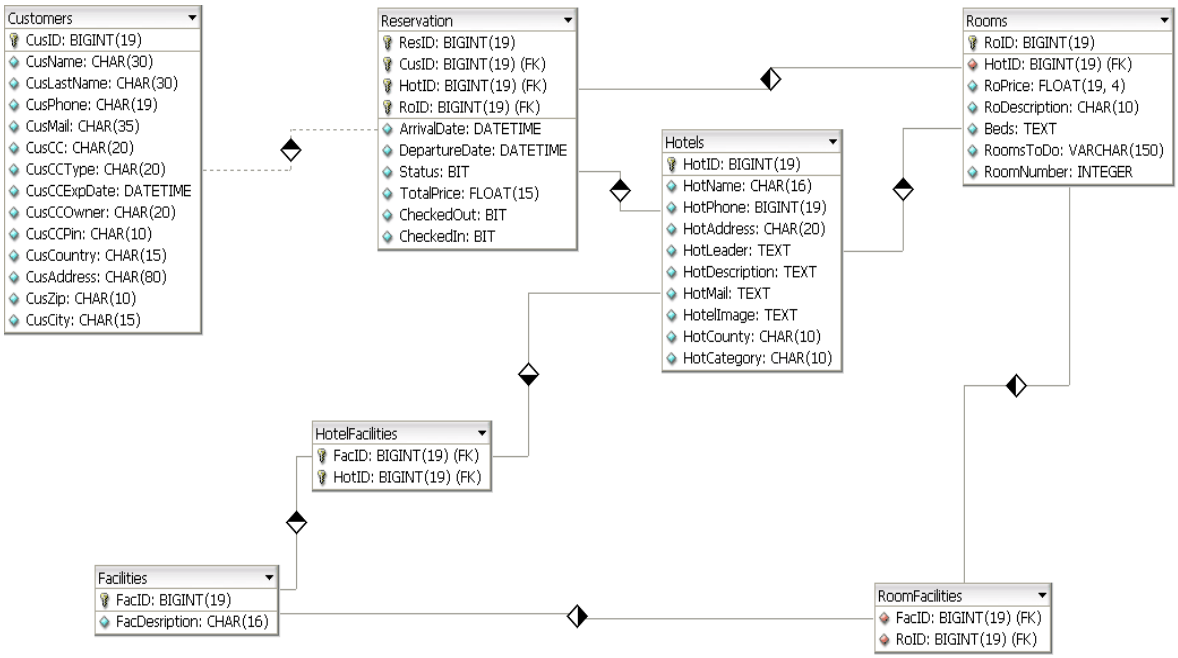
Ο παραπάνω πίνακας (Εικόνα 7) περιέχει τις κρατήσεις των δωματίων των συμβαλλόμενων ξενοδοχείων. Σε αυτόν το πίνακα γίνεται η αναζήτηση των διαθέσιμων δωματίων. Ο πίνακας αυτός έχει πρωτεύων κλειδί, τον κωδικό της κράτησης και επίσης τα κλειδιά: τον κωδικό του πελάτη, τον κωδικό του ξενοδοχείου και τον κωδικό του δωματίου.

3.1.2 Διάγραμμα βάσης δεδομένων – Συσχετίσεις πινάκων

Το διάγραμμα της βάσης δεδομένων είναι μια γραφική αναπαράσταση των πινάκων και των μεταξύ τους σχέσεων (Εικόνα 8). Η σχεδίασή του είναι ένας καλός τρόπος για να έχουμε μια εποπτική εικόνα της δομής και των συσχετίσεων της βάσης δεδομένων. Εκτός αυτού είναι ένας βολικός τρόπος τεκμηρίωσης της βάσης, επειδή εκεί απεικονίζονται αυτόματα όποιες αλλαγές και να κάνουμε. [Ref. 1]

Στην Εικόνα 8 φαίνεται το διάγραμμα της βάσης δεδομένων μας. Μπορούμε να παρατηρήσουμε τις σχέσεις μεταξύ των πινάκων. Ο πίνακας Customers συνδέεται με τον πίνακα Reservation στο πεδίο CusID με σχέση «ένα προς πολλά» δηλαδή ένας πελάτης μπορεί να κάνει πολλές κρατήσεις. Ο πίνακας Hotels συνδέεται με το πίνακα Reservation στο πεδίο HotID με σχέση «ένα προς πολλά» δηλαδή ένα ξενοδοχείο μπορεί να έχει πολλές κρατήσεις. Ο πίνακας Rooms συνδέεται με το πίνακα

Reservation στο πεδίο RoID με σχέση «ένα προς πολλά» δηλαδή ένα δωμάτιο μπορεί να έχει κλειστεί πολλές φορές. Ο πίνακας Hotels συνδέεται με τον πίνακα Rooms στο πεδίο HotID με σχέση «ένα προς πολλά» δηλαδή ένα ξενοδοχείο μπορεί να έχει πολλά δωμάτια. Ο πίνακας Hotels συνδέεται με τον πίνακα Facilities μέσω του ενδιάμεσου πίνακα HotelFacilities με σχέση «ένα προς πολλά» δηλαδή ένα ξενοδοχείο μπορεί να προσφέρει πολλές υπηρεσίες. Ο πίνακας Rooms συνδέεται με τον πίνακα Facilities μέσω του ενδιάμεσου πίνακα RoomFacilities με σχέση «ένα προς πολλά» δηλαδή ένα δωματίο μπορεί να προσφέρει πολλές υπηρεσίες.

