ΑΤΕΙ ΗΡΑΚΛΕΙΟΥ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΠΟΛΥΜΕΣΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΠΤΥΞΗ ΠΛΑΤΦΟΡΜΑΣ ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ

Σπουδαστής:
Ρουμελιώτης Αθανάσιος

Επιβλέπων Καθηγητής:
Ιωάννης Κοπανάκης, Επίκουρος καθηγητής

Ηράκλειο Οκτώβριος 2006

Ευχαριστίες

Πριν και πάνω απο όλους θα ήθελα να ευχαριστήσω τους γονείς μου Νίκο και Παναγιώτα, για την αμέριστη συμπαράστασή τους κατά την διάρκεια των σπουδών μου.

Ευχαριστώ ακόμα τον καθηγητή μου Ιωάννη Κοπανάκη που μοιράστηκε τις γνώσεις του μαζί μου, καθώς και για τις χρησιμες συμβουλές και την συνεχή υποστήριξη του.

Τέλος πολλές ευχαριστίες χρωστάω σε όλους τους καθηγητές μου όλα αυτά τα χρόνια, καθώς και σε όλους τους συναδέλφους και φίλους που με βοήθησαν στην πορεία μου με οποιονδήποτε τρόπο.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 ΕΙΣΑΓΩΓΗ 4

1.1 ΕΙΣΑΓΩΓΗ 4
1.2 ΗΛΕΚΤΡΟΝΙΚΟ ΕΜΠΟΡΙΟ 5
1.3 ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΩΝ ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ 6
1.4 ΔΟΜΗ 7

2 STATE OF THE ART 8

2.1 ΤΕΧΝΟΛΟΓΙΕΣ ΑΝΑΠΤΥΞΗΣ ΠΛΑΤΦΟΡΜΩΝ ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ 8
2.2 ΤΕΧΝΟΛΟΓΙΕΣ ΑΣΦΑΛΕΙΑΣ ΠΛΑΤΦΟΡΜΩΝ ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ 11
2.3 ΤΕΧΝΟΛΟΓΙΕΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ 17

3 ΑΝΑΛΥΣΗ ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΑΝΑΠΤΥΞΗ ΠΛΑΤΦΟΡΜΑΣ ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ 20

3.1 ΑΝΑΛΥΣΗ - ΣΧΕΔΙΑΣΜΟΣ ΕΡΓΟΥ 20
3.2 ΣΧΕΔΙΑΣΜΟΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ 27
3.3 ΑΝΑΠΤΥΞΗ ΠΛΑΤΦΟΡΜΑΣ 29

4 ΑΞΙΟΛΟΓΗΣΗ 46

5 ΣΥΜΠΕΡΑΣΜΑΤΑ 48

6 ΒΙΒΛΙΟΓΡΑΦΙΑ 50

1 ΕΙΣΑΓΩΓΗ

1.1 Εισαγωγή

Ο όρος Ηλεκτρονικό Εμπόριο (ΗΕ) χρησιμοποιείται για κάθε είδος εμπορικής συναλλαγής μεταξύ προσώπων (φυσικών και μη) που πραγματοποιείται με ηλεκτρονικά μέσα. Είναι η διάθεση και αγοραπωλησία προϊόντων ηλεκτρονικά, η διεκπεραίωση εμπορικών λειτουργιών και συναλλαγών χωρίς τη χρήση χαρτιού, συνήθως μέσω δικτύων ηλεκτρονικών υπολογιστών. Πρόκειται για την αγοραπωλησία αγαθών, πληροφοριών και υπηρεσιών μέσα από οποιοδήποτε δίκτυο (ηλεκτρονικών υπολογιστών κινητής τηλεφωνίας κ.α).

Το ηλεκτρονικό εμπόριο μπορεί να οριστεί από τέσσερις διαφορετικές οπτικές γωνίες:

Επιχειρήσεις: Ως εφαρμογή νέων τεχνολογιών προς την κατεύθυνση του αυτοματισμού των συναλλαγών και της ροής εργασιών.

Υπηρεσίες: Ως μηχανισμός που έχει στόχο να ικανοποιήσει την κοινή επιθυμία προμηθευτών και πελατών για καλύτερη ποιότητα υπηρεσιών, μεγαλύτερη ταχύτητα εκτέλεσης συναλλαγών και μικρότερο κόστος.

Απόσταση: Ως δυνατότητα αγοραπωλησίας προϊόντων και υπηρεσιών μέσω του Internet ανεξάρτητα από τη γεωγραφική απόσταση.

Επικοινωνία: Ως δυνατότητα παροχής πληροφοριών, προϊόντων ή υπηρεσιών, και πληρωμών μέσα από δίκτυα ηλεκτρονικών υπολογιστών.

Το ηλεκτρονικό εμπόριο σε πρακτικό επίπεδο, μπορεί να πάρει πολλές μορφές:

Εσωτερικό εμπόριο: Στόχος είναι η αποτελεσματικότερη λειτουργία των δραστηριοτήτων μιας επιχείρησης, ώστε να μπορεί να προσφέρει καλύτερα προϊόντα και υπηρεσίες στους πελάτες της. Οι εφαρμογές του συνήθως εντάσσονται στη λειτουργία ενός τοπικού δικτύου (Intranet) και μπορούν να είναι: επικοινωνία μεταξύ ομάδων εργασίας,

ηλεκτρονική δημοσίευση (άμεση διανομή πληροφοριών) κτλ. Μέρος αυτού του τομέα είναι οι εφαρμογές Web-EDI (Web Electronic Data Interchange) που αποσκοπούν στην ανάπτυξη της αποτελεσματικότητας της επιχείρησης και στο μειωμένο κόστος, καθώς και στην αυτοματοποίηση της διαδικασίας ανταλλαγής πληροφοριών που σχετίζονται με παραγγελίες, τιμολόγια και αποστολές.

Συναλλαγές μεταξύ επιχειρήσεων (Business-to-Business - B2B): Το ηλεκτρονικό εμπόριο επιτρέπει σε επιχειρήσεις να βελτιώσουν τη μεταξύ τους συνεργασία, απλοποιώντας τις διαδικασίες και το κόστος των προμηθειών, την ταχύτερη αποστολή των προμηθειών και τον αποτελεσματικότερο έλεγχο του επιπέδου αποθεμάτων. Επιπλέον καθιστά ευκολότερη την αρχειοθέτηση των σχετικών εγγράφων και ποιοτικότερη την εξυπηρέτηση πελατών. Η δυνατότητα ηλεκτρονικής σύνδεσης με προμηθευτές και διανομείς καθώς και η πραγματοποίηση ηλεκτρονικών πληρωμών βελτιώνουν ακόμη περισσότερο την αποτελεσματικότητα: οι ηλεκτρονικές πληρωμές περιορίζουν το ανθρώπινο σφάλμα, αυξάνουν την ταχύτητα και μειώνουν το κόστος των συναλλαγών. Το ηλεκτρονικό εμπόριο προσφέρει τη δυνατότητα αυξημένης πληροφόρησης σχετικά με τα προσφερόμενα προϊόντα - είτε από τους προμηθευτές είτε από ενδιάμεσους οργανισμούς που προσφέρουν υπηρεσίες ηλεκτρονικού εμπορίου.

Λιανικές πωλήσεις - Ηλεκτρονικό εμπόριο μεταξύ επιχείρησης και καταναλωτών (Business-to-Consumer - B2C): Πρόκειται για την πιο διαδεδομένη μορφή ηλεκτρονικού εμπορίου. Ο καταναλωτής έχει πρόσβαση σε μια τεράστια ποικιλία προϊόντων σε δικτυακούς κόμβους-καταστήματα, βλέπει, επιλέγει, αν επιθυμεί να αγοράσει είδη ένδυσης μπορεί ενίοτε και να τα δοκιμάζει (μέσω ειδικών προγραμμάτων), ανακαλύπτει προϊόντα τα οποία δεν θα μπορούσε να βρει εύκολα στη χώρα του, συγκρίνει τιμές και τέλος αγοράζει. Κι όλα αυτά χωρίς να βγει από το σπίτι του, κερδίζοντας πολύτιμο χρόνο και κόπο.

1.2 Ηλεκτρονικό Εμπόριο

Ως ηλεκτρονικό εμπόριο ορίζεται το εμπόριο που πραγματοποιείται με ηλεκτρονικά μέσα βασίζεται δηλαδή στην ηλεκτρονική μετάδοση δεδομένων. Το ηλεκτρονικό εμπόριο αποτελεί έκφανση των λεγόμενων υπηρεσιών εξ αποστάσεως (ΠΔ 39.2001) .

Ηλεκτρονικό εμπόριο αποτελεί μια ολοκληρωμένη συναλλαγή που πραγματοποιείται μέσω του διαδικτύου χωρίς να είναι απαραίτητη η φυσική παρουσία των συμβαλλομένων μερών, δηλαδή του πωλητή και του αγοραστή, οι οποίοι μπορούν να βρίσκονται ακόμα και σε διαφορετικές χώρες. Είναι οποιαδήποτε συναλλαγή που ενέχει διαδικτυακή δέσμευση για αγορά η πώληση αγαθών ή υπηρεσιών.

Ηλεκτρονικό εμπόριο θεωρούνται επίσης και οι συναλλαγές μέσω τηλεφώνου και φαξ. Το ηλεκτρονικό εμπόριο διακρίνεται σε έμμεσο και άμεσο. Ο πρώτος όρος χρησιμοποιείται όταν πρόκειται για την ηλεκτρονική παραγγελία υλικών αγαθών που μπορούν να παραδοθούν μόνο με παραδοσιακούς τρόπους όπως είναι το ταχυδρομείο.

Άμεσο είναι το ηλεκτρονικό εμπόριο που περιλαμβάνει παραγγελία, πληρωμή και παράδοση άυλων αγαθών και υπηρεσιών. Η πληρωμή των υπηρεσιών αυτών γίνεται είτε με πιστωτικές κάρτες είτε με ηλεκτρονικό χρήμα με την αρωγή πάντα και τη σύμπραξη των τραπεζών.

1.3 Ανάπτυξη εφαρμογών Ηλεκτρονικού Εμπορίου

Κατά την ανάπτυξη εφαρμογών Ηλεκτρονικού Εμπορίου οι σχεδιαστές και οι προγραμματιστές αντιμετωπίζουν μια σειρά από προβλήματα στα οποία καλούνται να αντεπεξέλθουν, ώστε η εφαρμογή να είναι αξιόπιστη και εύχρηστη. Τα πιο βασικά από αυτά είναι τα παρακάτω:

Κληροδοτημένη τεχνολογία: Ένα πρόβλημα που προκύπτει λόγω του αρχικού σχεδιασμού των διακομιστών είναι ότι αρχικά οι ιστοσελίδες προορίζονταν να είναι στατικές: ήταν αρχεία αποθηκευμένα σ' έναν υπολογιστή, που παραδίδονταν, μέσω ενός browser, στους χρήστες στην αποθηκευμένη τους μορφή. Πολλές από τις εφαρμογές ηλεκτρονικού εμπορίου και επιχειρηματικότητας απαιτούν κάτι πιο δυναμικό, π.χ. υπάρχει πληθώρα ιστοχώρων οικονομικών υπηρεσιών στο Web που παρέχουν στους πελάτες τους ολοκαίνουργιες υπηρεσίες και χαμηλές τιμές. Οι τιμές αυτές αποθηκεύονται σε ιστοσελίδες και χρειάζονται συχνή ανανέωση, συχνά κάθε λίγα δευτερόλεπτα.

Ασφάλεια: Από τη στιγμή που το Internet είναι ένα ανοιχτό σύστημα, όλοι έχουν πρόσβαση στις τεχνολογίες που το υποστηρίζουν. Αυτό σημαίνει ότι τα δεδομένα διαδίδονται μέσα στο Internet δημόσια. Οι συνέπειες αυτού είναι ότι, θεωρητικά, ο

καθένας που έχει τα σωστά εργαλεία μπορεί να «κρυφοκοιτάξει» στα δεδομένα που μεταβιβάζονται από τον ένα υπολογιστή στον άλλο μέσω του Internet. Βασική προϋποθεση για μία αξιόπιστη εφαρμογή είναι η εξασφάλισή της από κακόβουλες ενέργειες καθώς και η ασφαλής μεταφορά των δεδομένων ανάμεσα στους χρήστες που χρησιμοποιούν την εφαρμογή.

Ταχύτητα Ανάπτυξης: Οι εμπειρογνώμονες του ηλεκτρονικού εμπορίου μιλούν για έτος Web. Πρόκειται για τον χρόνο που χρειάζεται για να τεθεί σε εφαρμογή ένα συμβατικό σύστημα, που, υπό κανονικές συνθήκες, θα χρειαζόταν ένα ημερολογιακό έτος για να αναπτυχθεί. Σύμφωνα με τους τρέχοντες υπολογισμούς ένα ημερολογιακό έτος ισοδυναμεί με επτά έτη Web. Για τις εταιρείες, πουθενά αλλού δεν είναι πιο επιτακτική η ανάγκη για γρήγορη ανάπτυξη προϊόντων και υπηρεσιών, μαζί με την απαιτούμενη ηλεκτρονική υποδομή, παρά στο ηλεκτρονικό εμπόριο. Από την πλευρά της μηχανικής λογισμικού, η ανάγκη αυτή έδωσε ώθηση στην ανάπτυξη μίας σειράς μεθόδων λογισμικού που σε γενικές γραμμές περιγράφονται από τον όρο ταχεία ανάπτυξη εφαρμογών (rapid application development). Από την πλευρά της τεχνολογίας, έδωσε ώθηση σε μία σειρά ιδεών με σκοπό την δημιουργία εργαλείων που θα επιτρέψουν στις εταιρείες να αναπτύξουν συστήματα με την συνένωση, και μόνο, διαφόρων εξαρτημάτων, πολλά από τα οποία θα ξεχωρίζουν με τη χρήση σχεδιαστικών περιγραμμάτων.

1.4 Δομή

Στα επόμενα κεφάλαια θα καταγράψουμε τις ήδη υπάρχουσες τεχνολογίες που αφορούν στην ανάπτυξη εφαρμογών ηλεκτρονικού εμπορίου. Ακόμα θα αναλύσουμε την πρόταση μας, για την δημιουργία μιας εφαρμογής Web-EDI(Web Electronic Data Interchange) που αποσκοπεί στην ανάπτυξη της αποτελεσματικότητας της επιχείρησης στο μειωμένο κόστος, στην αυτοματοποίηση της διαδικασίας ανταλλαγής πληροφοριών που σχετίζονται με παραγγελίες, τιμολόγια και αποστολές καθώς και στην βελτιστοποίηση των εμπορικών σχέσεων της επιχείρησης. Θα αναφερθούμε στα προγράμματα που θα χρησιμοποιήσουμε για την ανάπτυξη της εφαρμογής. Τέλος θα αξιολογήσουμε την εφαρμογή μας και θα καταγράψουμε τα συμπεράσματά μας.

2 STATE OF THE ART

2.1 Τεχνολογίες ανάπτυξης πλατφόρμων Ηλεκτρονικού Εμπορίου

Τα δεδομένα (data) σε οποιαδήποτε μορφή αποτελούν τη ζωογόνο δύναμη της σύγχρονης επιχείρησης. Από ένα σύστημα διαχείρισης των πελατειακών σχέσεων μιας επιχείρησης (CRM,), μέχρι ένα σύστημα ERP ή ένα σύστημα διαχείρισης περιεχομένου, όλα χρησιμοποιούν διάφορα δεδομένα για να εκπληρώσουν τις εργασίες για τις οποίες έχουν προγραμματιστεί.