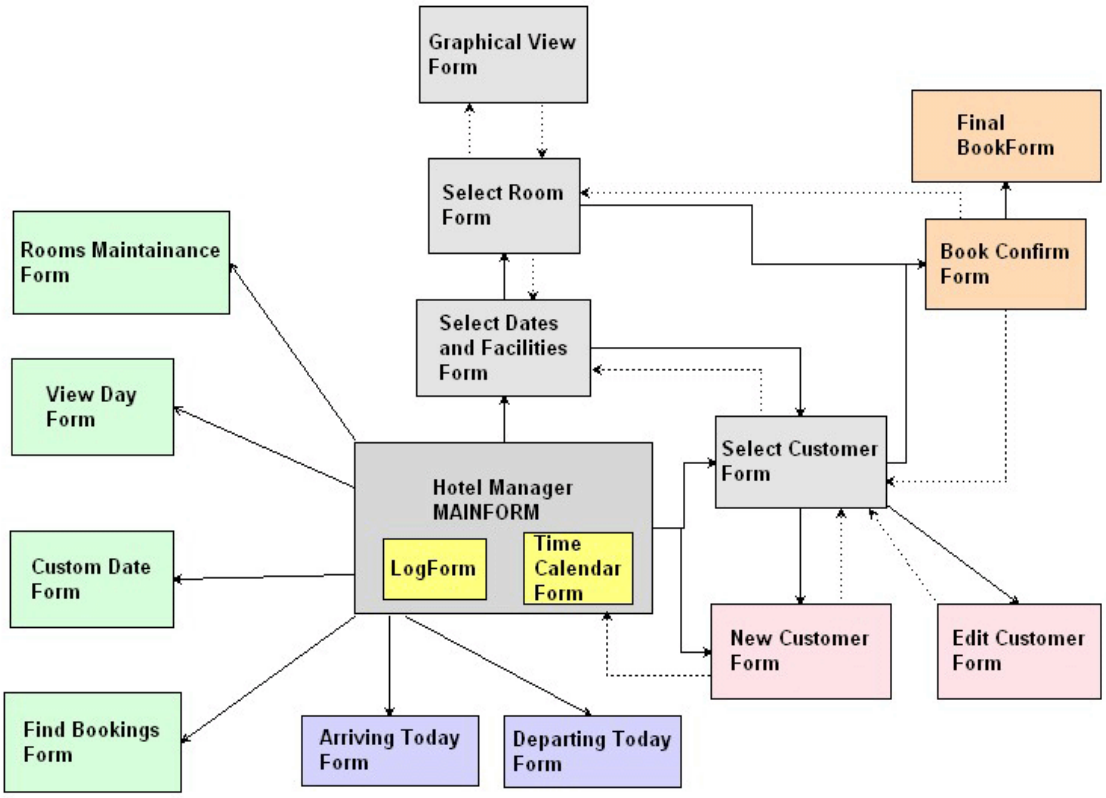


Εικόνα 8 - Διάγραμμα βάσης δεδομένων

3.2 Αρχιτεκτονική Εφαρμογής

3.2.1 Φόρμες και αλληλεπίδραση μεταξύ τους και με τη βάση δεδομένων

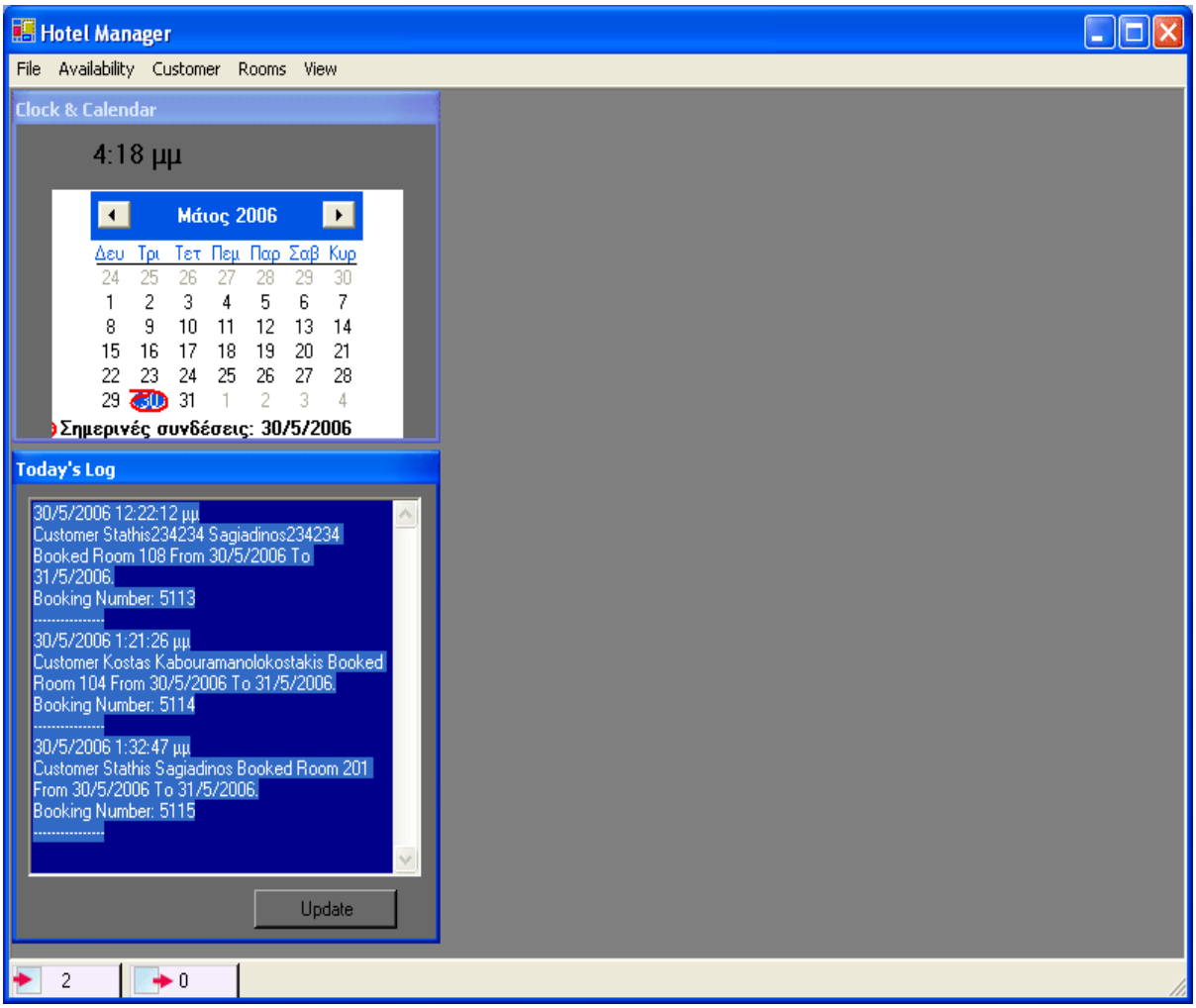
Στο παρακάτω σχήμα (Σχημα 3) βλέπουμε πως αλληλεπιδρούν οι διάφορες φόρμες μεταξύ τους.



Σχήμα 3 – Σύνολο φορμών και αλληλεπίδραση μεταξύ τους

3.2.1.1 Main Form

Ανοίγοντας την εφαρμογή Hotel Manager θα αντικρύσουμε τη κυρίως φόρμα (Main Form) από την οποία γίνεται όλη η διαχείρηση (Εικόνα 9). Βλέπουμε ότι εντός της κυρίως φόρμας εμπεριέχονται και δυο δευτερεύουσες φορμες. Η μια δείχνει την ώρα και την ημερομηνία και η άλλη μας εμφανίζει το ιστορικό (Log) της τρέχουσας ημέρας. Περισσότερα για αυτές τις φόρμες θα αναφερθούν παρακάτω.



Εικόνα 9 – Κυρίως Φόρμα (Main Form)

3.2.1.2 Booking Procedure

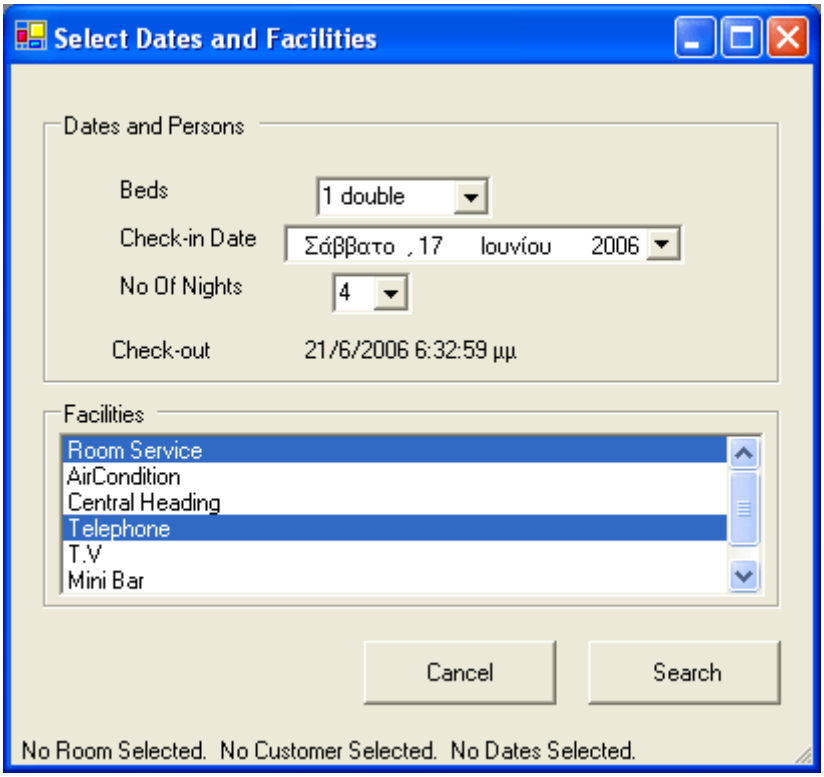
Στην Εικόνα 10 βλέπουμε με ποιό τρόπο θα ξεκινήσουμε τη διαδικασία του Booking. Για να γίνει ένα Booking υπάρχουν τρείς τρόποι. Ο ένας είναι να ξεκινήσουμε πρώτα από την επιλογή ημερομηνιών και στοιχειών των δωματίων, τον οποίο θα περιγράψουμε εδώ. Ένας άλλος τρόπος είναι να ξεκινήσουμε πρώτα περνώντας τα στοιχεία του πελάτη. Επίσης μπορούμε να ξεκινήσουμε το booking procedure απο τη φόρμα εύρεσης και επιλογής ήδη υφιστάμενων πελατών. Ουσιαστικά αυτό μας το επιτρέπει η εξης φιλοσοφία του προγράμματος. Έχουμε δημιουργήσει μια κλάση που ονομάζουμε myReservation η οποία εμπεριέχει τις ιδιότητες που χρειάζεται να

συμπληρωθούν για να ολοκληρωθεί μια κράτηση. Οι πληροφορίες αυτές είναι το όνομα του πελάτη, ο αριθμός δωματίου, οι ημερομηνίες check-in και check-out και συμπληρωματικά το κόστος διαμονής. Ένα object που ονομάζεται myBookingInfo και είναι instance της κλάσης myReservation περνιέται απο φόρμα σε φόρμα. Η κάθε φόρμα που σχετίζεται με το reservation διαπιστώνει αν λείπει κάτι απο τα properties του myBookingInfo και ανάλογα μας παραπέμπει στις υπόλοιπες απαραίτητες φόρμες για την συμπλήρωση αυτων των στοιχείων. Όταν συμπληρωθούν όλα τα στοιχεία του myBookingInfo(myReservation) τότε μας στέλνει αυτόματα για confirmation (Book Confirm Form). Οπότε, είτε ξεκινήσουμε από τη φόρμα "Select Room and Dates", είτε απο τη "Select Customer" είτε από την "Add Customer", μπορούμε να επιτύχουμε ενα booking χρησιμοποιώντας τις ίδιες φόρμες, αλλά απλώς με διαφορετική σειρά. Επίσης, παρατηρούμε ότι όλες οι φόρμες που σχετίζονται με το booking, αναγράφουν στις status bars τους την κατάσταση του object myBookingInfo, δηλ. Αν έχουν επιλεγεί Room, Customer και Dates. Έτσι ο χρήστης έχει πλήρη γνώση της κατάστασης του Booking κατα τη διαδικασία του.

Εικόνα 10 – Ξεκινάμε τη διαδικασία του Booking

Λοιπόν αφού πάμε στο «Availability» και επιλέξουμε «Book Now» θα μας ανοίξει η φόρμα «Select Dates and Facilities» (Εικόνα 11) στην οποία επιλέγουμε πόσες κλίνες θέλουμε να έχει το δωμάτιο, τις ημερομηνίες που μας ενδιαφέρουν καθώς και τα facilities (υπηρεσίες) που θέλουμε να μας παρέχονται στο δωμάτιο.

Όσον αφορά τις ημερομηνίες, η εφαρμογή μας δίνει τη δυνατότητα εφόσον επιλέξουμε την ημερομηνία που θέλει ο πελάτης να διαμείνει στο δωμάτιο, να επιλέξουμε απλά πόσα βράδυα σκοπεύει να μείνει και αυτόματα θα μας αναγραφεί στο πεδίο Check-out η ημερομηνία αναχώρησης του.