Εξ ορισμού, τα συστήματα αυτά αδυνατούν να επικοινωνήσουν μεταξύ τους ή -ακόμα και αν τελικά αυτό συμβεί- επιτυγχάνεται με σημαντικό κόστος σε χρόνο και χρήμα. Η δε επικοινωνία που επιτυγχάνεται είναι πολλές φορές "εύθραυστη" και μη ολοκληρωμένη. Κατά συνέπεια, βασικό ζητούμενο αναδεικνύεται η διαλειτουργικότητα, η δυνατότητα δηλ. διαφορετικά συστήματα να χρησιμοποιούν κοινά πρωτόκολλα επικοινωνίας.

Η τεχνολογία Microsoft .net είναι το βασικό συστατικό της πλατφόρμας λογισμικού της Microsoft, για την ασφαλή ανάπτυξη, προμήθεια και χρήση διαλειτουργικών εφαρμογών, τόσο σε ανεξάρτητους υπολογιστές όσο και σε δίκτυα υπολογιστών (Internet, Intranet, κ.λπ.). Διαλειτουργικές εφαρμογές ονομάζονται εκείνες οι εφαρμογές που εδραιώνουν την επικοινωνία και την πλήρη ενοποίηση διαφορετικών συστημάτων, εκ των οποίων τουλάχιστον ένα είναι λογισμικό της Microsoft. Για παράδειγμα, τα Windows ή οι εφαρμογές του Office, όπως το Excel, μπορούν να χρησιμοποιήσουν την τεχνολογία .net για να συνδεθούν με άλλα συστήματα και εφαρμογές, όπως με ένα σύστημα CRM, ή ERP ή ακόμα και με κάποιο άλλο λειτουργικό σύστημα, όπως με το Linux ή το Unix. Οι εφαρμογές τέτοιου τύπου καλούνται συχνά και smart client εφαρμογές.
Η τεχνολογία .net είναι στην πραγματικότητα μία δέσμη (σετ) λύσεων λογισμικού, στις οποίες κομβική θέση κατέχουν το **Microsoft .net Framework** και το **Microsoft Visual**

Studio .net.

Χωρίς να υπεισέλθουμε σε τεχνικές λεπτομέρειες, το Microsoft .net Framework περιλαμβάνει τις λύσεις λογισμικού που θα επιτρέψουν την επικοινωνία χρήστη και εφαρμογών, ενώ το Microsoft Visual Studio .net. προσφέρει τα εργαλεία για τη δημιουργία αυτών των εφαρμογών. Υπάρχουν όμως και άλλα, ιδιαίτερα ενδιαφέροντα στοιχεία.
Το πιο ενδιαφέρον ίσως συστατικό της τεχνολογίας .net είναι τα Web services. Πρόκειται για μικρές εφαρμογές λογισμικού που βασίζονται στα πρωτόκολλα επικοινωνίας XML (Extensible Markup Language), SOAP (Simple Object Access Protocol), και WSDL (Web Services Description Language) και οι οποίες επιτρέπουν την ανταλλαγή ή τη λήψη εφαρμογών (και δεδομένων) ανάμεσα σε υπολογιστές που τρέχουν διαφορετικά λειτουργικά συστήματα και χρησιμοποιούν διαφορετικές γλώσσες προγραμματισμού. Η ανταλλαγή δεδομένων μπορεί να πραγματοποιείται είτε μέσω του Διαδικτύου είτε μέσα σε κάποιο Intranet και μπορεί να πάρει 3 βασικές μορφές:

α) Η ανταλλαγή να γίνεται ανάμεσα σε υπολογιστές που παίζουν το ρόλο του "πελάτη" (client to client)

β) Η ανταλλαγή να γίνεται ανάμεσα στο διακομιστή και τους clients (server to client)

γ) Η ανταλλαγή να γίνεται μεταξύ διακομιστών (server to server)

Έτσι, οι εν λόγω εφαρμογές μπορεί λ.χ. να βρίσκονται σε κάποιο διακομιστή, απ' όπου μπορούν να παραληφθούν, μέσω Internet, και να χρησιμοποιηθούν από κάθε ενδιαφερόμενο. Απλούστερα, τα Web services είναι εφαρμογές λογισμικού που αναπτύσσονται από κάποια εταιρία, "ανεβαίνουν" κάπου στο Διαδίκτυο (σ' ένα διακομιστή), και προσφέρονται για χρήση στις ενδιαφερόμενες επιχειρήσεις. Από την πλευρά τους, οι επιχειρήσεις/πελάτες καταβάλλουν ενδεχομένως κάποια συνδρομή για τη χρήση των εφαρμογών και αποκτούν το δικαίωμα χρήσης. Με το μοντέλο αυτό, το Internet γίνεται ο δίαυλος και ταυτόχρονα η πηγή για την online άντληση εφαρμογών και τεχνολογικών λύσεων και όχι μόνο πληροφοριών.

Συμπερασματικά, η τεχνολογία .net είναι μία "πλατφόρμα διαλειτουργικότητας και ευελιξίας": Διαλειτουργικότητας, γιατί είναι η λογισμική βάση που επιτρέπει την ασφαλή επικοινωνία εφαρμογών και προγραμμάτων, μέσω του Internet ή άλλου είδους δικτύων. Ευελιξίας, γιατί δημιουργεί ένα ψηφιακό περιβάλλον που ενσωματώνει το Διαδίκτυο και τα χαρακτηριστικά του.

Σημειώνεται, ότι η τεχνολογία .net υποστηρίζει όλη την οικογένεια προϊόντων της Microsoft (λειτουργικά περιβάλλοντα, σουίτες γραφείου κ.λπ.), ενώ εκτιμάται ότι στο μέλλον θα διαδραματίσει ακόμη πιο ενεργό ρόλο στα προϊόντα της εταιρίας.

Oracle Fusion Middleware Η τεχνολογία Middleware αποτελεί μία ευέλικτη οικογένεια λύσεων, που προσφέρει στις επιχειρήσεις τη δυνατότητα να αξιοποιήσουν περαιτέρω τις ήδη υπάρχουσες επενδύσεις τους σε λογισμικό, συνενώνοντάς τες και φέρνοντάς τες σε στενή επαφή. Είναι πλήρως παραμετροποιήσιμη και προσαρμόζεται στις ανάγκες της επιχείρησης. Ανάμεσα σε άλλα, προσφέρει πρότυπα ολοκλήρωσης μέσω υπηρεσιών δικτύου, δυνατότητες συνεργασίας διαφόρων μερών, περιβάλλον Java, διαχείριση περιεχομένου, κεντρική διαχείριση και προεπισκόπηση όλων των τεχνικών δεδομένων, διαχείριση δικτυακής πύλης και κόμβου δεδομένων, διαχείριση αρχείων κ.ά.

SAP NetWeaver Το SAP NetWeaver είναι μια ολοκληρωμένη πλατφόρμα ενοποίησης εφαρμογών, που συνεργάζεται με τα ήδη εγκατεστημένα συστήματα πληροφορικής. Παρέχει δυνατότητες σχεδίασης, οικοδόμησης και υλοποίησης νέων επιχειρηματικών διαδικασιών με ταχύτητα και ευελιξία. Η πλατφόρμα SAP NetWeaver υποστηρίζει ανοιχτά διαδικτυακά πρότυπα όπως HTTP, XML και Web services και συνεργάζεται πλήρως με τις τεχνολογίες Microsoft .NET και Java 2 Platform Enterprise Edition (J2EE) σε περιβάλλοντα όπως IBM WebSphere. Ταυτόχρονα, αποτελεί την τεχνολογική υποδομή των λύσεων mySAP Business Suite, των σύνθετων εφαρμογών SAP xApps, λύσεων συνεργατών της SAP και εξειδικευμένων εφαρμογών των πελατών της. Αποτελεί επίσης το θεμέλιο στήριξης της Enterprise Services Architecture, της νέας αρχιτεκτονικής παροχής υπηρεσιών επιχειρησιακού προτύπου, για την υλοποίηση λύσεων προσανατολισμένων στις υπηρεσίες.

Βέβαια υπάρχουν και άλλες τεχνολογίες που χρησιμοποιούνται για την ανάπτυξη πλατφόρμων όπως είναι η κλασσική **asp (active server pages)** η οποία είναι μια τεχνολογία που επιτρέπει την δημιουργία δυναμικών ιστοσελίδων και είναι ο πρόδρομος της asp.net, η **php (Hypertext Preprocessor)** με την οποία μπορείς επίσης να δημιουργήσεις δυναμικές ιστοσελίδες που μπορούν να επικοινωνούν με βάσεις δεδομένων. Πλεονεκτήματά της η ευχέρεια να συνδέεται με πολλούς διαφορετικούς τύπους βάσεων δεδομένων και η υποστήριξη πολλών πρωτοκόλλων για την επικοινωνία μεταξύ

διαφορετικών δικτύων. Μια άλλη τεχνολογία είναι η **ColdFusion** της Macromedia που περιέχει ένα σύνολο εργαλείων για την ενοποίηση μιας βάσης δεδομένων και μιας ιστοσελίδας. Τέλος δεν πρέπει να παραλείψουμε την Java 2 Platform Enterprise Edition (J2EE) της Sun Microsystems και το NetBeans που προσφέρει τα εργαλεία για τη δημιουργία αυτών των εφαρμογών.

2.2 Τεχνολογίες Ασφάλειας πλατφόρμων Ηλεκτρονικού Εμπορίου

Για να υπάρχει ασφάλεια στις συναλλαγές, απαιτείται η παρουσία ενός ασφαλούς web server. Ο ασφαλής web server χρησιμοποιείται για την απόκρυψη δεδομένων μεταξύ ενός server και ενός browser. Τα δεδομένα κρυπτογραφούνται και προς τις δύο κατευθύνσεις, έτσι ώστε να μην μπορεί κάποιος να τα παρακολουθήσει κατά τη μεταφορά τους στο Internet.

Η πρόσβαση μέσω ενός ασφαλούς server είναι σαφώς πιο αργή σε σύγκριση με τη σύνδεση μέσω ενός κοινού server, και αυτό οφείλεται στην κρυπτογράφηση/ αποκρυπτογράφηση η οποία χρειάζεται να γίνει στα δεδομένα. Εξαιτίας αυτού του επιπλέον φόρτου στον web server, η επιλογή της χρήσης του ασφαλούς web server πρέπει να γίνεται μόνο όταν πρόκειται για την προστασία ευαίσθητων δεδομένων. Πριν τη λειτουργία ενός ηλεκτρονικού καταστήματος, θα πρέπει να γίνουν έλεγχοι έτσι ώστε να είναι βέβαιο πως ο αριθμός της πιστωτικής κάρτας του πελάτη ή οποιαδήποτε άλλα ευαίσθητα δεδομένα, είναι επαρκώς προστατευμένα κατά τη μεταφορά τους από τον browser του πελάτη στον server του καταστήματος ή οποιοδήποτε άλλο server με τον οποίο συνεργάζεται το κατάστημα.

Τα απαραίτητα στοιχεία για να υλοποιηθούν τα παραπάνω είναι τα εξής:

Ο web server θα πρέπει να είναι ένας ασφαλής server, ο οποίος προστατεύει τα δεδομένα που στέλνονται από τον web browser του πελάτη (π.χ. μέσω μιας Web φόρμας) στον κεντρικό server κωδικοποιώντας τα.Το URL ενός ασφαλούς server, μοιάζει με τα μέχρι τώρα χρησιμοποιούμενα, αλλά αντί για "HTTP" χαρακτηρίζεται ως "HTTPS" (HTTPSecure).

Ο πελάτης χρειάζεται να έχει έναν από τους δύο διαδεδομένους browsers της αγοράς, είτε τον Netscape Navigator είτε τον Microsoft Internet Explorer, έτσι ώστε να εξασφαλίζεται η μεταβίβαση των δεδομένων από τον πελάτη προς τον server με ασφαλή

τρόπο. Η εμφάνιση των web σελίδων, είναι πανομοιότυπη με αυτή κάθε άλλου web server αλλά με δύο διαφορές: υπάρχει μια μπλε γραμμή κατά μήκος του άνω μέρους του παραθύρου του browser, ενώ το κλειδί (στο Netscape Navigator) ή το λουκέτο (στον Microsoft Internet Explorer) στην κάτω αριστερή γωνία του παραθύρου είναι ενεργοποιημένο. Αυτές οι διαφορές κάνουν φανερό ότι εμφανίζεται μια ασφαλής (secure) σελίδα.

Πρέπει να σημειωθεί εδώ ότι άλλοι browsers, όπως το Mosaic, δεν έχουν πρόσβαση σε URL που έχουν HTTPS. Έτσι, οι πελάτες που επιθυμούν να κάνουν αγορές αγαθών και υπηρεσιών από διάφορα sites τα οποία χρησιμοποιούν ασφαλείς servers, θα πρέπει να προμηθευτούν τους browsers είτε από το site της Netscape είτε από αυτό της Microsoft.

Κρυπτογράφηση και πιστοποίηση αυθεντικότητας Ο φόβος πολλών ατόμων να χρησιμοποιήσουν την πιστωτική τους κάρτα μέσα στο Διαδίκτυο, αλλά και ο φόβος των εταιρειών ότι οι πελάτες τους δεν θα ακολουθήσουν τους όρους των συμβολαίων που έχουν κάνει ηλεκτρονικά, είναι προβλήματα τα οποία λύνονται με τη χρήση της κρυπτογράφησης. Αυτή βασίζεται στο ότι τα στοιχεία τροποποιούνται με βάση κάποιους κανόνες, έτσι ώστε μόνο όποιος ξέρει αυτούς τους κανόνες να μπορεί να τα διαβάσει. Υπάρχουν διάφορα είδη κρυπτογράφησης. Σε όλα αυτά χρησιμοποιούνται ένα ή περισσότερα κλειδιά, τα οποία όσο μεγαλύτερα είναι τόσο περισσότερη ασφάλεια παρέχουν. Στην απλή περίπτωση κατά την οποία κάποιος θέλει να προστατέψει τα δεδομένα που έχει, χρησιμοποιεί ένα κλειδί, μόνο με τη χρήση του οποίου γίνεται δυνατή η ανάγνωση των δεδομένων. Για την ασφαλή επικοινωνία, τόσο ο αποστολέας όσο και ο παραλήπτης πρέπει να γνωρίζουν το κατάλληλο κλειδί. Για να εξασφαλιστεί όμως η ασφαλής μεταφορά αυτού του κλειδιού, γίνεται χρήση δυο κλειδιών.

Ο αποστολέας κωδικοποιεί το μήνυμα με βάση το κλειδί του παραλήπτη, και μόνο ο δεύτερος, ο οποίος ξέρει και το υπόλοιπο μέρος του κλειδιού, μπορεί να διαβάσει το μήνυμα. Ένα άλλο πρόβλημα το οποίο παρουσιάζεται με τη χρήση του Διαδικτύου, είναι η πιστοποίηση της ταυτότητας του αποστολέα. Για αυτόν το σκοπό μπορεί να χρησιμοποιηθεί η κρυπτογράφηση, έτσι ώστε να κωδικοποιηθεί μόνο ένα μέρος του μηνύματος που θα περιέχει την ηλεκτρονική υπογραφή του αποστολέα. Με αυτόν τον τρόπο ο παραλήπτης, ο οποίος θα γνωρίζει την εν λόγω υπογραφή, θα γνωρίζει την προέλευση του μηνύματος. Το μόνο ίσως πρόβλημα όσον αφορά στην κρυπτογράφηση

γενικώς, είναι ότι για την πραγματική ασφάλεια πρέπει να χρησιμοποιηθούν κλειδιά της τάξεως των 1.024 ή 2.048 bits. Κάτι τέτοιο απαιτεί αρκετά μεγάλη υπολογιστική ισχύ, η οποία καθυστερεί τη διαδικασία κρυπτογράφησης. Υπάρχουν δύο διεθνή πρότυπα ασφάλειας για μεταφορά δεδομένων. Αυτά είναι το SSL (SecureSocketsLayer) και το SET (SecureElectronicTransaction). Το πρώτο χρησιμοποιείται διεθνώς για μεταφορά ευαίσθητων οικονομικά δεδομένων, ενώ το δεύτερο, το οποίο αναπτύχθηκε από τη Visa και τη MasterCard και υιοθετήθηκε και από την American Express, χρησιμοποιείται σε περιπτώσεις που υπάρχουν πληρωμές μέσω κάποιας ανάλογης κάρτας.

Η καθιέρωση της κρυπτογράφησης, λοιπόν, έδωσε τη λύση σε όλα τα προβλήματα μετάδοσης ευαίσθητων πληροφοριών. Επίσης, εκτός από την κωδικοποίηση-αποκωδικοποίηση της πληροφορίας, η κρυπτογράφηση, εισάγει και τη διαρκή πιστοποίηση (authentication) του δημιουργού αλλά και του τελικού αποδέκτη της πληροφορίας. Αποτέλεσμα είναι απλές, καθημερινές διαδικασίες, όπως η υπογραφή ενός κειμένου ως σημάδι γνησιότητας, μπορούν να μεταφερθούν μέσω της κρυπτογράφησης στο ηλεκτρονικό τους αντίστοιχο και να επικυρώσουν αποφάσεις, συμφωνίες, κτλ.

Επίπεδο Ασφαλών Συνδέσεων (SSL - Secure Sockets Layer) Το πρωτόκολλο αυτό σχεδιάστηκε προκειμένου να πραγματοποιεί ασφαλή σύνδεση με τον εξυπηρετητή (server). Το SSL χρησιμοποιεί "κλειδί" δημόσιας κρυπτογράφησης, με σκοπό να προστατεύει τα δεδομένα καθώς "ταξιδεύουν" μέσα στο Internet. Το SSL αναπτύχθηκε από την εταιρία Netscape και χρησιμοποιεί τεχνικές δημόσιου κλειδιού στην αρχική επικοινωνία, ώστε να επιτευχθούν οι ακόλουθοι στόχοι:

- Ο εξυπηρετητής δηλώνει την ταυτότητά του μέσω της ψηφιακής υπογραφής του.
- Εξυπηρετητής και εξυπηρετούμενος συμφωνούν στη χρήση ενός συγκεκριμένου κλειδιού/αλγορίθμου, με το οποίο θα κρυπτογραφηθεί το υπόλοιπο της συνομιλίας.

Οι συμμετρικοί αλγόριθμοι κρυπτογράφησης τους οποίους χρησιμοποιεί το SSL είναι συνήθως οι RC2/RC4 για την έκδοση SSL v2, ενώ στην έκδοση SSL v3 υπάρχουν και οι RC4 128bit και Triple DES (όπου αυτοί επιτρέπονται). Η αδυναμία του SSL για τους χρήστες όλων των χωρών εκτός των Ηνωμένων Πολιτειών είναι το μικρό μέγεθος key (40 bits) που χρησιμοποιεί για τη συμμετρική κρυπτογράφηση των δεδομένων. Πέρα από την αντικειμενική ή όχι ανεπάρκεια στη δύναμη της 40-bit κρυπτογράφησης, υπάρχει το δεδομένο ότι για συγκεκριμένες υπηρεσίες υψηλής ασφάλειας και ρίσκου, κυρίως στον

τραπεζικό χώρο, δεν γίνονται αποδεκτοί φυλλομετρητές που δεν προσφέρουν 128-bit κωδικοποίηση.

PGP(Pretty Good Privacy) Το πρόγραμμα PGP είναι ένα δημοφιλές πακέτο λογισμικού κρυπτογράφησης, που αναπτύχθηκε από την εταιρεία Network Associates και το οποίο συνδυάζει συμμετρική και ασύμμετρη κρυπτογράφηση για να παρέχει μεγαλύτερη ασφάλεια. Επίσης πριν εκκινηθεί η διαδικασία κρυπτογράφησης, τα δεδομένα συμπιέζονται.

Για την κρυπτογράφηση/αποκρυπτογράφηση χρησιμοποιούνται δυο κλειδιά, ένα κλειδί συνόδου session key και το δημόσιο κλειδί του παραλήπτη. Το κλειδί συνόδου δημιουργείται με τυχαίο τρόπο με βάση την συμπεριφορά του χρήστη. Στη συνέχεια το αρχείο που πρόκειται να αποσταλεί, κρυπτογραφείται με τη μέθοδο της συμμετρικής κρυπτογράφησης με το κλειδί συνόδου και το κλειδί συνόδου είναι αυτό που κρυπτογραφείται με βάση το δημόσιο κλειδί του παραλήπτη. Κατά την αποστολή του μηνύματος αποστέλλεται επίσης και το κρυπτογραφημένο κλειδί συνόδου.

Στον παραλήπτη ακολουθείται η ακριβώς αντίστροφη διαδικασία. Το κλειδί συνόδου αποκρυπτογραφείται με βάση το ιδιωτικό κλειδί του παραλήπτη, το οποίο στη συνέχεια μπορεί να χρησιμοποιηθεί για να αποκρυπτογραφήσει το αρχικό κείμενο.

Συνήθως για την κρυπτογράφηση του κειμένου χρησιμοποιείται το πρότυπο DES ενώ για το κλειδί συνόδου το πρότυπο RSA . Το σύστημα με αυτό τον τρόπο συνδυάζει ευελιξία και ασφάλεια.

Ασφαλείς Ηλεκτρονικές Συναλλαγές (SET - Secure Electronic Transactions) Έως σήμερα αρκετές επιχειρήσεις χρησιμοποιούν συστήματα ηλεκτρονικών συναλλαγών, τα οποία στις περισσότερες περιπτώσεις είναι ασύμβατα μεταξύ τους, ενώ άλλες απέχουν από το ηλεκτρονικό εμπόριο έως ότου σιγουρευτούν ότι υπάρχει ένα ευρύτερα αποδεκτό και εγγυημένα ασφαλές πρότυπο συναλλαγών. Η πιθανότητα για τη δημιουργία ενός τέτοιου προτύπου άρχισε να καθίσταται σημαντική, από το Φεβρουάριο του 1996, όταν η Visa και η Mastercard αποφάσισαν να προβούν στην από κοινού υλοποίησή του. Το SET (Secure Electronics Transaction), όπως ονομάστηκε το νέο πρότυπο, προήλθε από τη συνεργασία κολοσσών στο χώρο της Πληροφορικής, συμπεριλαμβανομένων των GTE, IBM, Microsoft, Netscape και VeriSign. Ορισμένες εξ αυτών είχαν συνεργαστεί με τους δύο

μεγάλους χρηματοπιστωτικούς οργανισμούς προτού αυτοί ξεκινήσουν τη μεταξύ τους συνεργασία και, ως εκ τούτου, το SET συγκεντρώνει τα καλύτερα στοιχεία που είχαν προκύψει από τις μέχρι τώρα έρευνες. Η λειτουργία του SET βασίζεται στην κρυπτογράφηση και τη χρήση ψηφιακών υπογραφών, με σκοπό τη διασφάλιση ότι ένα μήνυμα λαμβάνεται μόνο από τον επιθυμητό παραλήπτη, χωρίς αλλαγές στο περιεχόμενο, ενώ παράλληλα περιέχει στοιχεία που επιτρέπουν την επαλήθευση του αποστολέα του.

Το SET χρησιμοποιεί και συμμετρική αλλά και σύμμετρη μέθοδο κρυπτογράφησης, με αποτέλεσμα η διαδικασία της συναλλαγής να γίνεται μεν πιο πολύπλοκη, αλλά και περισσότερο ασφαλής.

Οι εταιρίες Microsoft και Netscape έχουν ήδη ανακοινώσει browsers που θα ενσωματώνουν το πρότυπο SET. Προκειμένου ο χρήστης να πραγματοποιήσει μια ηλεκτρονική συναλλαγή μέσω του WWW, θα χρειάζεται έναν εξ αυτών των browsers και έναν λογαριασμό σε οργανισμό που θα υποστηρίζει το SET. Μέσω του browser ο χρήστης επισκέπτεται το web site που τον ενδιαφέρει, επιλέγει προϊόντα και, με το πάτημα ενός κουμπιού, εκτελείται η παραγγελία του και επαληθεύονται τα στοιχεία που αφορούν στον ίδιο και το λογαριασμό του.

Στο παρασκήνιο, η διαδικασία είναι αρκετά πιο πολύπλοκη και αποτελείται από τα εξής βήματα: Ο πελάτης "ανοίγει" έναν λογαριασμό, ο οποίος μπορεί να είναι τραπεζικός, με πιστωτική κάρτα ή με κάποιο πιο σύγχρονο σύστημα πληρωμών, όπως για παράδειγμα το DigiCash. Το τελευταίο επιτρέπει στο χρήστη να μετατρέψει πραγματικό χρήμα σε άυλο. Ο λογαριασμός θα είναι με κάποιο χρηματοπιστωτικό οργανισμό, όπως η Mastercard και η Visa, ο οποίος θα υποστηρίζει το SET και στο εξής θα αναφέρεται ως Τράπεζα.

Ο πελάτης λαμβάνει ένα πιστοποιητικό. Με το άνοιγμα του λογαριασμού στην Τράπεζα, ο πελάτης λαμβάνει ένα ηλεκτρονικό αρχείο, το οποίο θα αναφέρεται ως Πιστοποιητικό και λειτουργεί ως πιστωτική κάρτα για on-line αγορές. Το αρχείο αυτό περιέχει πληροφορίες για τον πελάτη, συμπεριλαμβανομένου του δημόσιου κλειδιού. Το πιστοποιητικό έχει μια ημερομηνία λήξης και μια ηλεκτρονική υπογραφή της τράπεζας η οποία εξασφαλίζει την πιστότητά του.

Οι έμποροι έχουν τα δικά τους πιστοποιητικά. Κάθε έμπορος που συναλλάσσεται με την Τράπεζα διαθέτει Πιστοποιητικό, στο οποίο περιλαμβάνεται το δικό του δημόσιο κλειδί και το δημόσιο κλειδί της Τράπεζας. Το κλειδί αυτό διαθέτει επίσης ημερομηνία λήξεως και είναι υπογεγραμμένο ηλεκτρονικά προκειμένου να εξασφαλίζεται η πιστότητά του.

Ο πελάτης κάνει μια παραγγελία. Με τη χρήση ηλεκτρονικού ταχυδρομείου ή μέσω μιας web σελίδας, ο έμπορος ενημερώνεται σχετικά με τα προϊόντα ή τις υπηρεσίες που θέλει να αγοράσει ο πελάτης. Από τη στιγμή που θα δοθεί η εντολή αγοράς, συμβαίνουν τα εξής:

α) Ο browser λαμβάνει ένα αντίγραφο του Πιστοποιητικού του εμπόρου, γεγονός που εξασφαλίζει ότι το κατάστημα είναι διαπιστευμένο από την Τράπεζα. Ο πελάτης βλέποντας το πιστοποιητικό, εξασφαλίζεται ότι ο έμπορος είναι αυτός που υποστηρίζει ότι είναι, καθώς και ότι έχει το δικαίωμα να εκτελεί συναλλαγές.

β) Ο browser στέλνει στον έμπορο την παραγγελία, η οποία είναι κρυπτογραφημένη με το δημόσιο κλειδί του εμπόρου, ώστε να είναι αναγνώσιμη μόνο από αυτόν, την πληροφορία που αφορά στην πληρωμή, η οποία είναι κρυπτογραφημένη με το δημόσιο κλειδί της Τράπεζας και επομένως ο έμπορος δεν μπορεί να δει την πληροφορία αυτή, και, τέλος, έναν κωδικό που συνθέτει στοιχεία της παραγγελίας και της πληρωμής, προκειμένου να είναι σίγουρο ότι υπάρχει αντιστοιχία μεταξύ των δύο.

Ο έμπορος λαμβάνει την παραγγελία και ελέγχει τα προϊόντα που έχει παραγγείλει ο πελάτης και την ηλεκτρονική υπογραφή του.

Ο έμπορος επαληθεύει την ταυτότητα του πελάτη, χρησιμοποιώντας την Τράπεζα ή κάποιον τρίτο οργανισμό (VeriSign, Nortel) που έχει πληροφορίες για την αξιοπιστία του. Η διαδικασία αυτή είναι αντίστοιχη με την επαλήθευση της πιστωτικής κάρτας σας σε ένα κατάστημα. Αφού γίνει η επαλήθευση ο έμπορος στέλνει στον πελάτη ένα μήνυμα, προκειμένου να τον ενημερώσει ότι η παραγγελία έχει ληφθεί.

Ο έμπορος στέλνει την πληρωμή στην Τράπεζα, χρησιμοποιώντας το δημόσιο κλειδί που έχει στην κατοχή του. Το μήνυμα προς την Τράπεζα εκτός από τις πληροφορίες πληρωμής περιλαμβάνει και το Πιστοποιητικό του εμπόρου.

Η Τράπεζα ελέγχει τον έμπορο και το μήνυμα. Το λογισμικό της Τράπεζας ελέγχει πρώτα το αν ο έμπορος είναι εξουσιοδοτημένος και στη συνέχεια, μέσω της ηλεκτρονικής υπογραφής του μηνύματος, ελέγχει την αξιοπιστία του.

Η Τράπεζα ελέγχει την πληρωμή και παράλληλα ελέγχει ότι το συγκεκριμένο μήνυμα αφορά στον συγκεκριμένο έμπορο και τη συγκεκριμένη παραγγελία.

Η πληρωμή εγκρίνεται από την Τράπεζα και ένα κρυπτογραφημένο μήνυμα αποστέλλεται στον έμπορο, ο οποίος μπορεί πλέον να στείλει τα προϊόντα.

Έξυπνες κάρτες (Smart Cards) Οι "έξυπνες κάρτες" αποτελούν εξέλιξη των καρτών μαγνητικής λωρίδας (παθητικό μέσο αποθήκευσης, τα περιεχόμενα του οποίου μπορούν να διαβαστούν και να αλλαχθούν). Οι έξυπνες κάρτες μπορούν να αποθηκεύσουν μεγάλη ποσότητα δεδομένων και παρέχουν δυνατότητες κρυπτογράφησης και χειρισμού ηλεκτρονικών υπογραφών για την ασφάλεια των περιεχομένων τους. Η ιδέα της έξυπνης κάρτας ξεκίνησε στη Γαλλία το 1974. Το 1975 τα δικαιώματα ανάπτυξης πέρασαν σε μεγάλες εταιρίες ηλεκτρονικού εξοπλισμού. Η νέα αυτή τεχνολογία παρουσιάστηκε στο κοινό το 1981. Μια σειρά από πιλοτικά σχέδια ξεκίνησε αμέσως, και το 1984 με μια συλλογική αξιολόγησή τους εκδόθηκαν νέες ολοκληρωμένες προδιαγραφές. Σήμερα επικρατεί η λανθασμένη εντύπωση ότι οι Smart Cards είναι τραπεζικές ή πιστωτικές κάρτες, με αποτέλεσμα να μην αναγνωρίζεται το μεγάλο εύρος των δυνατοτήτων τους. Η τεχνολογία των έξυπνων καρτών προσφέρει απεριόριστες δυνατότητες χρήσης στη βιομηχανία, το εμπόριο και τη δημόσια διοίκηση.