Εικόνα 11 – BookForm1 (Select Dates and Facilities)

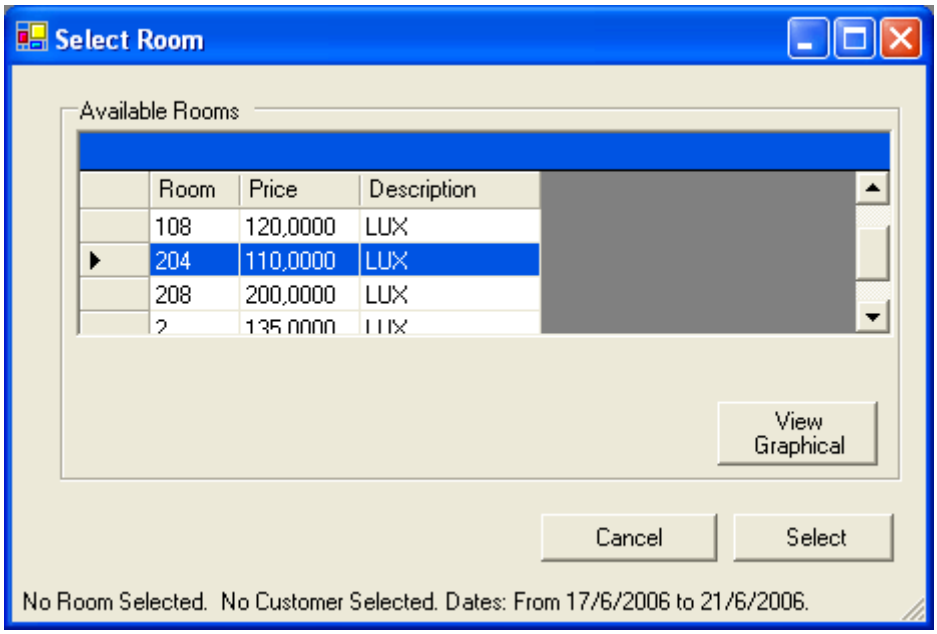
Επιλέγοντας τα επιθυμητά στοιχεία, πατάμε το κουμπί Search. Το σύστημα συνδέεται στη βάση δεδομένων και περνάει το ακόλουθο query:

```
SELECT Distinct(Rooms.RoID), Rooms.RoomNumber, Rooms.RoPrice,
Rooms.RoDescription FROM Rooms, RoomFacilities WHERE RoomFacilities.RoID =
Rooms.RoID AND   Rooms.Beds LIKE '" & TypeofBeds & "%' And Rooms.HotID = 1
AND EXISTS   (Select Distinct RoId  from roomFacilities WHERE " &
BuildFacQuery() & " AND RoomFacilities.RoId = Rooms.RoID) AND NOT EXISTS
(SELECT Reservation.RoId FROM Reservation WHERE Reservation.RoID =
Rooms.RoID AND ((@ArrivalDate >= Reservation.ArrivalDate AND @DepartureDate
<= Reservation.DepartureDate) OR (@ArrivalDate < Reservation.ArrivalDate AND
@DepartureDate > Reservation.ArrivalDate) OR (@ArrivalDate <
Reservation.DepartureDate AND @DepartureDate > Reservation.DepartureDate)))
```

* Μεταβλητές και Functions στο παραπάνω query:

- TypeofBeds: Πέρνει 3 διαφορετικές τιμές (1 double, 2 single, 3 single) απο το DropDown Menu
- BuildFacQuery: Query Builder function για τα facilities των δωματίων ανάλογα με τις επιλογές στο ListBox

Βάση του παραπάνω query επιστρέφει τα δωμάτια σε ένα καινούριο dataset (DsRooms). Κατόπιν ανοίγει την επόμενη φόρμα (SelectRoomsForm) και γεμίζει το datatagrid με τα αποτελέσματα αφού πρώτα μορφοποιηθούν σωστά σε ένα καινούριο dataTable.



Εικόνα 12 – Επιλογή δωματίου από DataGrid

Στη φόρμα Select Room μπορούμε να επιλέξουμε απευθείας το δωμάτιο που επιθυμούμε ή να πάμε να το επιλέξουμε από τη Graphical View μορφή του ξενοδοχείου.

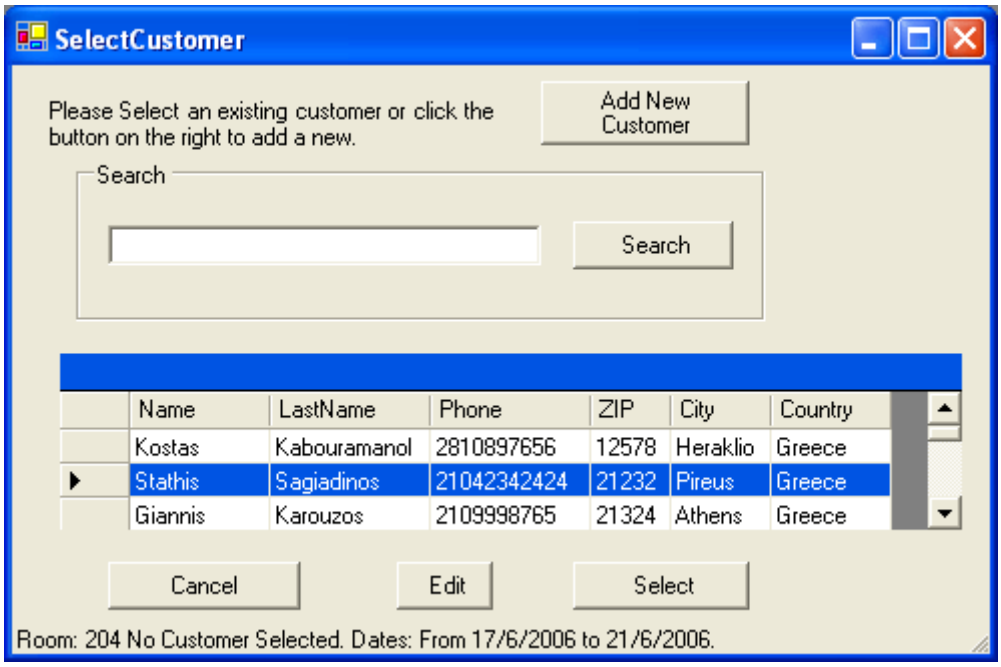
Εικόνα 13 - Graphical View

Όταν περάσουμε σε Graphical View θα παρατηρούμε ότι υπάρχουν πάνω από τα δωμάτια κάποια χρώματα. Πιο συγκεκριμένα τα χρώματα είναι τρία κόκκινο, πράσινο

και μπλέ. Το κάθε χρώμα προσδίδει και μία συγκεκριμένη κατάσταση για το δωμάτιο που το έχει. Το κόκκινο χρώμα δείχνει ότι το δωμάτιο δεν είναι ελεύθερο και συνεπώς δε μπορούμε να το επιλέξουμε, το πράσινο ότι το δωμάτιο είναι ελεύθερο και τέλος το μπλέ δείχνει ότι το δωμάτιο είναι ελεύθερο αλλά έχει ήδη επιλεχθεί. Περισσότερα για το Graphical View θα δούμε παρακάτω.

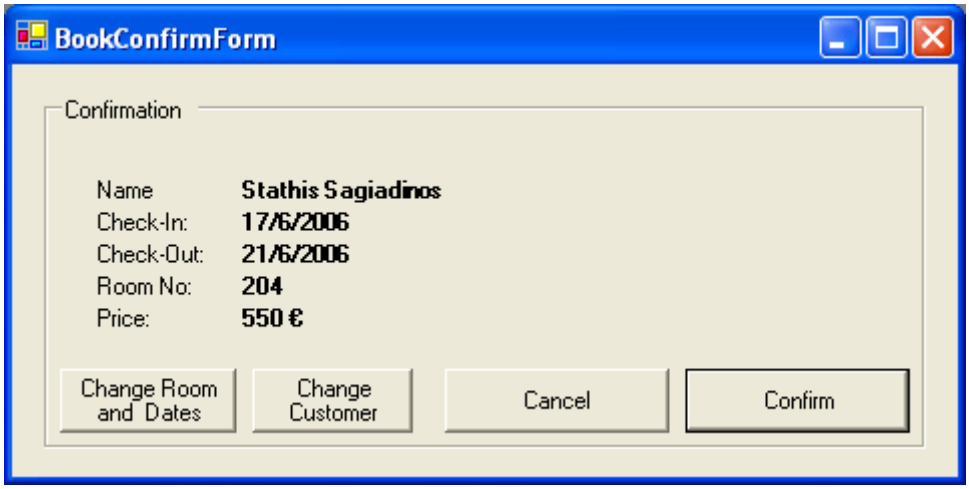
Σε περίπτωση που το επιλέξουμε απευθείας από το datagrid, τότε το πρόγραμμα μας στέλνει στην επιλογή πελάτη (Select Customer Form) (Εικόνα 14). Εκεί μπορούμε να κάνουμε μια αναζήτηση με βάση το επώνυμο του πελάτη. Το παρακάτω query το αντικατοπτρίζει:

```
"SELECT * FROM Customers WHERE CusLastName LIKE @CusLastName"
```



Εικόνα 14 – Φόρμα Select Customer

Επιλέγοντας και τον πελάτη, τότε το myBookingInfo (Instance του myReservation) πλέον περιέχει όλα τα απαραίτητα στοιχεία που χρειάζονται για να γίνει το booking. Οπότε μας παραπέμπει στη φόρμα επιβεβαίωσης (Book Confirm Form).