2.3 Τεχνολογίες βάσεων δεδομένων

Καθώς οι επιχειρήσεις συνειδητοποιούν ότι τα δεδομένα είναι το σημαντικότερο περιουσιακό τους στοιχείο, τα συστήματα δικτυακής αποθήκευσης έρχονται για να βελτιώσουν την απόδοση, να αυξήσουν την παραγωγικότητα και να μειώσουν το κόστος διαχείρισης του εταιρικού αποθηκευτικού περιβάλλοντος. Βασικό στοιχείο για την αποθήκευση των δεδομένων μια εταιρίας είναι η ύπαρξη ενός συστήματος διαχείρισης βάσεων δεδομένων (database management system(DBMS).

Οι βάσεις δεδομένων είναι εφαρμογές λογισμικού (προγράμματα) που προσφέρουν ένα μεθοδικό και συστηματικό τρόπο συλλογής, καταχώρησης και συσχετισμού δεδομένων, ενώ παράλληλα επιτρέπουν την πρόσβαση σε αυτά και την ανάλυσή τους, με διάφορους τρόπους. Ουσιαστικά, πρόκειται για συλλογές δεδομένων, όπου δεδομένα θεωρούνται γεγονότα, στοιχεία και πληροφορίες που μπορούν να καταγραφούν ρητά και με σαφήνεια. Υπάρχουν διάφορα είδη βάσεων δεδομένων (σχεσιακές, ιεραρχικές, αντικειμενοστραφείς, δικτυωτές), όπως επίσης υπάρχουν και διάφορες γλώσσες προγραμματισμού για την κατασκευή τους. Πέραν αυτών των διακρίσεων, οι βάσεις δεδομένων χωρίζονται άτυπα σε "απλές" και "σύνθετες" ή "επαγγελματικές".

Οι απλές ταυτίζονται με μικρού μεγέθους εφαρμογές, που μπορούν να αξιοποιηθούν από έναν απλό χρήστη, μια μικρή επιχείρηση, γραφείο κ.λπ. και δεν απαιτούν τη χρήση πρόσθετου εξοπλισμού. Το κόστος των εν λόγω εφαρμογών (απλών βάσεων δεδομένων) είναι είτε μηδενικό (όταν περιλαμβάνονται μαζί με άλλα προγράμματα σε κάποιο πακέτο εφαρμογών γραφείου) είτε μικρό (όταν πρόκειται για μεμονωμένες εφαρμογές που διατίθενται χωριστά). Παραδείγματα απλών βάσεων δεδομένων είναι οι Lotus Approach, Corel Paradox, Filemaker Pro και Microsoft Access, με την τελευταία να αποτελεί το πιο διαδεδομένο πρόγραμμα αυτής της κατηγορίας.

Αναφορικά με τις σύνθετες (επαγγελματικές) βάσεις δεδομένων, αυτές δεν εντάσσονται στο στενό πλαίσιο ενός απλού προγράμματος ή μιας εφαρμογής που "τρέχει" σε κάποιον υπολογιστή, καθώς πρόκειται για ολοκληρωμένα συστήματα, που εκτός από εξειδικευμένο software απαιτούν και δαπανηρό εξοπλισμό (λ.χ. database servers, συστοιχίες οπτικών ή μαγνητικών δίσκων κ.ά.). Εδώ πλέον κάνουμε λόγο για ολοκληρωμένα συστήματα βάσεων δεδομένων και όχι για μια απλή εφαρμογή ή ένα πρόγραμμα του είδους. Τα ολοκληρωμένα αυτά συστήματα έχουν υψηλό κόστος, η υλοποίησή τους απαιτεί χρόνο, τακτική συντήρηση καθώς και την απασχόληση ειδικευμένου προσωπικού.

Οι επαγγελματικές βάσεις δεδομένων απευθύνονται σε μεγάλες επιχειρήσεις, με εκατοντάδες ή χιλιάδες εργαζομένους. Οι ευρύτερα χρησιμοποιούμενες από αυτές είναι:

MS SQL Server: είναι ένα σύστημα διαχείρισης βάσεων δεδομένων (database management system,DBMS), που έχει αναπτυχθεί και προωθείται από την Microsoft.Το σύστημα αυτό είναι το σημαντικότερο τμήμα του Microsoft Back Office, ενός εμπορικού πακέτου εφαρμογών πελάτη/διακομιστή (client/server).Το MS SQL Server εκτελείται αποκλειστικά στα λειτουργικά συστήματα της Microsoft.Αυτή η ιδιαιτερότητα έχει και ένα πολύ σημαντικό μειονέκτημα: δεν μπορεί να εκμεταλλευτεί τις προχωρημένες ιδιότητες ενός λειτουργικού συστήματος όπως το UNIX, το οποίο, σε μερικές περιοχές, όπως οι εμπλουτισμένες παράλληλες αρχιτεκτονικές ή εταιρικές εφαρμογές, συνεχίζει να έχει ορισμένα πλεονεκτήματα σε σχέση με τα Windows.Τα σημαντικότερα στοιχεία του SQL Server είναι:

α)Είναι εύκολο στην χρήση

β)Μπορεί να χρησιμοποιηθεί από έναν φορητό υπολογιστή μέχρι σε συστήματα συμμετρικών πολυεπεξεργαστών (SMP).

γ)Παρέχει χαρακτηριστικά αποθηκών δεδομένων, που μέχρι τώρα διατίθονται μόνο στην Oracle και σε πιο ακριβά συστήματα DBMS.

Το SQL Server ήταν από την αρχή σχεδιασμένο σαν ένα DBMS πελάτη/διακομιστή. Η αρχιτεκτονική πελάτη/διακομιστή αναπτύχθηκε για να γίνεται διαχείριση ενός μεγάλου αριθμού υπολογιστών (PC,σταθμών εργασίας,SMP), που είναι συνδεδεμένοι σε ένα δίκτυο. Η λειτουργικότητα του SQL Server διαιρείται ανάμεσα σε πελάτες και διακομιστές. Ένας πελάτης παρέχει μια ή περισσότερες διασυνδέσεις χρήστη, που χρησιμοποιούνται για δημιουργία μιας αίτησης χρήστη προς ένα DBMS. Ο διακομιστής (δηλ.το DBMS) επεξεργάζεται αυτή την αίτηση και στέλνει το αποτέλεσμα πίσω στον πελάτη.

Oracle Database: Είναι όπως η MS SQL SERVER μια σχεσιακή βάση δεδομένων. Η Oracle μια από τις μεγαλύτερες εταιρίες έχει δέσει στην Oracle Database 10g η οποία είναι η τελευταία έκδοση του λογισμικού της για βάσεις δεδομένων κάποια χαρακτηριστικά που ξεχωρίζουν την Oracle Database 10g από τα ανταγωνιστικά προϊόντα. Η Oracle Database 10g παρέχει το clustering και το workload balancing, την υψηλή διαθεσιμότητα και την αυτοματοποιημένη διαχείριση που απαιτούνται για την εφαρμογή των υποδομών επιχειρηματικού Grid Computing. Η Oracle Database 10g, διαθέσιμη από το 2004, παρέχει ενισχυμένα χαρακτηριστικά που διευκολύνουν την υλοποίηση ενώ απλοποιούν τη διαχείριση της βάσης δεδομένων. Χαρακτηριστικά όπως το Automatic Storage Management, το οποίο απαλείφει, εικονικά, την ανάγκη για χειροκίνητη παραμετροποίηση και ρύθμιση του database storage, και οι εξελιγμένες XML δυνατότητες .

Υπάρχουν βεβαια και λύσεις απο τον χώρο των open source εφαρμογών όπως είναι η βάση δεδομένων MySQL και η βάση δεδομένων PostgreSQL.

3 ΑΝΑΛΥΣΗ ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΑΝΑΠΤΥΞΗ ΠΛΑΤΦΟΡΜΑΣ ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ

Σε αυτό το κεφάλαιο θα αναπτύξουμε μια Web-EDI εφαρμογή που αποσκοπεί στην ανάπτυξη της αποτελεσματικότητας μιας επιχείρησης καθώς και στην ηλεκτρονική ανταλλαγή δεδομένων ανάμεσα στους χρήστες της εφαρμογής.

Το case study είναι μια εφαρμογή για το δημοπρατήριο ενός Συνεταιρισμού Αγροτικών Προϊόντων.

Η ανάπτυξη της πλατφόρμας θα γίνει με τεχνολογία ASP.NET και η βάση δεδομένων θα υλοποιηθεί σε MS SQL Server.

3.1 Ανάλυση - Σχεδιασμός έργου

Αρχικά ήρθαμε σε επαφή με τους υπεύθυνους του συνεταιρισμού με σκοπό να καταγράψουμε τις απαιτήσεις τους και να συγκεκριμενοποιήσουμε ποιες λειτουργίες θα πρέπει να κάνει το σύστημα. Η απαίτηση της εταιρίας ήταν να δημιουργήσουμε για λογαριασμό της μια εμπορική εφαρμογή για την καταγραφή και την επεξεργασία των δεδομένων που προκύπτουν απο τις εργασίες της εταιρίας. Η εφαρμογη θα πρεπει να εχει την δυνατοτητα να δεχεται ταυτοχρονη επεξεργασία απο πολλους χρήστες. Μετά απο αυτές τις συναντήσεις καταληξαμε στις τελικές απαιτησεις και μπορουμε πλεον να σχεδιασουμε το συστημα.

Το συστημα θα εχει τι παρακατω οντοτητες:

Παραγωγός: Οντοτητα που περιγραφει εναν παραγωγό

Εμπορος: Οντοτητα που περιγραφει εναν εμπορο

Προϊόν: Οντοτητα σχετική με τα προϊόντα

Προσφορά: Ένα σύνολο απο προϊόντα τα οποία φέρνει ένας παραγωγός προς δημοπράτηση

Παρτίδα: Ένα σύνολο προσφορών

Ζήτηση: Μια ζήτηση που γίνεται απο εναν Εμπορο και αντιστοιχεί σε μια συγκεκριμένη προσφορά

Τιμολόγιο: Η Οντοτητα Τιμολόγιο που καλύπτει Τιμολόγιο Πώλησης ,Δελτίο Αποστολης, Δελτιο Αποστολής-Τιμολόγιο Πωλησης, Τιμολογιο Αγορας καθως και Πιστωτικα Τιμολόγια.

Πληρωμή: Οντοτητα που ασχολείται με τις οικονομικές συναλλαγές μεταξύ συνεταιρισμού-εμπόρων και συνεταιρισμού-παραγωγών.

Λογαριασμός Κλουβών: Καταγράφει τις συναλλαγές στις κλούβες του συνεταιρισμού

Χρήστης: Περιγραφει τους χρήστες του συστήματος

Οι διεργασίες που θα εκτελούνται είναι οι παρακάτω:

Εισαγωγή-Ενημέρωση-Διαγραφή Παραγωγού.

Εισαγωγή-Ενημέρωση-Διαγραφή Εμπόρου.

Εισαγωγή-Ενημέρωση-Διαγραφή Παρτίδας Προϊόντων.

Εισαγωγή-Ενημέρωση-Διαγραφή Ζήτησης.

Εισαγωγή-Ενημέρωση-Διαγραφή Ζήτησης Εκτός Δημοπρασίας.

Εκτέλεση Δημοπρασίας.

Εισαγωγή-Ενημέρωση-Ακύρωση Δελτίου Αποστολης.

Εισαγωγή-Ενημέρωση-Ακύρωση Τιμολογίου Πωλησης.

Εισαγωγή-Ενημέρωση-Ακύρωση Δελτίου Αποστολης - Τιμολογίου Πωλησης .

Εισαγωγη-Ενημερωση-Ακύρωση Τιμολογίου Αγοράς

Εισαγωγή-Ενημέρωση-Διαγραφή Πληρωμής.

Εισαγωγή-Ενημέρωση-Διαγραφή Προϊόντος.

Εισαγωγή-Ενημέρωση-Διαγραφή στον λογαριασμό Κλουβών.

Εισαγωγή-Ενημέρωση-Διαγραφή Χρήστη.

Δημιουργία Reports

Οι χρήστες του συστήματος θα χωρίζονται σε δύο κατηγορίες, τους απλούς χρήστες και τους διαχειριστές του συστήματος. Οι διαχειριστές του συστήματος θα έχουν πρόσβαση σε

όλες τις διεργασίες του συστήματος. Απο την άλλη πλευρά οι απλοί χρήστες δεν έχουν πρόσβαση στις παρακάτω διεργασίες:

Δεν μπορεί να κάνει Εισαγωγή-Ενημέρωση-Διαγραφή Χρήστη.

Δεν μπορεί να κάνει Εισαγωγή-Ενημέρωση-Διαγραφή Προϊόντος.

Δεν μπορεί να κάνει Εκτέλεση Δημοπρασίας.

Δεν μπορεί να κάνει Ενημέρωση-Διαγραφή Προσφοράς που έχει δημοπρατηθεί.

Δεν μπορεί να κάνει Ενημέρωση-Διαγραφή Ζήτησης που έχει δημοπρατηθεί.

Δεν μπορεί να κάνει Διαγραφή Παραγωγού.

Δεν μπορεί να κάνει Διαγραφή Εμπόρου.

Για το workflow του συστήματος θα ορίσουμε τους παρακάτω κανόνες:

Για το καθορισμό του τύπου ενός τιμολογίου θα πρεπει να χρησιμοποιήσουμε στη βάση στον πίνακα των τιμολογίων το οποίο ανάλογα με το είδος του τιμολογίου θα παίρνει τις παρακατω τιμές:

Οι τύποι των Τιμολογίων είναι:

Τιμολόγιο Αγοράς με type = 1. Είναι το τιμολόγιο που εκδίδεται για την αγορά προϊόντων απο ένα παραγωγό.

Δελτίο Αποστολής με type = 2. Είναι το τιμολογιο που εκδίδεται για την μεταφορά προϊόντων προς ένα εμπορο. Μετά απο το δελτίο αποστολής πρεπει να κοπεί τιμολόγιο αγοράς για το συγκεκριμένο δελτιο αποστολης.

Τιμολόγιο Πώλησης με type = 3. Είναι το τιμολόγιο που εκδίδεται για την πώληση προϊόντων προς ένα εμπορο. Αναφέρεται σε ένα ή περισσότερα δελτία αποστολής.

Δελτίο Αποστολής - Τιμολόγιο Πώλησης με type = 4. Είναι ουσιαστικά ένα ενοποιημένο τιμολόγιο που συνδυάζει τις δυο προηγούμενες περιπτώσεις.

Dummy Τιμολόγιο type = 5. Περιλαμβάνει πιστωτικά τιμολόγια που δημιουργούνται όταν επιστρέφονται κλούβες, πιστώνει διάφορες που προκύπτουν κατά τις πληρωμές και καλύπτει και την περίπτωση των ακυρωτικών τιμολογίων.

Για τον έλεγχο της κατάστασης στην οποία βρίσκεται μια προσφορά θα πρεπει να χρησιμοποιήσουμε στη βάση στον πίνακα των προσφορών ένα πεδίο status το οποίο να δείχνει σε πια κατάσταση βρίσκεται η κάθε προσφορά.

Το πεδίο αυτό θα παίρνει ανάλογα τις τιμές:

Προσφορά Status 1: Η προσφορά είναι καταχωρημένη και έτοιμη για δημοπράτηση.

Προσφορά Status 2: Η προσφορά εχει δημοπρατηθεί.

Προσφορά Status 3: Έχει κοπεί τιμολόγιο αγοράς για την συγκεκριμένη προσφορά.

Ακόμα θα δημιουργήσουμε ένα πεδίο IpolipoKloubes το οποίο θα δείχνει αν απο την προσφορά υπάρχουν ακόμα μη δημοπρατημένες κλούβες. Η τιμή του θα είναι ίση με τις αρχικές κλούβες μείον τις δημοπρατημένες

Για τον έλεγχο της κατάστασης στην οποία βρίσκεται μια ζήτηση θα πρεπει να χρησιμοποιήσουμε στη βάση στον πίνακα των ζητήσεων ένα πεδίο status το οποίο να δείχνει σε πια κατάσταση βρίσκεται η κάθε ζήτηση.

Το πεδίο αυτό θα παίρνει ανάλογα τις τιμές:

Ζήτηση Status 1: Η ζήτηση είναι καταχωρημένη και έτοιμη για δημοπράτηση.

Ζήτηση Status 2: Η ζήτηση εχει δημοπρατηθεί και είναι έτοιμη για τιμολόγηση.

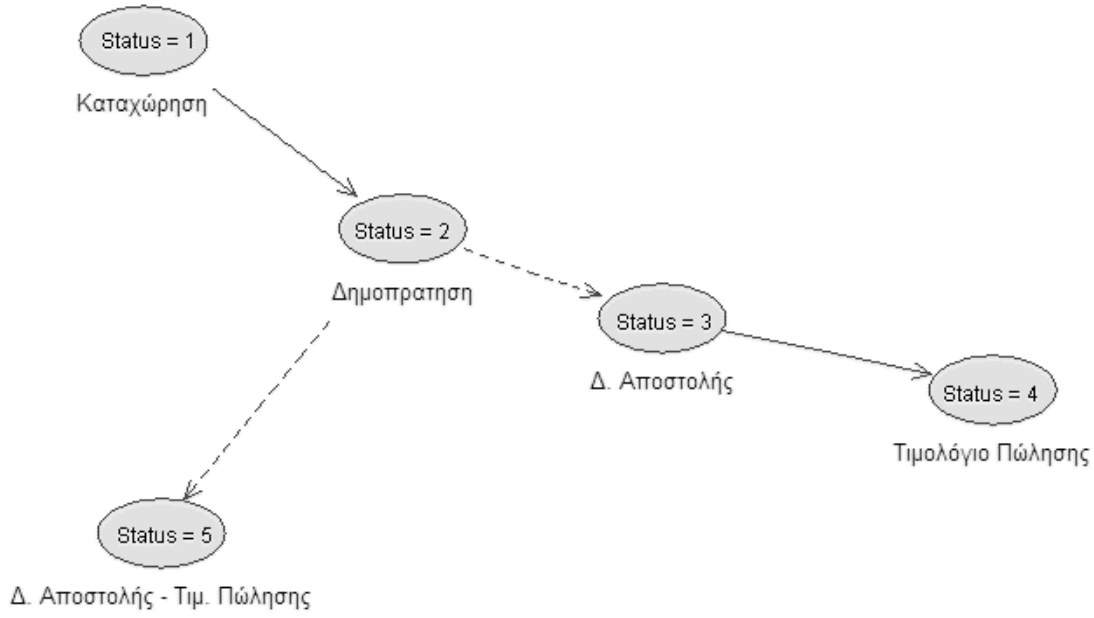
Ζήτηση Status 3: Έχει κοπεί Δελτίο Αποστολής για την συγκεκριμένη ζήτηση.

Ζήτηση Status 4: Η ζήτηση Τιμολόγιο Πώλησης για την συγκεκριμένη ζήτηση.

Ζήτηση Status 5: Έχει κοπεί Δελτίο Αποστολής - Τιμολόγιο Πώλησης για την συγκεκριμένη ζήτηση.

Για τους παραπάνω κανόνες φτιάξαμε διαγράμματα ροής, περιπτώσεων χρήσης και αλληλεπίδρασης, τα οποία μας βοήθησαν να ελέγχουμε όλες τις παραμέτρους που θα έχει το συστημα.

Στο παρακάτω διάγραμμα φαίνεται το workflow της ζήτησης



Εικόνα 1. Το workflow της ζήτησης

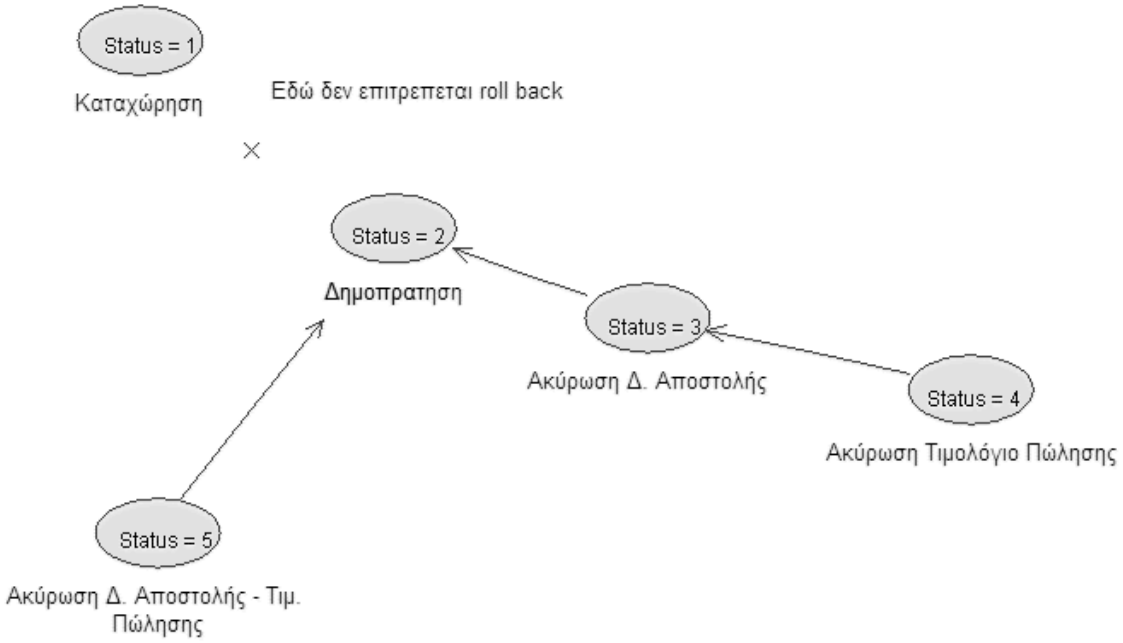
Όπως φαίνεται απο το διάγραμμα τα πιθανά σενάρια για την ζήτηση είναι τα εξής δύο

α)Καταχώρηση Ζήτησης → Δημοπράτηση Ζήτησης → Δελτίο Αποστολής - Τιμολόγιο Πώλησης για την Ζήτηση

β)Καταχώρηση Ζήτησης → Δημοπράτηση Ζήτησης → Δελτίο Αποστολής για την Ζήτηση → Τιμολόγιο Πώλησης για την Ζήτηση

Για όλους τους παρακατω κανόνες θα πρεπει να μπορεί να γίνει roll back δηλαδή η αντίστροφη διαδικασία σε περίπτωση ακύρωσης ή διαγραφής.

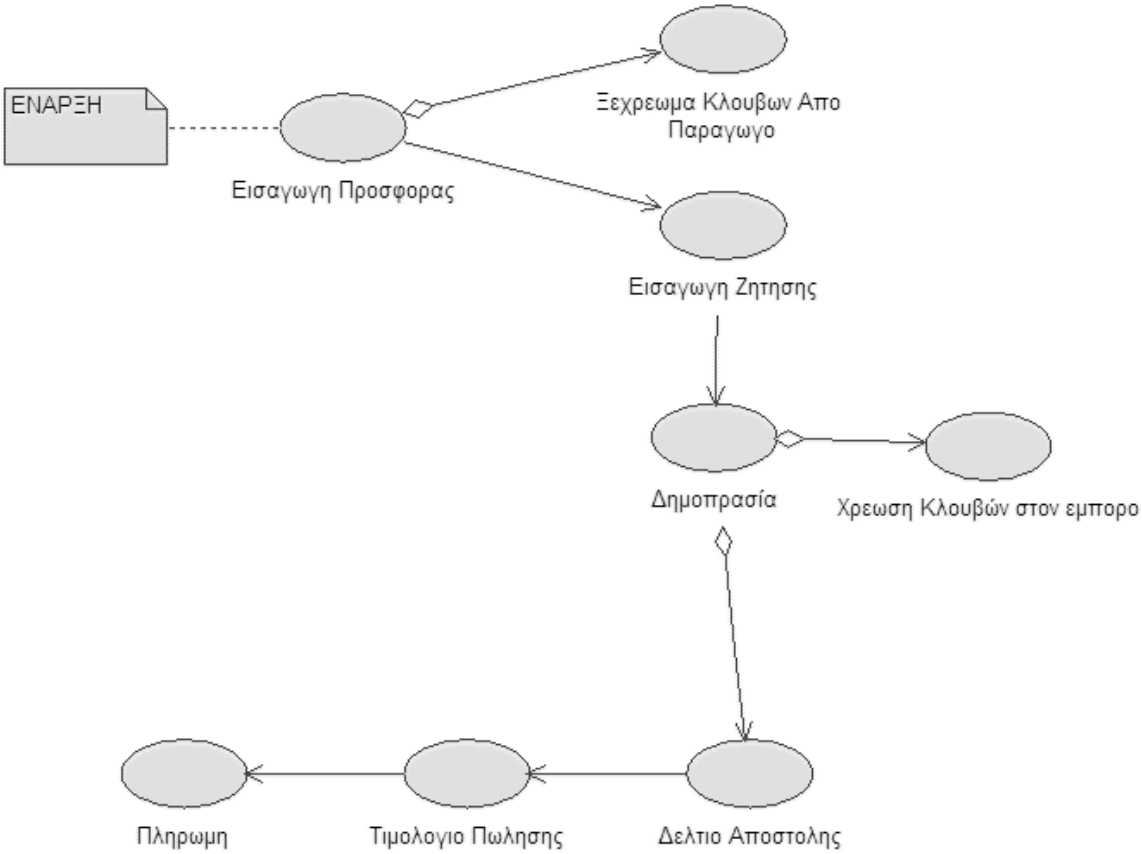
Ενδεικτικό είναι το παρακάτω διάγραμμα για τη ζήτηση



Εικόνα 2. Το workflow σε περίπτωση roll back της ζήτησης

Όπως φαίνεται στο διάγραμμα τα σενάρια είναι αντίστροφα με αυτά της εικόνας 1 και το status της ζήτησης αλλάζει κατά την ακύρωση των τιμολογίων ως post-condition. Επειδή μια δημοπρασία που εχει γίνει δεν μπορεί να αλλάξει το status της ζήτησης δεν μπορεί να ξαναγίνει 1.

ένα διάγραμμα ροής για ένα πιθανό σενάριο χρήσης είναι αυτό που φαίνεται στο επόμενο διάγραμμα.



Εικόνα 3. Διάγραμμα ροής ενός πιθανού σεναρίου

3.2 Σχεδιασμός βάσης δεδομένων

Μετά το στάδιο της ανάλυσης επόμενο βήμα είναι ο σχεδιασμός της βάσης δεδομένων την βάση δεδομένων. Η σχεδίαση μιας βάσης δεδομένων είναι πολύ σημαντική φάση του κύκλου ζωής της βάσης δεδομένων, που προηγείται όλων των άλλων φάσεων, εκτός της συλλογής απαιτήσεων και της ανάλυσης. Αν η σχεδίαση της βάσης δεδομένων δημιουργείται κυρίως διαισθητικά και χωρίς κάποιο σχέδιο, τότε η βάση δεδομένων μάλλον δεν θα ικανοποιεί τις απαιτήσεις του χρήστη σε ότι αφορά την απόδοση. Μια άλλη συνέπεια μιας κακής σχεδίασης βάσης δεδομένων είναι ο υπερβολικός πλεονασμός δεδομένων, που με την σειρά του έχει δύο μειονεκτήματα: την ύπαρξη ανωμαλιών δεδομένων και την χρήση μη αναγκαίου χώρου του δίσκου.

Η **κανονικοποίηση** δεδομένων είναι μία διαδικασία κατά την διάρκεια της οποίας, οι υπάρχοντες πίνακες μιας βάσης δεδομένων δοκιμάζονται για να βρεθούν ορισμένες εξαρτήσεις ανάμεσα στις στήλες ενός πίνακα. Αν υπάρχουν τέτοιες εξαρτήσεις, τότε ο πίνακας αναδομείται σε πολλούς (συνήθως σε δύο) πίνακες, κάτι που εξαλείφει τις τυχόν εξαρτήσεις στηλών. Αν ένας από αυτούς τους παραγόμενους πίνακες συνεχίζει να περιέχει εξαρτήσεις δεδομένων, η διαδικασία της κανονικοποίησης επαναλαμβάνεται μέχρι να μην υπάρχουν άλλες οι εξαρτήσεις.

Η διαδικασία εξάλειψης πλεονασμού δεδομένων σε έναν πίνακα βασίζεται στην θεωρία των λειτουργικών εξαρτήσεων. Μια **λειτουργική εξάρτηση** σημαίνει ότι χρησιμοποιώντας την γνωστή τιμή μιας στήλης, η αντίστοιχη τιμή μιας άλλης στήλης μπορεί να καθορίζεται πάντα μοναδικά. Οι λειτουργικές εξαρτήσεις ανάμεσα στις στήλες Α και Β δηλώνονται σαν Α -> Β, καθορίζοντας ότι μιας τιμή στην στήλη Α μπορεί να χρησιμοποιείται πάντα για να καθορίσει την αντίστοιχη τιμή στην στήλη Β.(«η Β εξαρτάται λειτουργικά από την Α.»)

Κανονικές μορφές

Οι κανονικές μορφές χρησιμοποιούνται για την διαδικασία κανονικοποίησης δεδομένων και έτσι, για την σχεδίαση της βάσης δεδομένων. Θεωρητικά, υπάρχουν τουλάχιστον πέντε διαφορετικές κανονικές μορφές, από τις οποίες οι τρείς πρώτες είναι οι πιο σημαντικές για

πρακτική χρήση. Η τρίτη κανονική μορφή για έναν πίνακα μπορεί να επιτευχθεί ελέγχοντας την πρώτη και δεύτερη κανονική μορφή στις ενδιάμεσες καταστάσεις, και έτσι ο σκοπός της καλής σχεδίασης μιας βάσης δεδομένων μπορεί να επιτευχθεί συνήθως αν όλοι οι πίνακες μιας βάσης δεδομένων είναι στην τρίτη μορφή.

Σύμφωνα με τους κανόνες που αναφέραμε πιο πάνω και με οδηγό την ανάλυση του έργου σχεδιάσαμε την βάση, με τους παρακάτω πινακες και τα αντιστοιχα πεδία σε καθενα απο αυτους.

Πίνακας Παραγωγός: ParagogosID, Onoma, Eponimo, Til1, Til2, Taytotita, AFM, DOY, ApalagiELGA, ApofasiELGA, FPA, Ektasi, ArithmKikloforias.

Πίνακας Έμπορος: EmporosID, Onoma, Eponimo, Dieythinsi, Til1, Til2, Taytotita, AFM, DOY, ArithmKikloforias, Protereotita.

Πίνακας Προϊον: ProionID, Kodikos, Eidos, Sintomeysi.

Πίνακας Προσφορά: ProsforaID, PartidaID, Katigoria, Piotita, Fotografia, Sxolia, Kloubes, Mikta, Apobaro, Dialogi, Kathara, Status, IpolipoKloubes.

Πίνακας Παρτίδα: PartidaID, ParagogosID, Imerominia.

Πίνακας Ζήτηση: ZitisiID, EmporosID, ProsforaID, TimologioID, TimologioAgorasID, AritmDeltiouApostolis, Katigoria, ImerominiaZitisis, KloubesZitisis, TimiZitisis, ImerominiaDimoprasias, KloubesDimoprasias, TimiDimoprasias, KilaAnaKlouba, Kostos, Status.