Εικόνα 15 – Φόρμα Book Confirm

Η Φόρμα επιβεβαίωσης εμφανίζει τα στοιχεία του myBookingInfo και δίνει τη δυνατότητα στο χρήστη είτε να τα επιβεβαιώσει είτε να αλλάξει Ημερομηνίες, δωμάτια ή ακόμα και τον πελάτη. Εαν πχ. Επιλέξουμε να αλλάξουμε τον πελάτη, το σύστημα επιστρέφει στη φορμα Select Customer, όπου διαλέγουμε καινούριο άτομο και μας επιστρέφει πίσω στο Confirmation.

Άπαξ και ο χρήστης πατήσει το πλήκτρο "Confirm" τότε το σύστημα συνδέεται με τη βάση και περνάει τα στοιχεία που επιλέχθηκαν απο τις προηγούμενες φόρμες και εμπεριέχονται στο myBookingInfo(myReservation). Αυτό φαίνεται απο το παρακάτω query:

```
INSERT INTO Reservation(CusId, HotId, RoID, ArrivalDate,
DepartureDate, TotalPrice) VALUES (@CusId, @HotId, @RoID,
@ArrivalDate, @DepartureDate, @TotalPrice)"
```

Με το confirm επίσης, συντάσσεται ενα log text το οποίο περνιέται στη βάση για να υπάρχει έλεγχος όλων των διεργασιών που έχουν ακολουθηθεί. Η φόρμα LogForm αμέσως ανανεώνει τα στοιχεία της έτσι ώστε να περιλαμβάνει και το τελευταίο Booking.

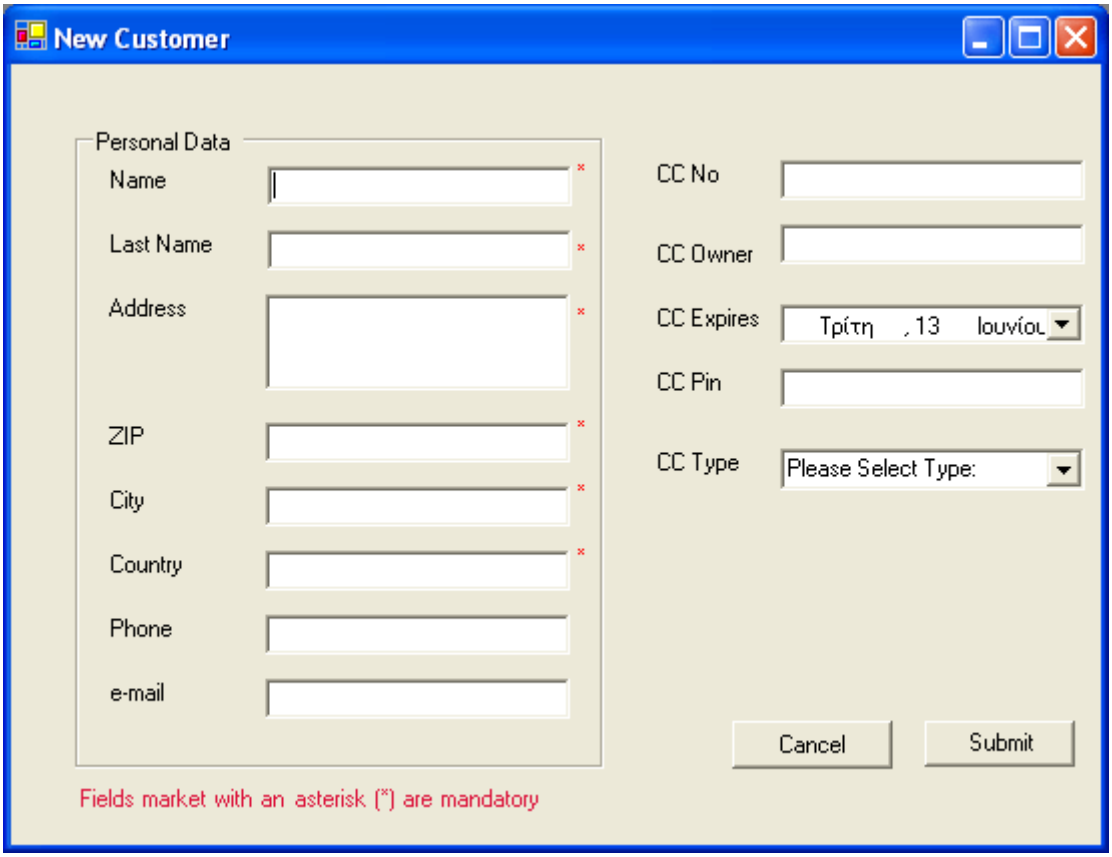
Τέλος, και εφόσον πλέον το booking έχει καταχωρηθεί στη βάση, εύκολα παίρνουμε το τελευταίο πρωτεύον κλειδί του πίνακα Reservation (το ResID) το οποίο είναι

μοναδικό και εμφανίζεται ως Booking Number στη φόρμα FinalBookForm (Εικόνα 16).

Εικόνα 16 – Final Book Form

3.2.1.3 Add Customer Form

Στην περίπτωση που έχουμε ένα νεο πελάτη, και επιθυμούμε να τον περάσουμε στο σύστημα, τότε επιλέγουμε απο το menu bar: Customer – New Customer. Η φόρμα New Customer (Εικόνα 17) ανοίγει και μας ζητάει να εισάγουμε τα προσωπικά του στοιχεία όπως το όνομα, το επώνυμο, Διεύθυνση, ΤΚ, Πόλη, Χώρα, E-mail, και τηλέφωνο. Επιπλέον, σε περίπτωση που ο χρήστης επιθυμεί να χρησιμοποιήσει την πιστωτική του κάρτα για τις πληρωμές προς το ξενοδοχείο, δίδεται η δυνατότητα να εισάγουμε τα στοιχεία της πιστωτικής κάρτας. Στην περίπτωση αυτή, το σύστημα πραγματοποιεί μια σειρά απο validations για να ελαχιστοποιηθεί η περίπτωση λάθους ή παράλειψης την ώρα της εισαγωγής στοιχείων. Πχ, ελέγχει αν όλα τα στοιχεία έχουν εισαχθεί. Επίσης ελέγχει αν ο αριθμός της κάρτας έχει παραπάνω απο 13 ψηφία κλπ.



Εικόνα 17 – Add Customer Form (New Customer)

Ύστερα απο την εισαγωγή νέου πελάτη, το σύστημα ρωτάει αν θέλουμε να συνεχίσουμε να κάνουμε booking με τον καινούριο πελάτη. Αν ο χρήστης επιλέξει 'yes' τότε, περνάει τα απαραίτητα στοιχεία του χρήστη πάλι σε ένα instance του myReservation (myBookingInfo) και το πασάρει στη φόρμα BookForm1 (Select Room and Dates) η οποία συνεχίζει τη διαδικασία του booking όπως προαναφέραμε.

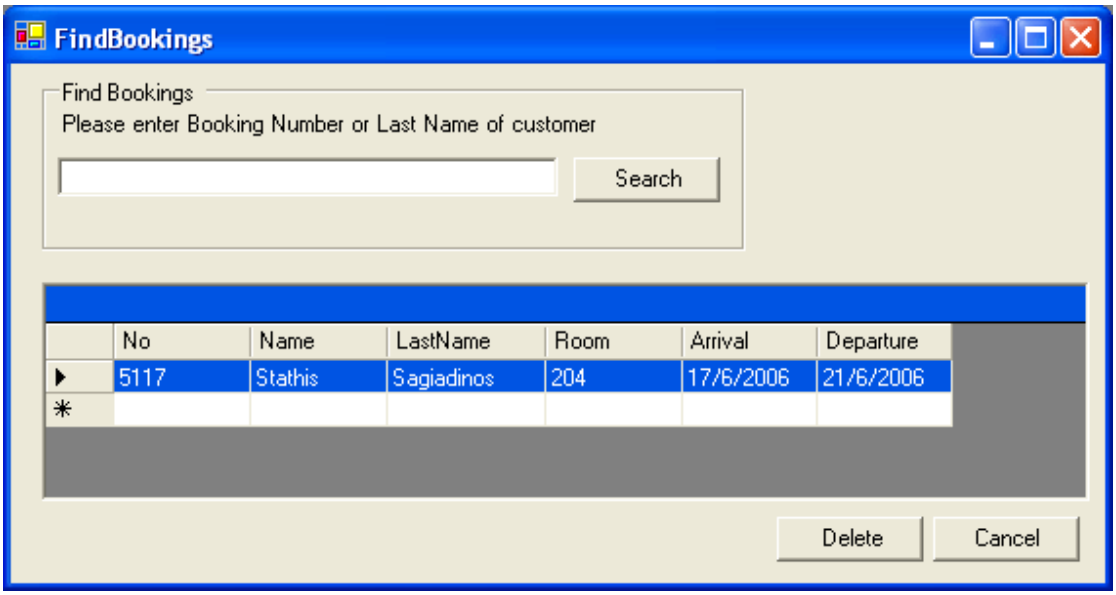
3.2.1.4 Edit Customer Form

Επιλέγοντας κάποιο πελάτη από τη φόρμα Select Customer (Εικόνα 14), εμφανίζεται η επιλογή Edit. Απο εκεί ο χρήστης έχει τη δυνατότητα να τροποποιήσει ενα ή περισσότερα απο τα στοιχεία του πελάτη. Μπορεί να χρησιμοποιηθεί επίσης για επιβεβαίωση των στοιχείων σε πραγματικό χρόνο.