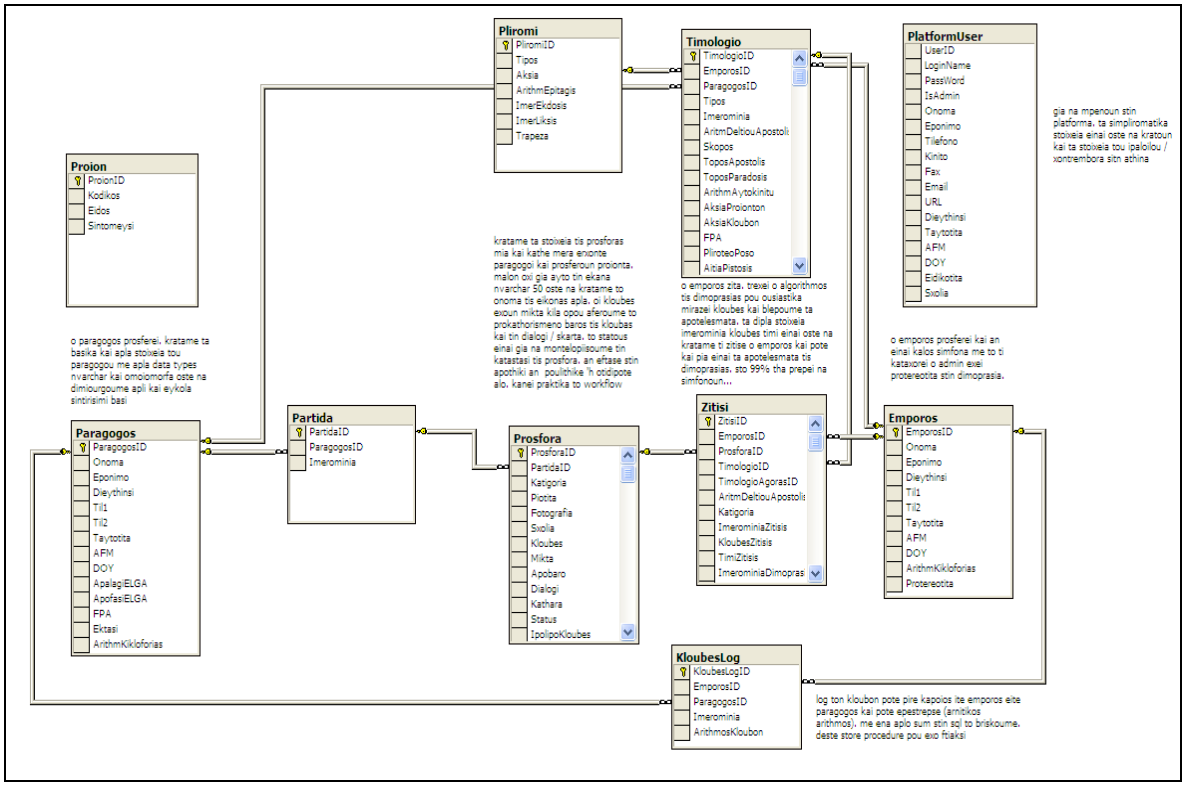
Πίνακας Τιμολόγιο: TimologioID, EmporosID, ParagogosID, Tipos, Imerominia, AritmDeltiouApostolis, Skopos, ToposApostolis, ToposParadosis, ArithmAytokinitu, AksiaProionton, AksiaKloubon, FPA, PliroteoPoso, AitiaPistosis, PliromiID.

Πίνακας Πληρωμή: PliromiID, Tipos, Aksia, ArithmEpitagis, ImerEkdosis, ImerLiksis, Trapeza.

Πίνακας Λογαριασμός Κλουβών: KloubesLogID, EmporosID, ParagogosID, Imerominia, ArithmosKloubon.

Πίνακας Χρήστης: UserID, LoginName, PassWord, IsAdmin, Onoma, Eponimo, Tilefono, Kinito, Fax, Email, URL, Dieythinsi, Taytotita, AFM, DOY, Eidikotita, Sxolia.

Η βάση είναι οσο το δυνατόν περισσοτερο κανονικοποιημένη και τα relationships φαίνονται στο επόμενη εικόνα



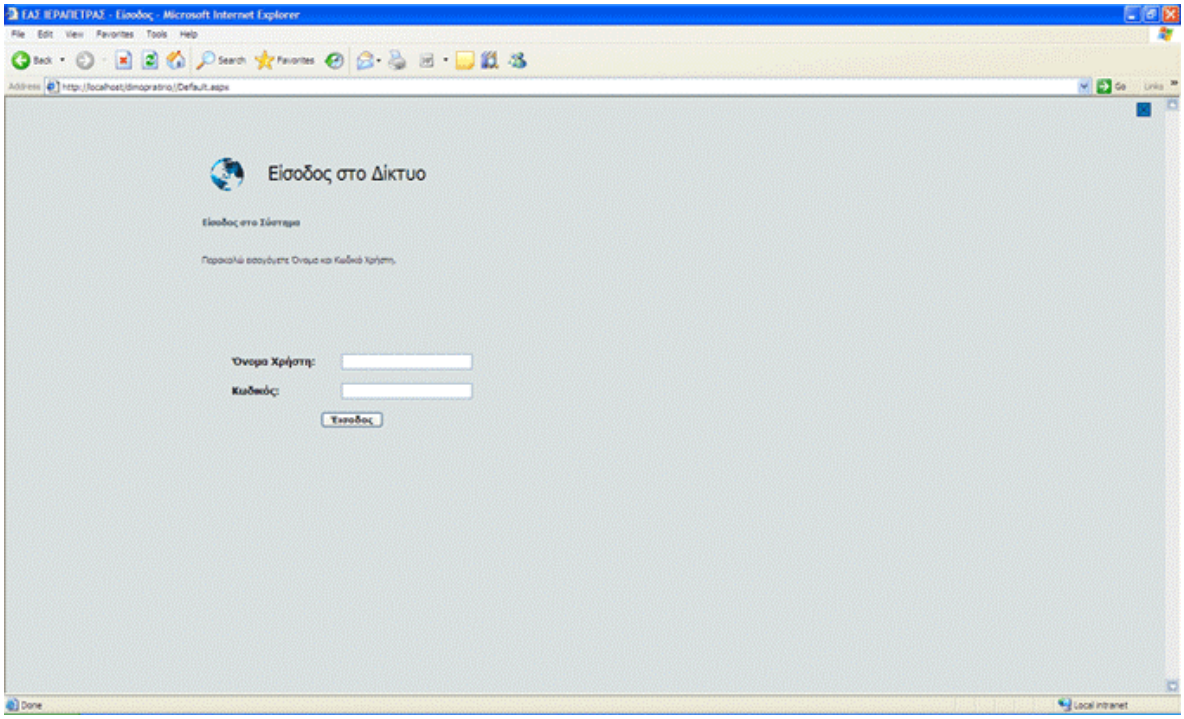
Εικόνα 4. Το διάγραμμα της βάσης δεδομένων

Αφου δημιουργήσαμε την βάση δεδομένων μπορούμε τώρα να αρχισουμε την ανάπτυξη της πλατφόρμας στο Microsoft Visual Studio .net.

3.3 Ανάπτυξη Πλατφόρμας

Παρότι δεν είναι δυνατόν να περιγραψουμε πλήρως την αναπτυξη της πλατφορμας σε ένα περιορισμενης εκτασης κειμενο, θα προσπαθήσουμε να παρουσιάσουμε τα βασικότερα σημεία της αναπτυξης καθως και να δωσουμε στον αναγνώστη την ευκαιρία να αντιληφθει τον τροπο σκεψης που χρησιμοποιηθηκε κατα την ανάπτυξη του συστήματος. Βασικο στοιχειο κατα την αναπτυξη είναι η προσπάθεια να επιτευχθεί η απλοτητα του κωδικα καθως και η αναπτυξη του με τετοιο τροπο ωστε να είναι δυνατη η ευκολη επαναχρησιμοποιηση του.

Πρώτα απ ολα θα φτιαξουμε μια σελιδα για τον ελεγχο της προσβασης στην εφαρμογή. Ο χρηστης θα πρεπει να εισάγει, username και password για να αποκτήσει πρόσβαση στο συστημα. Η οθόνη για την ταυτοποιηση του χρηστη φιανεται στην παρακατω εικόνα



Εικόνα 5. Η οθόνη για την είσοδο στο συστημα

Αν αυτός που επιθυμεί να εισέλθει στο συστημα εισάγει username και password που δεν αντιστοιχούν σε εγγραφή στον πίνακα χρήστες της βάσης δεδομένων τότε του επιστρέφεται μήνυμα ότι η εισαγωγή δεν είναι δυνατή και ότι θα πρεπει να επιλέξει διαφορετικό username και password.

Αν η εισαγωγή αντιστοιχεί σε εγγραφή στον πίνακα χρήστες της βάσης δεδομένων τότε ανοίγει η σελίδα για την επιλογή διεργασίας. Ακόμα ελέγχεται αν ο χρήστης είναι διαχειριστής ώστε να μπορέσουμε παρακάτω να του δώσουμε τα ανάλογα δικαιώματα, ή αν δεν είναι, αντίστοιχα να τον περιορίσουμε σε συγκεκριμένες διεργασίες τις οποίες θα μπορεί να εκτελέσει..

Παράλληλα δημιουργείται μια νέα συνοδός (session) για την συγκεκριμένη σύνδεση με το σύστημα. Μέσα απο τις παραμέτρους που ορίζονται σε αυτή τη session μπορουμε να αντλήσουμε διάφορες πληροφορίες στη συνέχεια. Οι πληροφορίες αυτές καταγράφονται στο αρχείο global.asax μέρος του οποίου φαίνεται πιο κάτω.

```
Sub Session_Start(ByVal sender As Object, ByVal e As EventArgs)
    '---Set session timeout to 14 minutes
    Session.Timeout = 140

    'Session.LCID=1033 'English - United States
    'Session.LCID=1032 'Greek

    Session("GLB_UserID") = -1
    Session("GLB_LoginName") = ""
    Session("GLB_UserName") = ""
    Session("GLB_IsAdmin") = 0

    Session("GLB_RUNTYPE") = "DEBUG" '"DEBUG" / "RELEASE"

    '---Paths
    Session("GLB_ServerIP") = "http://" &
Request.ServerVariables("SERVER_NAME") & "/"

    If Session("GLB_RUNTYPE") = "DEBUG" Then
       Session("GLB_SiteRootPATH") = Session("GLB_ServerIP") & "dimopratirio/"
    Else
       Session("GLB_SiteRootPATH") = Session("GLB_ServerIP") & "dimopratitio/"
    End If
    Session("GLB_RootPATH") = Session("GLB_SiteRootPATH") & "/"
 End Sub
```

Χρησιμοποιώντας κάποιες απο τις session μεταβλητές που ορίζονται πιο πάνω μπορουμε κατά την διάρκεια χρήσης του προγράμματος να κάνουμε διάφορους έλεγχους και να έχουμε πληροφορίες για την κατάσταση της συνόδου.

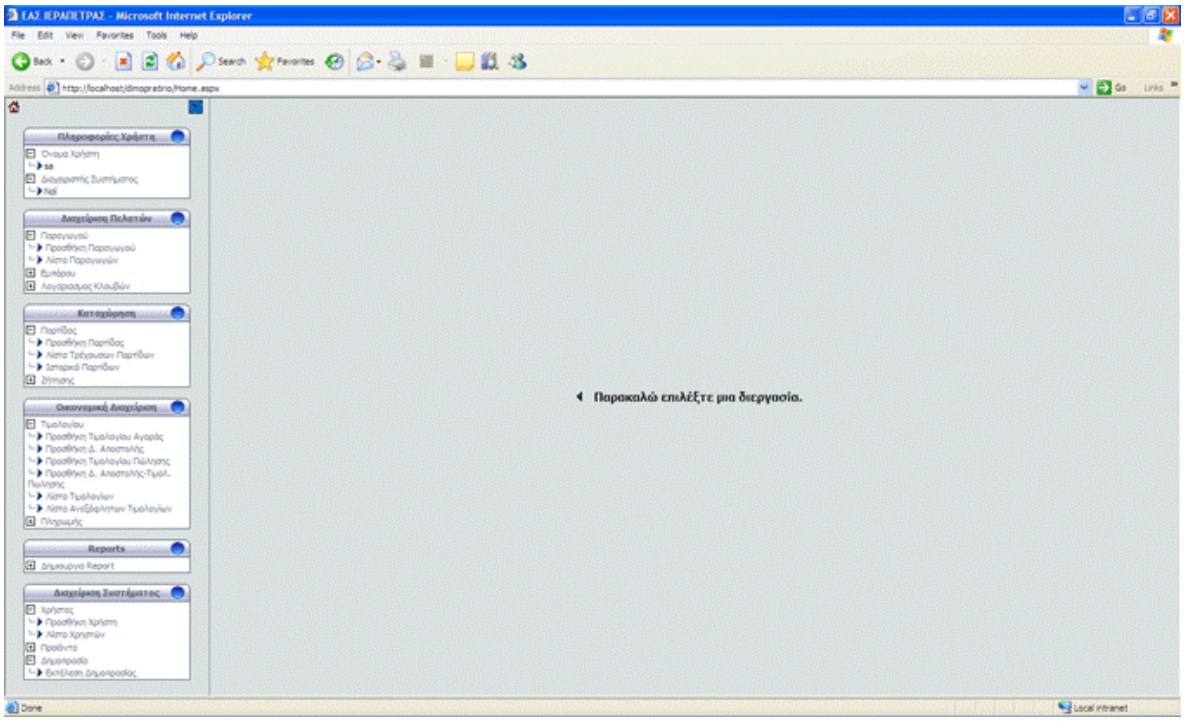
Κατά την έξοδο του χρήστη απο την εφαρμογη η σύνοδος αυτή τελειώνει και οι πληροφορίες της καταστρέφονται.

Για να αποτρέψουμε μη εξουσιοδοτημένη χρήση στην περίπτωση που κάποιος πληκτρολογήσει την διεύθυνση κάποιας από τις σελίδες του προγράμματος δημιουργήσαμε ένα include file, το οποίο ελέγχει αν εχει γίνει login και αν δεν εχει γίνει τότε απαγορεύει την πρόσβαση στη ζητούμενη σελίδα και οδηγεί στη σελίδα για την ταυτοποίηση του χρήστη. Ο κώδικας αυτού του αρχείου το οποίο κάνουμε include σε κάθε σελίδα της εφαρμογής είναι ο παρακάτω.

```
Response.Buffer=True

Response.CacheControl = "no-cache"
```

```
Response.AddHeader ("Pragma", "no-cache" )
Response.Expires = -1

If session("GLB_LoginName")="" then
  Response.Redirect (session("GLB_RootPATH") & "LoginNotEntered.aspx")
End if
```

Αφού ο χρήστης κάνει επιτυχημένο login τότε μεταφέρεται στην σελίδα επιλογής διεργασίας η οποία φαίνεται στην επόμενη εικόνα.



Εικόνα 6. Η οθόνη επιλογής διεργασίας

Όπως φαίνεται και στην εικόνα η οθόνη είναι χωρισμένη σε δύο frames. Στο αριστερό frame εμφανίζονται πληροφορίες σχετικές με το χρήστη και υπάρχει το menu απο το οποίο μπορεί ο χρήστης να επιλέξει ποια διεργασία θέλει να πραγματοποιήσει.

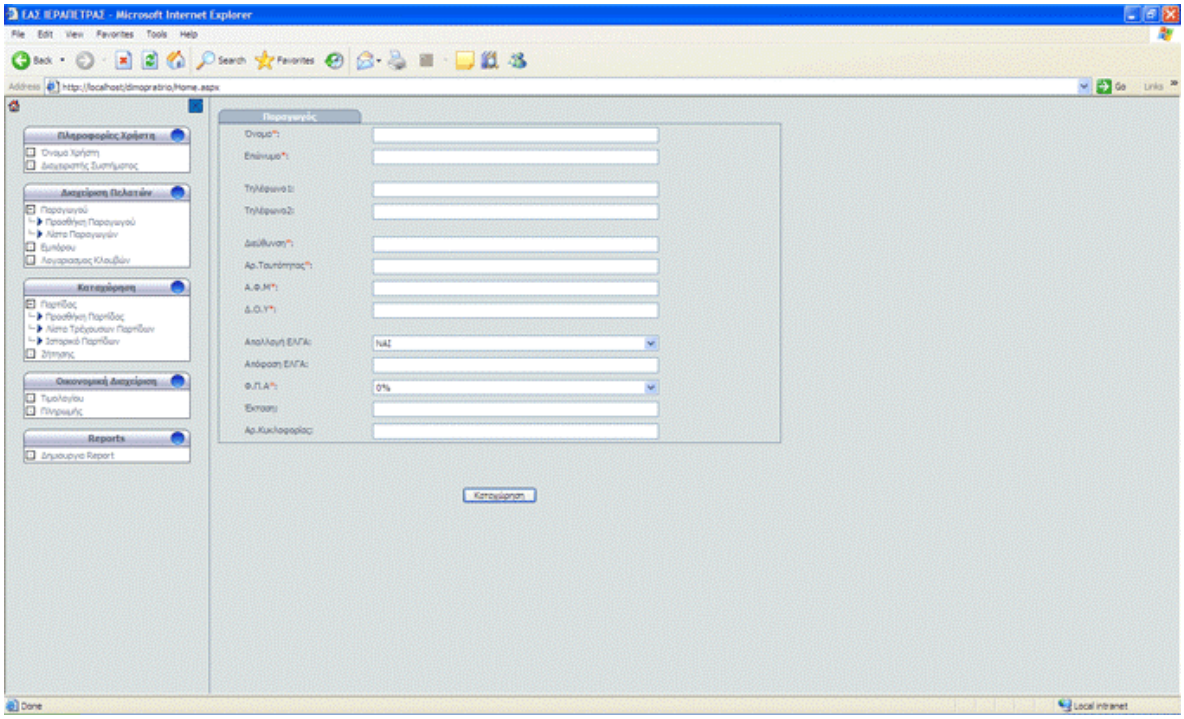
Το δεξιό frame είναι ο κύριος χώρος εργασίας και σε αυτό εμφανίζεται η σχετική σελίδα ανάλογα με την διεργασία την οποία εχει επιλέξει ο χρήστης.

Πιο κάτω θα αναλύσουμε τον μέρος του κώδικα που αφορά στο σύνολο των διεργασιών που αφορούν μια Οντοτητα και συγκεκριμένα την οντοτητα του παραγωγού.

.

Ας επιλέξουμε την διεργασία Καταχώρηση Παραγωγού. Επιλέγοντας την διεργασία στο δεξιό frame θα ανοίξει η σελίδα Paragogos.aspx?WF_Activity=INSERT. Δίνοντας το

query string WF_Activity κάνουμε την εφαρμογη δυναμική, αφού χρησιμοποιούμε την ίδια σελίδα σε πολλές διεργασίες τροποποιώντας την ανάλογα με την διεργασία. Ας δούμε τώρα πως λειτουργεί η παραπάνω διεργασία.



Εικόνα 7. Η οθόνη για την καταχώρηση Παραγωγού

Εδώ ο χρήστης πρεπει να εισαγάγει τις τιμές στα πεδία και πατώντας το κουμπί Καταχώρηση θα δημιουργηθεί στη βάση μία εγγραφή στον πίνακα των παραγωγών.

Ας ρίξουμε όμως μια μάτια σε κάποια σημεία του κώδικα υποστήριξης για την σελίδα Paragogos.aspx.

Ας δούμε τι γίνεται όταν φορτώνεται η σελίδα.

```
 Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    If Not IsPostBack Then

        WF_Activity = Request.QueryString("WF_Activity")
        If (WF_Activity = "UPDATE" Or WF_Activity = "DELETE") Then
            Dim ID As Integer
            ID = Request.QueryString("ID")
            da = New SqlDataAdapter("select * from Paragogos where ParagogosID=" & ID,
MyConnection)
            MyConnection.Open()
            da.Fill(ds, "Paragogos")
            txtOnoma.Text = ds.Tables("Paragogos").Rows(0).Item("Onoma").ToString
            txtEponimo.Text = ds.Tables("Paragogos").Rows(0).Item("Eponimo").ToString
```

```
            txtDieythinsi.Text =
ds.Tables("Paragogos").Rows(0).Item("Dieythinsi").ToString
            txtTil1.Text = ds.Tables("Paragogos").Rows(0).Item("Til1").ToString
            txtTil2.Text = ds.Tables("Paragogos").Rows(0).Item("Til2").ToString
            txtTaytotita.Text = ds.Tables("Paragogos").Rows(0).Item("Taytotita").ToString
            txtAFM.Text = ds.Tables("Paragogos").Rows(0).Item("AFM").ToString
            txtDOY.Text = ds.Tables("Paragogos").Rows(0).Item("DOY").ToString
            ddApalagiElga.SelectedValue =
ds.Tables("Paragogos").Rows(0).Item("ApalagiELGA").ToString
            txtApofasiELGA.Text =
ds.Tables("Paragogos").Rows(0).Item("ApofasiELGA").ToString
            ddFPA.SelectedValue = ds.Tables("Paragogos").Rows(0).Item("FPA").ToString
            txtEktasi.Text = ds.Tables("Paragogos").Rows(0).Item("Ektasi").ToString
            txtArithmKikloforias.Text =
ds.Tables("Paragogos").Rows(0).Item("ArithmKikloforias").ToString
            MyConnection.Close()
        End If


        If (WF_Activity = "UPDATE") Then
            btnKataxorisi.Visible = False
            btnDiagrafi.Visible = False
        End If

        If (WF_Activity = "DELETE") Then
            btnAnaneosi.Visible = False
            btnKataxorisi.Visible = False
            txtOnoma.Enabled = False
            txtEponimo.Enabled = False
            txtDieythinsi.Enabled = False
            txtTil1.Enabled = False
            txtTil2.Enabled = False
            txtTaytotita.Enabled = False
            txtAFM.Enabled = False
            txtDOY.Enabled = False
            ddApalagiElga.Enabled = False
            txtApofasiELGA.Enabled = False
            ddFPA.Enabled = False
            txtEktasi.Enabled = False
            txtArithmKikloforias.Enabled = False
        End If
        If (WF_Activity = "INSERT") Then
            btnAnaneosi.Visible = False
            btnDiagrafi.Visible = False
        End If
    End If
  End Sub
```

Όπως φαίνεται γίνεται έλεγχος τουWF_Activity και ανάλογα με την τιμή του εμφανίζονται ή κρύβονται κάποια στοιχεία της σελίδας. Ακόμα ελέγχεται αν είναι η πρώτη φορά που φορτώνεται η σελίδα και αν ισχύει αυτό, τότε δίνονται οι κατάλληλες τιμές στα διάφορα πεδία της σελίδας.

Ας δούμε τώρα τι γίνεται σε περίπτωση που πατήσουμε το κουμπί καταχώρηση.

```
 Private Sub btnKataxorisi_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnKataxorisi.Click

    If Page.IsValid Then

        MyCommand = New SqlCommand("spParagogosInsert", MyConnection)
        MyCommand.CommandType = CommandType.StoredProcedure

        Dim Onom As New SqlParameter("@Onoma", SqlDbType.NVarChar, 50)
        Dim Eponim As New SqlParameter("@Eponimo", SqlDbType.NVarChar, 50)
        Dim Dieythins As New SqlParameter("@Dieythinsi", SqlDbType.NVarChar, 255)
        Dim Ti1 As New SqlParameter("@Til1", SqlDbType.NVarChar, 50)
        Dim Ti2 As New SqlParameter("@Til2", SqlDbType.NVarChar, 50)
        Dim Taytotit As New SqlParameter("@Taytotita", SqlDbType.NVarChar, 50)
        Dim AF As New SqlParameter("@AFM", SqlDbType.NVarChar, 50)
        Dim D As New SqlParameter("@DOY", SqlDbType.NVarChar, 50)
        Dim ApalagiELG As New SqlParameter("@ApalagiELGA",
SqlDbType.NVarChar, 50)
        Dim ApofasiELG As New SqlParameter("@ApofasiELGA",
SqlDbType.NVarChar, 50)
        Dim FP As New SqlParameter("@FPA", SqlDbType.Int, 4)
        Dim Ektas As New SqlParameter("@Ektasi", SqlDbType.Float, 8)
        Dim ArithmKikloforia As New SqlParameter("@ArithmKikloforias",
SqlDbType.NVarChar, 50)
        Dim retva As New SqlParameter("@retval", SqlDbType.Int)

        Onom.Value = txtOnoma.Text
        Eponim.Value = txtEponimo.Text
        Dieythins.Value = txtDieythinsi.Text
        If txtTil1.Text = "" Then
           Ti1.Value = DBNull.Value
        Else
           Ti1.Value = txtTil1.Text
        End If
        If txtTil2.Text = "" Then
           Ti2.Value = DBNull.Value
        Else
           Ti2.Value = txtTil2.Text
        End If
```

```
Taytotit.Value = txtTaytotita.Text
AF.Value = txtAFM.Text
D.Value = txtDOY.Text
ApalagiELG.Value = ddApalagiElga.SelectedItem.Value
If txtApofasiELGA.Text = "" Then
   ApofasiELG.Value = DBNull.Value
Else
   ApofasiELG.Value = txtApofasiELGA.Text
End If
FP.Value = CInt(ddFPA.SelectedItem.Value)
If txtEktasi.Text = "" Then
   Ektas.Value = DBNull.Value
Else
   Ektas.Value = txtEktasi.Text
End If
If txtArithmKikloforias.Text = "" Then
   ArithmKikloforia.Value = DBNull.Value
Else
   ArithmKikloforia.Value = txtArithmKikloforias.Text
End If
retva.Direction = ParameterDirection.Output

With MyCommand.Parameters
   .Add(Onom)
   .Add(Eponim)
   .Add(Dieythins)
   .Add(Ti1)
   .Add(Ti2)
   .Add(Taytotit)
   .Add(AF)
   .Add(D)
   .Add(ApalagiELG)
   .Add(ApofasiELG)
   .Add(FP)
   .Add(Ektas)
   .Add(ArithmKikloforia)
   .Add(retva)
End With

MyCommand.Connection.Open()
MyCommand.ExecuteNonQuery()
MyCommand.Connection.Close()

If CInt(MyCommand.Parameters("@retval").Value) > 0 Then
   Response.Redirect("success.aspx")
Else
   Response.Redirect("failed.aspx")
```

```
        End If
    End If
End Sub
```

Τι είναι λοιπόν αυτό που κάνει το παραπάνω snippet κώδικα; Καταρχήν ελέγχει αν η σελίδα είναι έγκυρη δηλαδή αν έχουν συμπληρωθεί τα απαραίτητα πεδία. Αν είναι έγκυρη τότε περνάει τις τιμές των πεδίων σε παραμέτρους τις οποίες στέλνει στη βάση δεδομένων η οποία με την σειρά της εκτελεί μια αποθηκευμένη διαδικασία για την εισαγωγή μίας νέας εγγραφής παραγωγού με τα στοιχεία της φόρμας. Η σύνδεση με τον SQL Server γίνεται μέσω του ενός Connection String το οποίο έχουμε αρχικοποιήσει στο αρχείο web.config ώστε αν το αλλάξουμε σε αυτό το αρχείο η αλλαγή να γίνει γνωστή σε όλες τις σελίδες της εφαρμογής. Αν όλες αυτές η διαδικασίες εκτελεστούν σωστά τότε εμφανίζεται η σελίδα success.aspx που ενημερώνει το χρήστη ότι η καταχώρηση έγινε με επιτυχία, ενώ σε διαφορετική περίπτωση εμφανίζεται η σελίδα failed.aspx η οποία ενημερώνει ότι η διεργασία δεν εκτελέστηκε.

Η αποθηκευμένη διαδικασία (stored procedure) που εκτελείται κατά την καταχώρηση παραγωγού είναι η εξής:

```
CREATE PROCEDURE spParagogosInsert
(
@Onoma nvarchar(50), @Eponimo nvarchar(50), @Dieythinsi nvarchar(255), @Til1
nvarchar(50), @Til2 nvarchar(50), @Taytotita nvarchar(50),@AFM nvarchar(50), @DOY
nvarchar(50), @ApalagiELGA nvarchar(50), @ApofasiELGA nvarchar(50), @FPA int,
@Ektasi float, @ArithmKikloforias nvarchar(50), @retval int output
)

AS INSERT INTO Paragogos
(
Onoma,
Eponimo,
Dieythinsi,
Til1,
Til2,
Taytotita,
AFM,
```

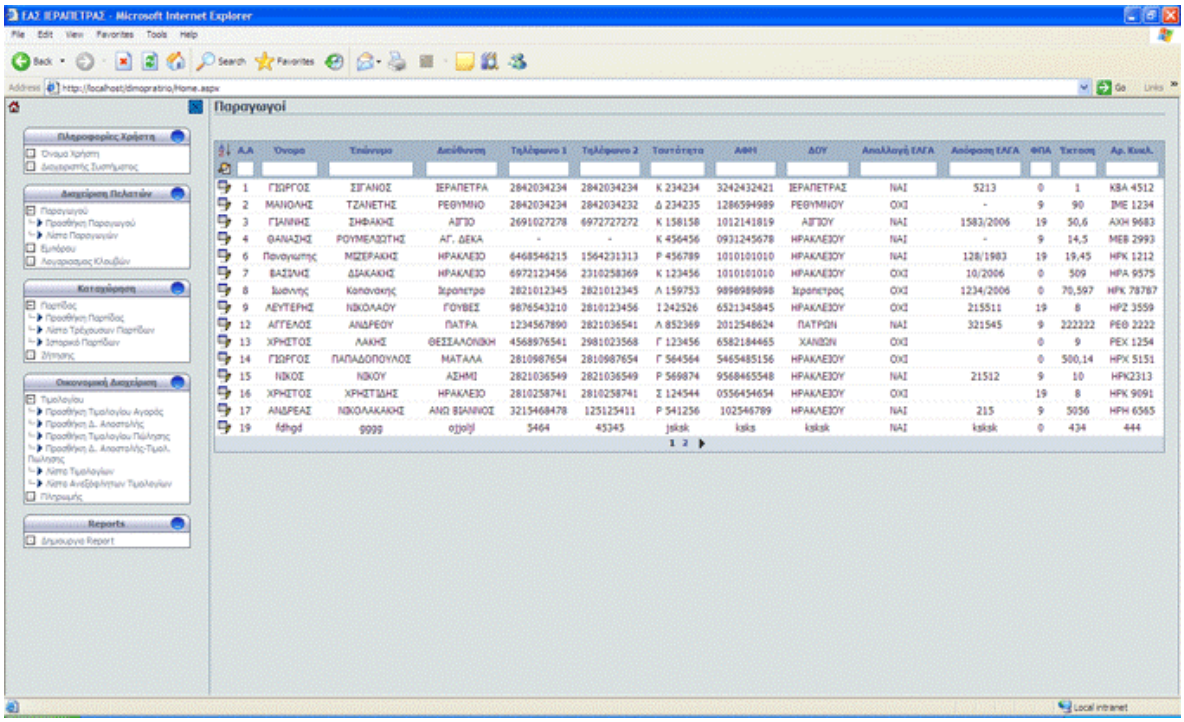
```
DOY,
ApalagiELGA,
ApofasiELGA,
FPA,
Ektasi,
ArithmKikloforias
)

VALUES
(
@Onoma,
@Eponimo,
@Dieythinsi,
@Til1,
@Til2,
@Taytotita,
@AFM,
@DOY,
@ApalagiELGA,
@ApofasiELGA,
@FPA,
@Ektasi,
@ArithmKikloforias
)

   SELECT @retval = @@IDENTITY
GO
```

Η παραπάνω αποθηκευμένη διαδικασία καταχωρεί τις τιμές στην βάση και αν όλα γίνουν σωστά επιστρέφει στην εφαρμογη το id της νέας έγγραφης, ώστε να ενημερώνεται ο χρηστής για τη επιτυχία ή αποτυχία της διεργασίας

Σε αντιστοιχία με αυτή την περίπτωση λειτουργούν και οι υπόλοιπες φόρμες της εφαρμογής που χρησιμοποιούνται για την εισαγωγή – ενημέρωση – διαγραφή δεδομένων στην βάση. Βεβαια χρησιμοποιείται πιο πολύπλοκος κώδικας για την εκτέλεση σκοπών όπως είναι το ανέβασμα αρχείων στο server, η προεπισκόπηση αρχείων που βρίσκονται στο server και η εισαγωγή – ενημέρωση – διαγραφή δεδομένων μέσα απο φόρμες master-detail.