3.2.1.5 Find and Delete Booking

Στη Main Form επιλέγοντας από την menu bar αρχικά View και κατόπιν πηγαίνοντας στο Find Bookings ανοίγουμε τη αντίστοιχη φόρμα 'Find Bookings' (Εικόνα 19).

Εικόνα 18 – Ανοίγουμε τη Find Booking Form



Εικόνα 19 – Find Bookings Form

Με αυτή τη λειτουργία, ο χρήστης έχει τη δυνατότητα να βρεί πληροφορίες για κάποιο booking, για να ενημερωθεί ή για να το διαγράψει. Υπάρχει ένα search field στο οποίο γράφεται μέρος ή ολόκληρη η λέξη κλειδί (booking number ή Επώνυμο

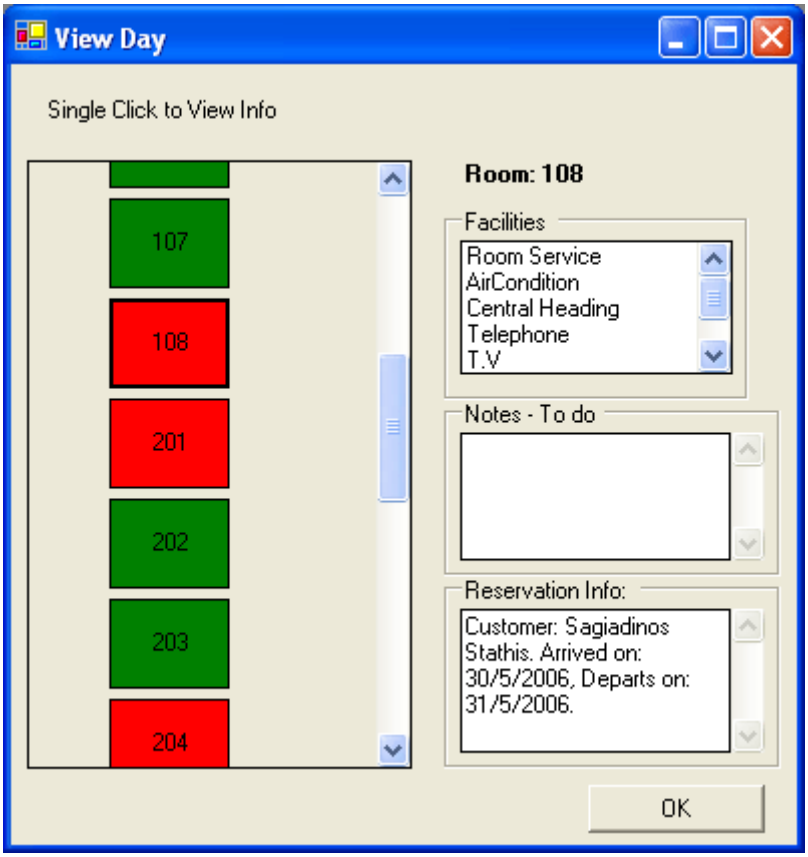
πελάτη) και ύστερα το σύστημα μας επιστρέφει στο datagrid τα αποτελέσματα. Το query το οποίο περνάει περιγράφεται εδω:

```
SELECT * FROM Reservation INNER JOIN Customers ON Reservation.CusId =
Customers.CusId INNER JOIN Rooms on Rooms.RoID = Reservation.RoID
WHERE (ResID LIKE @Criteria OR CusLastName LIKE @Criteria) AND
DepartureDate >= @DateNow
```

3.2.1.6 View Day

Επιλέγοντας στο menu bar της Main Form από το Rooms → Today (Εικόνα 20), εμφανίζουμε τη φόρμα ViewDay (Εικόνα 21).

Εικόνα 20 – Επιλέγουμε από Rooms το Today



Εικόνα 21 – View Day Form

Στη συγκεκριμένη φόρμα, ο χρήστης μπορεί να έχει αναλυτική εικόνα για το ποιά δωμάτια του ξενοδοχείου ειναι ελεύθερα και ποιά όχι την τρέχουσα ημέρα. Τα διαθέσιμα εμφανίζονται με πράσινο χρώμα ενώ τα μη διαθέσιμα με κόκκινο. Παράλληλα, κάνοντας κλικ σε κάποιο δωμάτιο, μπορεί να δεί διάφορες πληροφορίες στα δεξιά textboxes όπως: Τα facilities του δωματίου, τα Notes / To do / Εκκρεμότητες ή πληροφορίες κράτησης (σε περίπτωση που το δωμάτιο είναι κλεισμένο).

Η συγκεκριμένη φόρμα λειτουργεί με την εξής λογική:
Έχει δημιουργηθεί μια κλάση η οποία κληρονομεί από το windows.forms.button αλλά προσθέτει ένα extra property (myDate), που αναφέρεται στην ημερομηνία. Οπότε στο καινούριο μας κουμπί, μπορούμε να περάσουμε την πληροφορία του Room Number, την οποία περνάμε στο property text, αλλά πλέον και την ημερομηνία στην οποία αναφερόμαστε.

Οπότε πρώτα το πρόγραμμα μπαίνει στη βάση και ψάχνει όλα τα δωμάτια τα οποία βάζει σε ένα array of buttons (myRoomsButton) τα οποία και χρωματίζει πράσινα χρησιμοποιώντας το `System.Drawing.Color` property του button.

```
SELECT *  FROM Rooms WHERE HotID = 1
```

Κατόπιν, ξαναπερνάει απο τη βάση και ψάχνει να βρεί ποιά απο τα δωμάτια είναι κλεισμένα τα οποία χρωματίζει κόκκινα.:

```
SELECT * FROM Rooms INNER JOIN RoomFacilities ON RoomFacilities.RoID
= Rooms.RoID WHERE (HotID = 1) AND NOT EXISTS (SELECT
Reservation.RoId FROM Reservation WHERE Reservation.RoID = Rooms.RoID
AND (@DateNow > Reservation.ArrivalDate AND @DateNow <=
Reservation.DepartureDate) ))
```

Αμέσως μετά, διαβάζει το array of buttons (myRoomsButton) και τα τοποθετεί στη φόρμα στο συγκεκριμένο panel. Εφόσον λοιπόν έχουμε τα buttons στη φόρμα και γνωρίζουμε τον αριθμό δωματίου απο το text property του κάθε κουμπιού αλλά και την ημερομηνία, μπορούμε εύκολα με ένα απλό κλικ event να περάσουμε άλλο ένα

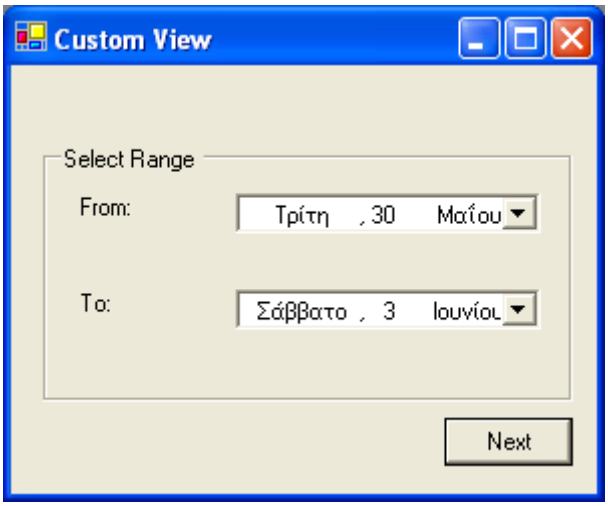
Select query στη βάση και να βρούμε και τις απαραίτητες πληροφορίες που εμφανίζουμε στα textboxes δεξιά.

3.2.1.7 Custom View

Επιλέγοντας στο menu bar της Main Form από το Rooms → Custom Date (Εικόνα 22), εμφανίζουμε τη φόρμα Custom View 1 (Εικόνα 23).

Εικόνα 22 – Επιλέγουμε από Rooms το Custom Date

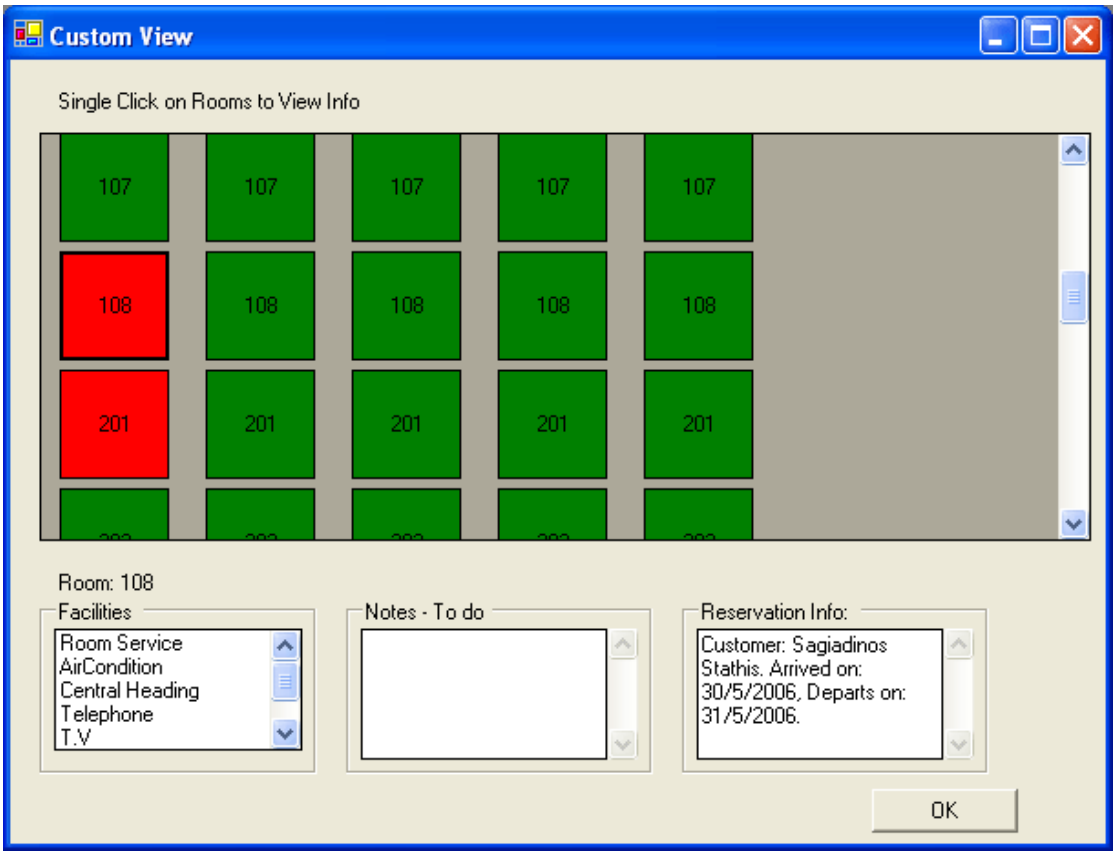
Από εδώ ο χρήστης μπορεί να επιλέξει τις ημερομηνίες για τις οποίες θέλει να εμφανιστούν πληροφορίες για τα δωμάτια. Έτσι πατώντας το κουμπί Next περνάμε στην φόρμα CustomView2 (Εικόνα 24).



Εικόνα 23 – Custom View Form 1

Εδώ ο χρήστης μπορεί να έχει την ίδια λειτουργικότητα με το View Day, δηλ. να βλέπει σε γραφική μορφή την κατάσταση των δωματίων και άλλες πληροφορίες, μόνο που εδώ δίνεται η δυνατότητα να επιλεγούν παραπάνω απο 1 ημέρες και για οποιοδήποτε χρονικό διάστημα. Οπότε, αφού γνωρίζουμε απο την Custom View1 την

αρχική και την τελική ημερομηνία, το σύστημα μας εμφανίζει μια φόρμα παρόμοια με του ViewDay η οποία πλέον έχει σειρές και στηλες. Η κάθε στήλη αναφέρεται στα δωμάτια και η κάθε σειρά στις διαδοχικές ημέρες που έχουμε επιλέξει.



Εικόνα 24 – Custom View Form 2

Ο τρόπος που το πρόγραμμα επιτυγχάνει αυτό είναι ακριβώς ο ίδιος με το view day με τις εξής διαφορές:

1) Το array of Buttons (myRoomsButton) που δημιουργείτε είναι πλέον δισδιάστατο για να μπορεί να περιέχει διαφορετικά κουμπιά ανάλογα με την ημερομηνία.
2) Τα κουμπιά αν αναφέρονται σε αριθμό ημερών απο 7 εως 14 τότε μικραίνουν ανάλογα. Για λιγότερες μέρες απο 7 έχουν το standard μέγεθος. Αλλιώς απο τις 7 εώς τις 14 μικραίνουν αναλογικά μέχρι το μισό. Πέραν των 14 ημερών δεν μικραίνουν άλλο και απλά ο χρηστης έχει ΄τη δυνατότητα να κάνει scroll οριζοντίων για να εμφανιστούν όλα τα αποτελέσματα. Η function SizeFactor παρακάτω μας δείχνει τη διαδικασία:

```
Function SizeFactor()
        Dim x As Single
        If CountDays > 7 And CountDays <= 14 Then
            x = 7 / CountDays
            Return x
        ElseIf CountDays > 14 Then
            x = 7 / 14
            Return x
        Else
            Return 1
        End If
    End Function
```

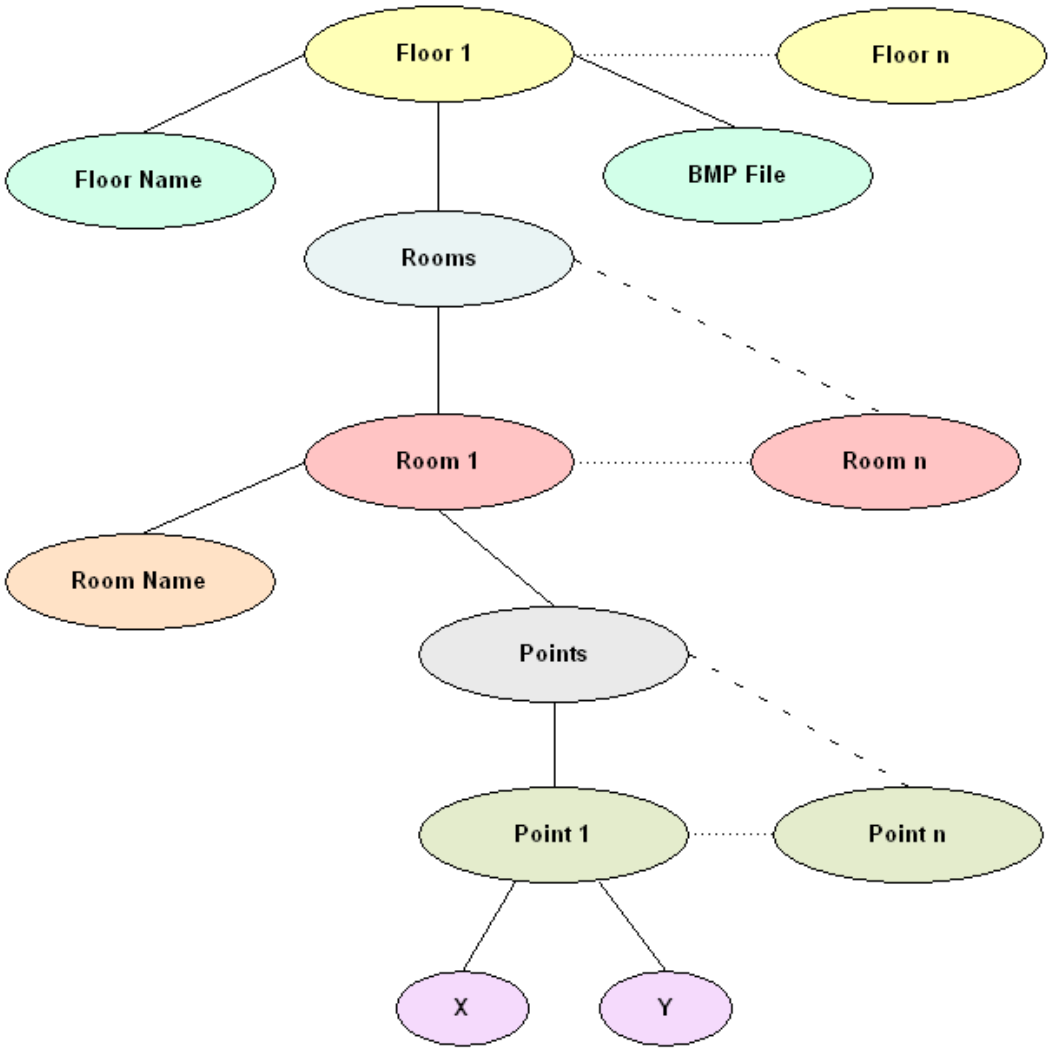
3.2.1.8 Graphical View

Οπως προαναφέραμε, στη φόρμα Select Rooms υπάρχει το κουμπί 'View Graphical' Πατώντας το, μας ανοίγει η Graphical View Form (Εικόνα 13) η οποία οπως είπαμε παρουσιάζει σε ψευδοτρισδιάστατη μορφή όλα τα δωμάτια του ξενοδοχείου με χρωματισμό ανάλογα με την κατάστασή τους. Η συγκεκριμένη φόρμα μας επέτρεψε να εξερευνήσουμε καινούριες τεχνικές δομησης δεδομένων όπως είναι η XML.

Η επιλογή μας να χρησιμοποιήσουμε την XML εδώ δεν ήταν επιτακτική εφόσον τα ίδια δεδομένα θα μπορούσαν να υπάρχουν στη βάση μας. Παρ'όλα αυτά, χρησιμοποιήθηκε αυτή η προσέγγιση για να πάρουμε μια γεύση από τη χρήση αυτής της τόσο συζητημένης τεχνολογίας και επίσης επειδή τα συγκεκριμένα δεδομένα χαρακτηρίζονται βραχυπρόθεσμα στατικά (δεν προσθέτει το ξενοδοχείο δωματια κάθε μερα...) θεωρήσαμε ότι στην περιπτωση που χρειαζόταν αλλαγές στη δομή των δωματίων θα ήταν πολύ πιο εύκολο να προσφέρουμε την υπηρεσία των updates σε ενα απλό XML αρχείο αντί να στέλναμε κάποιο updater της βάσης.

Η φόρμα λειτουργεί με τον εξής τρόπο:

Το σύστημα γνωρίζει ποιά δωμάτια είναι ελεύθερα απο το query της προηγούμενης φόρμας (select Rooms Form). Επίσης γνωρίζει και την απεικόνιση του κάθε ορόφου αφού αυτή εμπεριέχεται σε standard bitmap αρχεία. Το πρόβλημα ήταν να βρούμε κάποιο τρόπο με τον οποίο να μπορούμε να περιγράψουμε ποιό δωμάτιο βρίσκεται πού, και σε ποία τοποθεσία πάνω στο bitmap. Αυτο το επιτύχαμε περνώντας τα χωροταξικά στοιχεία όλων των δωματίων σε ενα αρχείο XML που ονομάσαμε GraphicalData.xml. Το αρχείο έχει την εξής δομή:



Σχήμα 4 – Δομή του αρχείου GraphicalData.xml σε γραφική μορφή

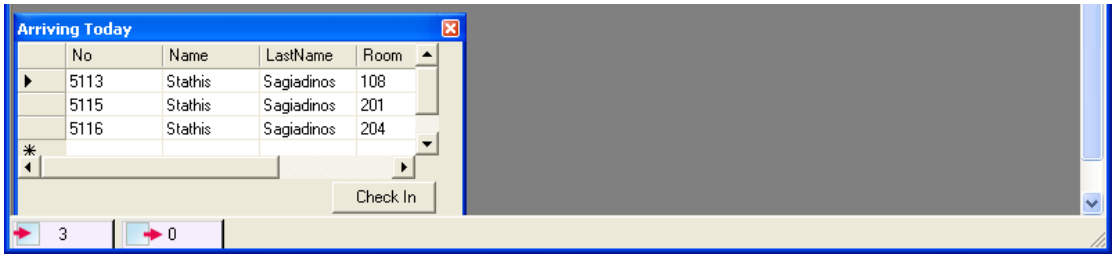
Οπότε για τον κάθε όροφο μπορούμε να έχουμε δωμάτια (Rooms), την γραφική απεικόνιση (bmp file), το όνομα του ορόφου. Για κάθε δωμάτιο (Room) έχουμε το

όνομά του (RoomName) και τις συντεταγμένες (Points) του πάνω στο αρχείο bitmap. Για κάθε Point φυσικά έχουμε μια τιμή X και Y.
Πρώτα, διαβάζοντας το XML αρχείο, το προγραμμά μας σχεδιάζει πάνω στη φόρμα τόσα tabs όσοι είναι οι ορόφοι. Ύστερα φορτώνει στο background του κάθε tab το αντίστοιχο bitmap αρχείο. Τέλος φτιάχνει ένα array of buttons για όλα τα δωμάτια και τα σχεδιάζει στη φόρμα χρησιμοποιώντας polygon paths βάση των points(X, Y). Τέλος, γνωρίζοντας την κατάσταση των δωματίων, με ένα if then else τα χρωματίζει ανάλογα. Οπότε εάν το δωμάτιο είναι ήδη επιλεγμένο από την προηγούμενη φόρμα τότε χρωματίζεται με μπλέ χρώμα. Εάν είναι κλεισμένο χρωματίζεται με κόκκινο και εάν είναι ελεύθερο τότε παίρνει πράσινο χρώμα. Επιλέγοντας οποιοδήποτε ελεύθερο δωμάτιο (πράσινο) η επιλογή μας κατευθείαν αντανακλάται στο datagrid της προηγούμενης φόρμας (select Rooms Form).

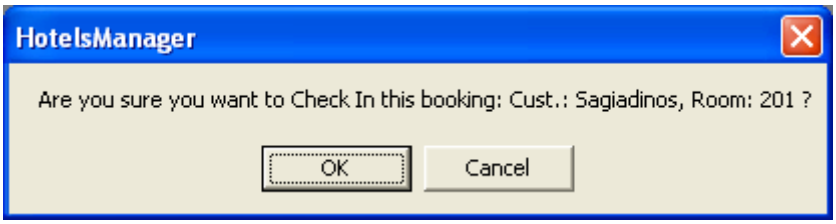
Να σημειώσουμε ότι μπορούμε να δούμε τα δωμάτια σε Graphical View χωρίς να χρειάζεται να κάνουμε κράτηση. Έχουμε τη δυνατότητα να επιλέξουμε από την Menu bar της Main Form, Rooms → View Graphical (Εικόνα 20) και να εμφανίσουμε τη Graphical View, με τη μόνη διαφορά ότι εδώ δεν υπάρχει καμία λειτουγία όπως αυτή που περιγράφεται παραπάνω. Αυτη η δυνατότητα υπάρχει μόνο για να μας πληροφορεί για την τοποθεσία των δωματίων στο ξενοδοχείο (έτσι ωστε πχ. Να γνωρίζουμε αν το δωμάτιο βλέπει ανατολικά κλπ)

3.2.1.9 Arriving Today – Check In Procedure

Στην κεντρική φόρμα (Main Form) στο status bar βλέπουμε ότι υπάρχουν 2 κουμπιά. Το ένα μας δείχνει εαν και πόσοι πελάτες έχουν άφιξη σήμερα και το άλλο μας δείχνει εάν και πόσοι αναχωρούν. Πατώντας πάνω στο αριστερό κουμπί ανοίγει η φόρμα 'Arriving Today' η οποία μας εμφανίζει σε ένα datagrid τις αφίξεις που περιμένουμε. Μόλις ο πελάτης ερθει στο ξενοδοχείο, ο χρήστης απλά τον επιλέγει και γίνεται το check-in πατώντας το κουμπί στο κάτω μέρος. Η αλλαγή αντανακλάται με ένα check-in flag στη βάση δεδομένων στον πίνακα Reservation.



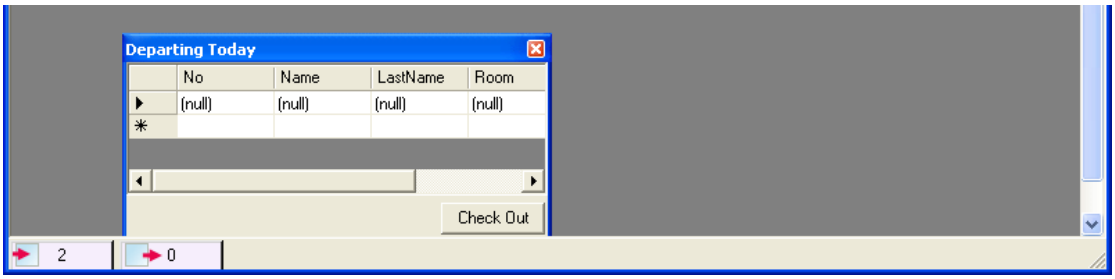
Εικόνα 25 – Arriving Today για Check In



Εικόνα 26 – Επιβεβαίωση για το Check In

3.2.1.10 Departing Today – Check Out Procedure

Παρομοίως με το check-in, πατώντας στο δεύτερο κουμπί της status bar, ο χρήστης μπορεί να κάνει check-out τους πελάτες που φεύγουν. Έτσι, το σύστημα προσφέρει σημαντική ευκολία για τις τρέχουσες διαδικασίες check-in – check-out χωρίς ο χρήστης να χρειάζεται κάθε φορά να ανατρέχει σε αναζήτηση booking.



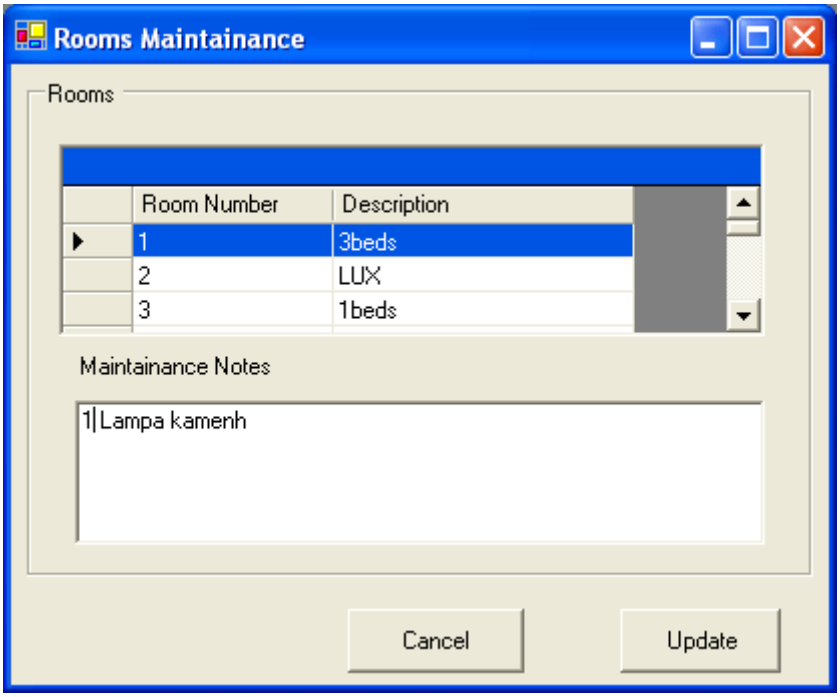
Εικόνα 27 – Departing Today για Check Out

3.2.1.11 Rooms Maintainance (Rooms To Do)

Στο σύστημα, υπάρχει η πρόνοια οι χρήστες να μπορούν να σημειώνουν Notes σχετικά με τα δωμάτια. Ετσι, ανα πάσα στιγμή είναι σε θέση να γνωρίζουν αν κάποιο δωμάτιο έχει ελλείψεις ή χρειάζεται επισκευές.

Εικόνα 28 – Επιλέγουμε από Rooms το Rooms Maintainance

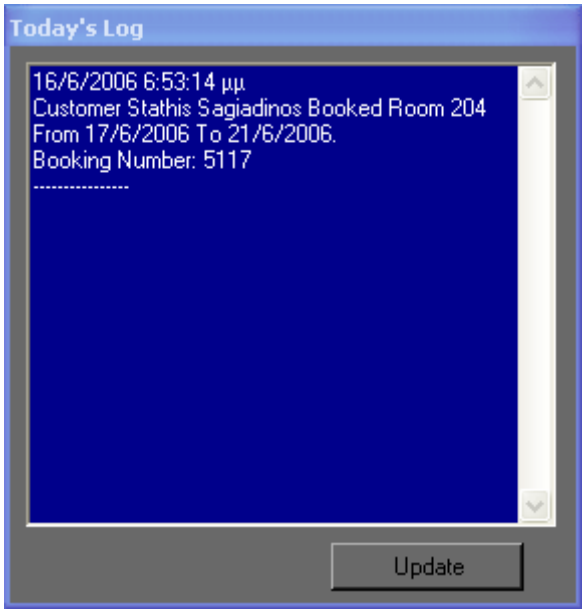
Απο το menu bar, επιλέγοντας Rooms → Rooms Maintainance μας ανοιγει η RoomsToDo Form (Εικόνα 29) στην οποία υπάρχει ένα datagrid με όλα τα δωμάτια του ξενοδοχείου καθώς και ενα textbox απο κάτω. Επιλέγοντας κάποιο δωμάτιο, ο χρήστης έχει τη δυνατότητα να εισάγει Notes στο textbox. Πατώντας το κουμπί Update, η βάση δεδομένων ενημερώνεται. Κατόπιν οι αλλαγές στα Notes είναι διαθέσιμες και απο τη φόρμα Rooms Maintainance αλλά και απο τις φόρμες View Day και View Custom. Έτσι, υπάρχει ιδιαίτερη ευκολία στην πληροφόρηση για το αν ενα δωμάτιο χρειάζεται κατι.



Εικόνα 29 – RoomsToDo Form (Rooms Maintainance)

3.2.1.12 Log Form

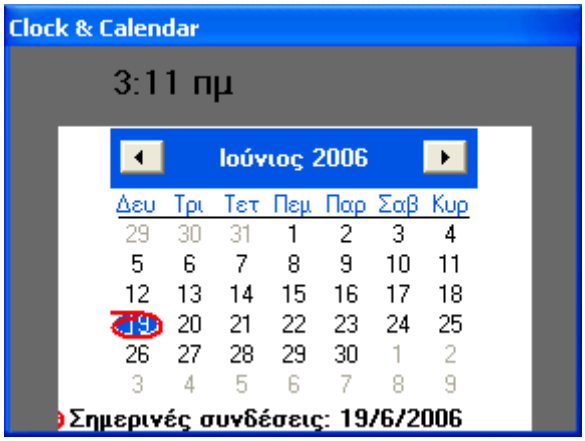
Εντός της κεντρικής φόρμας της εφαρμογής μας, υπάρχει η Log form η οποία εμφανίζει το Log της ημέρας. Όπως προαναφέρθηκε, οι διεργασίες που σχετίζονται με την πραγματοποίηση κάποιας κράτησης ή της διαγραφής της, καταγράφονται στο Table 'Log' της βάσης δεδομένων. Η Log form εμπεριέχει ένα timer control το οποίο κάθε 1 λεπτό διαβάζει τις σημερινές εγγραφές Log και τα παρουσιάζει στο textbox της. Έτσι ανα λεπτό έχει τη δυνατότητα να γνωρίζει τα τελευταία bookings ή booking cancellations από όλα τα συστήματα. Επίσης, σε τοπικό επίπεδο, ανανεώνεται κάθε φορα που κανουμε confirm ή delete κάποιο booking. Επιπλέον υπάρχει και το κουμπί update το οποίο κάνει χειροκίνητα update τα δεδομένα. Έτσι, μπορεί ανα πάσα στιγμή ο ενδιαφερόμενος (πχ. Ο διευθυντής) να παρακολουθεί την πορεία των κρατήσεων κάθε στιγμή της ημέρας.



Εικόνα 30 – Log Form

3.2.1.13 Calendar Form

Εντός της κεντρικής φόρμας υπάρχει επίσης η TimeCalendar Form, η οποία απλώς εμπεριέχει ένα control ημερολογίου το οποίο είναι χρήσιμο να υπάρχει.



Εικόνα 31 – Time Calendar Form

Κεφάλαιο 4

4. Ανακεφαλαίωση και Μελλοντικές Προσθήκες Βελτιώσεις

Στα προηγούμενα κεφάλαια έγινε εκτενής αναφορά στο θεωρητικό κομμάτι διαφόρων τεχνολογιών οι οποίες χρησιμοποιήθηκαν στην παρούσα πτυχιακή εργασία.. Πιο συγκεκριμένα, εγινε αναφορά στις τεχνολογίες VB .net, Object Orientation, ADO .NET, XML, SQL Server. Επίσης παρουσιάσαμε αναλυτικά την εφαρμογή με εικόνες (snapshots), όπου χρειάστηκε. Έγινε πλήρης περιγραφή της λειτουργίας του προγράμματός και παρατέθησαν αξιοσημείωτα κομμάτια κώδικα κάθως και σημαντικά queries.

Με την παρούσα προσέγγιση, μας δόθηκε η ευκαιρία να μελετήσουμε τις παραπάνω τεχνολογίες στο πλαίσιο ενός ολοκληρωμένου project και αποκτήσαμε σημαντική εμπειρία στην πορεία. Παρ' όλα αυτά, αυτή η εργασία δεν αποτελεί παρα μόνο την αρχική έκδοση της εφαρμογής Hotels Manager. Έχουμε επισημάνει σημαντικές βελτιώσεις και διορθώσεις οι οποίες προορίζονται για την επόμενη έκδοση και περιγράφονται παρακάτω:

1) Δυνατότητα προσωρινού κλειδώματος δωματίου κατά τη διαρκεια του Booking ωστε να αποφευχθεί η πιθανότητα διπλοκράτησης στη βάση δεδομένων.
2) Επαλήθευση διαθεσιμότητας δωματίου ύστερα απο την επιλογή δωματίου για τη σπάνια περίπτωση 2 χρήστες να ξεκινήσουν την διαδικασία επιλογής δωματίων ταυτόχρονα και παράλληλα να τύχει επιλέξουν το ίδιο δωμάτιο.
3) Δυνατότητα αποικόνισης Logs για καθοριζόμενο από το χρήστη χρονικό διάστημα.
4) Στατιστικές σχετικά με της κρατήσεις, τα δωμάτια που προτιμούν οι πελάτες, τις ημερομηνίες, ποσοστά πληρότητας ανα περιόδους κλπ.
5) Συμπληρωματική εφαρμογή μαζικής αποστολής e-mail ενημερωτικού και διαφημιστικού χαρακτήρα (προσφορές κλπ) σε όσους επιθυμούν.
6) Αναπτυξή XML web service ωστε να μπορεί και το σύστημα online-κρατήσεων να απεικονίζει γραφικά τα δωμάτιά μας.

7) Έκδοση για Access-Jet database για μικρά ξενοδοχεία με 1 μόνο σύστημα για μείωση του κόστους εγκατάστασης και συντήρησης βάσης δεδομένων.

Βιβλιογραφία – Πηγές (Reference)

1. **ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΜΕ ΘΕΜΑ: «Εφαρμογή Υποστήριξης Κρατήσεων σε Ξενοδοχεία από πελάτες μέσω Διαδικτύου», Σπουδαστές: Κατσίλης Ιωάννης – Σανταμούρης Μιχαήλ**
2. **Visual Basic .NET Προγραμματισμός Δεύτερη Έκδοση, by H. M. Deitel, P. J. Deitel, T. R. Nieto, Εκδότης: M. Γκιούρδας**
3. **Beginning VB.NET 2003, by Thearon Wills, Jonathan Crossland, Richard Blair, Published by Wiley Publishing, Inc.**
4. http://www.devarticles.com/c/a/ADO.NET/Introduction-to-.NET/ 22/5/2006
5. Karl Moore, http://www.developer.com/net/vb/article.php/1540311 22/5/2006
6. http://samples.gotdotnet.com/quickstart/howto/doc/adoplus/ADOPlusOverview.aspx 22/5/2006 **Copyright 2002 Microsoft Corporation**
7. **Essential Skills for Visual Basic .NET written for Visual Basic .NET Complete, by Matt Tagliaferri**
8. **ADO and ADO .NET Programming, by Mike Gunderloy**
9. http://www.developer.com/net/csharp/article.php/10918_1465681_1 24/5/2006**, By Yan Locas and Erik Renaud**
10. http://en.wikipedia.org/wiki/Object-oriented_programming#Fundamental_concepts 29/5/2006
11. http://www.courses.vcu.edu/INFO258-gs/object_orientation1.htm 29/5/2006
12. http://www.prestwood.com/aspsuite/kb/document_view.asp?qid=100137 29/5/2006
13. http://www.softwareag.com/xml/about/xml_ben.htm 10/6/2006
14. **Mastering Visual Basic .NET, by Evangelos Petroutsos**
15. **Visual Basic .NET Developer's Handbook, by Evangelos Petroutsos and Kevin Hough**