Ας δούμε και ένα παράδειγμα μιας φόρμας που χρησιμοποιείται ως λίστα για την εμφάνιση των περιεχομένων της βάσης δεδομένων. Έστω λοιπόν ότι απο το menu

επιλέγουμε τη διεργασία Λίστα Παραγωγών. Επιλέγοντας την διεργασία στο δεξιό frame θα ανοίξει η σελίδα Paragogoslist.aspx.



Εικόνα 7. Η οθόνη για την Λίστα Παραγωγών

Εδώ ο χρήστης μπορεί να δει τις εγγραφές που υπάρχουν στον πίνακα των παραγωγών στη βάση. Μπορεί ακόμα ψάξει για αποτελέσματα σχετικά με τα κριτήρια που θα εισάγει. Ακόμα μπορεί να δει τα αποτελέσματα ταξινομημένα κατά πεδίο και σε αύξουσα η φθίνουσα σειρά. Ακόμα μπορεί πατώντας τα κουμπιά στα αριστερά της λίστας να τροποποιήσει ή να διαγράψει μια εγγραφή.

Ας εστιάσουμε τώρα σε κάποια σημεία του κώδικα υποστήριξης για την σελίδα Paragogoslist.aspx.

Πρώτα απ'ολα ας δούμε πως γεμίζει η λίστα με δεδομένα

```
Sub BindGrid()
    If Session("parfilparagogos") Is Nothing Then
      'Trexei mono tin proti fora
      ds = New DataSet
      Dim sCommand As String

      If viewstate("sortfield") Is Nothing Then
        sCommand = "select * from " & tablename
      Else
        sCommand = "select * from " & tablename & " order by " &
viewstate("sortfield") & " " & viewstate("sortdirection") & ""
```

```
        End If

        MyDataAdapter = New SqlDataAdapter(sCommand, MyConnection)
        MyDataAdapter.Fill(ds, tablename)
        Datagrid1.DataSource = ds
        Datagrid1.DataMember = tablename
        Datagrid1.DataBind()

    Else

        Dim mycom As String = CType(Session.Item("parmycom"), String)

        dsfilter = New DataSet

        Dim sCommand As String

        If viewstate("sortfield") Is Nothing Then
            sCommand = mycom
        Else
            sCommand = mycom + "order by " & viewstate("sortfield") & " " &
viewstate("sortdirection") & ""

        End If

        MyDataAdapter = New SqlDataAdapter(sCommand, MyConnection)
        MyDataAdapter.Fill(dsfilter, tablename)
        Datagrid1.DataSource = dsfilter
        Datagrid1.DataMember = tablename
        Datagrid1.DataBind()
    End If
  End Sub
```

Το παραπάνω κομμάτι ελέγχει αν κάποιο εχει κριτήριο αναζήτησης. Αν εχει επιστρέφει απο τη βάση τις εγγραφές που ταιριάζουν στο κριτήριο. Αν δεν υπάρχει κάποιο πεδίο αναζήτησης συμπληρωμένο τότε επιστρέφει απο τη βάση όλες τις εγγραφές. Παράλληλα ελέγχει αν ο χρήστης εχει επιλέξει να ταξινομήσει τα αποτελέσματα ως προς κάποια στήλη, και αν ναι, τότε τα ταξινομεί ανάλογα.

Ας δούμε και το κομμάτι του κώδικα που εκτελείται όταν ο χρήστης πατήσει το κουμπί για την αναζήτηση μεταξύ των αποτελεσμάτων

```
Public Sub okclick(ByVal sender As Object, ByVal e As EventArgs) Handles go.Click
    ' TODO:filter the results from the query and display the filtered ones
    Dim filterParagogosID As TextBox
    Dim filterOnoma As TextBox
```

```
        Dim filterEponimo As TextBox
        Dim filterDieythinsi As TextBox
        Dim filterTil1 As TextBox
        Dim filterTil2 As TextBox
        Dim filterTaytotita As TextBox
        Dim filterAFM As TextBox
        Dim filterDOY As TextBox
        Dim filterApalagiELGA As TextBox
        Dim filterApofasiELGA As TextBox
        Dim filterFPA As TextBox
        Dim filterEktasi As TextBox
        Dim filterArithmKikloforias As TextBox

        Dim myCommand As String
        dsfilter = New DataSet
        Datagrid1.CurrentPageIndex = 0
        filterbar = go.Parent.Parent
        viewstate("sortfield") = Nothing

        MyDataAdapter = New SqlDataAdapter(myCommand, MyConnection)
        filterParagogosID = filterbar.Controls(2).Controls(0)
        filterOnoma = filterbar.Controls(3).Controls(0)
        filterEponimo = filterbar.Controls(4).Controls(0)
        filterDieythinsi = filterbar.Controls(5).Controls(0)
        filterTil1 = filterbar.Controls(6).Controls(0)
        filterTil2 = filterbar.Controls(7).Controls(0)
        filterTaytotita = filterbar.Controls(8).Controls(0)
        filterAFM = filterbar.Controls(9).Controls(0)
        filterDOY = filterbar.Controls(10).Controls(0)
        filterApalagiELGA = filterbar.Controls(11).Controls(0)
        filterApofasiELGA = filterbar.Controls(12).Controls(0)
        filterFPA = filterbar.Controls(13).Controls(0)
        filterEktasi = filterbar.Controls(14).Controls(0)
        filterArithmKikloforias = filterbar.Controls(15).Controls(0)

        'TODO:Filter in multiple fields

        sCommand = "select * from " & tablename & " where 1=1"
        If ((filterParagogosID.Text <> "")) Then
            Session("parstrParagogosID") = filterParagogosID.Text
            sCommand = sCommand + " and ParagogosID  like '%" &
(CType(Session.Item("parstrParagogosID"), String)) & "%'"
        Else
            Session("parstrParagogosID") = Nothing
        End If
        If ((filterOnoma.Text <> "")) Then
            Session("parstrOnoma") = filterOnoma.Text
```

```
            sCommand = sCommand + "and Onoma  like '%" &
(CType(Session.Item("parstrOnoma"), String)) & "%'"
        Else
            Session("parstrOnoma") = Nothing
        End If
        If ((filterEponimo.Text <> "")) Then
            Session("parstrEponimo") = filterEponimo.Text
            sCommand = sCommand + "and Eponimo  like '%" &
(CType(Session.Item("parstrEponimo"), String)) & "%'"
        Else
            Session("parstrEponimo") = Nothing
        End If
        If ((filterDieythinsi.Text <> "")) Then
            Session("parstrDieythinsi") = filterDieythinsi.Text
            sCommand = sCommand + "and Dieythinsi  like '%" &
(CType(Session.Item("parstrDieythinsi"), String)) & "%'"
        Else
            Session("parstrDieythinsi") = Nothing
        End If
        If ((filterTil1.Text <> "")) Then
            Session("parstrTil1") = filterTil1.Text
            sCommand = sCommand + "and Til1  like '%" &
(CType(Session.Item("parstrTil1"), String)) & "%'"
        Else
            Session("parstrTil1") = Nothing
        End If
        If ((filterTil2.Text <> "")) Then
            Session("parstrTil2") = filterTil2.Text
            sCommand = sCommand + "and Til2  like '%" &
(CType(Session.Item("parstrTil2"), String)) & "%'"
        Else
            Session("parstrTil2") = Nothing
        End If
        If ((filterTaytotita.Text <> "")) Then
            Session("parstrTaytotita") = filterTaytotita.Text
            sCommand = sCommand + "and Taytotita  like '%" &
(CType(Session.Item("parstrTaytotita"), String)) & "%'"
        Else
            Session("parstrTaytotita") = Nothing
        End If
        If ((filterAFM.Text <> "")) Then
            Session("parstrAFM") = filterAFM.Text
            sCommand = sCommand + "and AFM  like '%" &
(CType(Session.Item("parstrAFM"), String)) & "%'"
        Else
            Session("parstrAFM") = Nothing
        End If
        If ((filterDOY.Text <> "")) Then
```

```
            Session("parstrDOY") = filterDOY.Text
            sCommand = sCommand + "and DOY  like '%" &
(CType(Session.Item("parstrDOY"), String)) & "%'"
        Else
            Session("parstrDOY") = Nothing
        End If
        If ((filterApalagiELGA.Text <> "")) Then
            Session("parstrApalagiELGA") = filterApalagiELGA.Text
            sCommand = sCommand + "and ApalagiELGA  like '%" &
(CType(Session.Item("parstrApalagiELGA"), String)) & "%'"
        Else
            Session("parstrApalagiELGA") = Nothing
        End If
        If ((filterApofasiELGA.Text <> "")) Then
            Session("parstrApofasiELGA") = filterApofasiELGA.Text
            sCommand = sCommand + "and ApofasiELGA  like '%" &
(CType(Session.Item("parstrApofasiELGA"), String)) & "%'"
        Else
            Session("parstrApofasiELGA") = Nothing
        End If
        If ((filterFPA.Text <> "")) Then
            Session("parstrFPA") = filterFPA.Text
            sCommand = sCommand + "and FPA  like '%" &
(CType(Session.Item("parstrFPA"), String)) & "%'"
        Else
            Session("parstrFPA") = Nothing
        End If
        If ((filterEktasi.Text <> "")) Then
            Session("parstrEktasi") = filterEktasi.Text
            sCommand = sCommand + "and Ektasi  like '%" &
(CType(Session.Item("parstrEktasi"), String)) & "%'"
        Else
            Session("parstrEktasi") = Nothing
        End If
        If ((filterArithmKikloforias.Text <> "")) Then
            Session("parstrArithmKikloforias") = filterArithmKikloforias.Text
            sCommand = sCommand + "and ArithmKikloforias  like '%" &
(CType(Session.Item("parstrArithmKikloforias"), String)) & "%'"
        Else
            Session("parstrArithmKikloforias") = Nothing
        End If
        MyDataAdapter = New SqlDataAdapter(sCommand, MyConnection)
        MyDataAdapter.Fill(dsfilter, tablename)
        Datagrid1.DataSource = dsfilter
        Datagrid1.DataMember = tablename
        Datagrid1.DataBind()
```

```
        Session("parmycom") = sCommand
        Session("parfilparagogos") = "notnothing"


    End Sub
```

Ο παραπάνω κώδικας ελέγχει αν ο χρήστης εχει πληκτρολογήσει σε κάποιο textbox αναζήτησης κάποια τιμή. Αν αυτό ισχύει τότε προσθέτει στο query που χρησιμοποιείται για το γέμισμα της λίστας την αντίστοιχη παράμετρο.

Ο κώδικας που ασχολείται με την ταξινόμηση είναι ο εξής:

```
Public Sub SortCommand(ByVal sender As Object, _
  ByVal e As System.Web.UI.WebControls.DataGridSortCommandEventArgs)

        'krataw to viewstate gia na doulepsei to sorting mazi me to paging

        viewstate.Add("sortfield", e.SortExpression)
        If viewstate("sortdirection") Is Nothing Then
           viewstate.Add("sortdirection", "ASC")
        Else
           If viewstate("sortdirection") = "ASC" Then
              viewstate("sortdirection") = "DESC"
           Else
              viewstate("sortdirection") = "ASC"
           End If
        End If



        BindGrid()

    End Sub
```

Αυτό το πιο πάνω κώδικας ελέγχει με βάση ποια στήλη θα γίνει η ταξινόμηση και αν θα είναι αύξουσα ή φθίνουσα και ανάλογα ταξινομεί τα αποτελέσματα.

Τέλος ο κώδικας που ασχολείται με το paging είναι ο παρακάτω:

```
    Sub PageIndexChanged(ByVal sender As Object, ByVal e As EventArgs)
        'TODO:Handles the page button change

        Dim strCmdName As String = CType(sender, LinkButton).CommandName.ToUpper
        Dim strCmdArg As String = CType(sender,
LinkButton).CommandArgument.ToUpper
        Dim dsfilter As New DataSet
```

```
    Dim mycom As String = CType(Session.Item("parmycom"), String)
    Select Case strCmdName
       Case "NEXT"
          Datagrid1.CurrentPageIndex += 1
       Case "PREV"
          Datagrid1.CurrentPageIndex -= 1
       Case "PAGE"
          Datagrid1.CurrentPageIndex = CType(strCmdArg, Integer) - 1
    End Select

    BindGrid()

 End Sub
```

Η διαδικασία της δημοπρασίας εχει το εξής σκεπτικό. Καταρχήν ελέγχουμε ποιες είναι οι υπάρχουσες προσφορές καθως και ποιες είναι οι υπάρχουσες ζητήσεις που αντιστοιχούν σε αυτές τις προσφορές. Μετά ελέγχουμε ανάμεσα στις ζητήσεις για μια συγκεκριμένη προσφορά ποια εχει την μεγαλύτερη τιμή ζήτησης. Αν έχουν όλες την ίδια τιμή ζήτησης τότε ελέγχουμε ποια απο τις ζητήσεις ζητά τις περισσότερες κλούβες. Επόμενο κριτήριο είναι η προτεραιότητα του κάθε εμπόρου. Τέλος αν και αυτό το κριτήριο δεν καταχωρεί την δημοπρασία το τελευταίο κριτήριο είναι το ποια ζήτηση καταχωρήθηκε πρώτη.

Για την εκτέλεση της δημοπρασίας δημιουργήσαμε ένα αρχείο dll το οποίο κάθε φορά αν υπάρχουν οι κατάλληλες εγγραφές τρέχει τον αλγόριθμο της δημοπρασίας και καταχωρεί τα αποτελέσματα στη βάση δεδομένων.

4 ΑΞΙΟΛΟΓΗΣΗ

Επόμενο βήμα της διαδικασίας είναι το τεσταρισμα της εφαρμογής και η γενικότερη αξιολόγηση της.

Αφού δημιουργήσαμε το release version της εφαρμογής την εγκαταστήσαμε σε server με λειτουργικό συστημα windows server 2003 και με βάση δεδομένων SqlServer 2000. Τεστάραμε την εφαρμογη σε τοπικό δίκτυο με 10 σταθμούς εργασίας. Αφού κάναμε σε κάθε ένα σταθμό εργασίας login στην εφαρμογη, αρχίσαμε να εκτελούμε ταυτόχρονα διεργασίες σε κάθε σταθμό ξεχωριστά και να ελέγχουμε τα δεδομένα που δημιουργούνταν στη βάση δεδομένων. Τα αποτελέσματα του testing ήταν απολύτως ικανοποιητικά αφού δεν σημειώθηκε το παραμικρό πρόβλημα στην εκτέλεση των διεργασιών και τα δεδομένα της βάσης ήταν ακριβή, πράγμα που ήταν αναμενόμενο αφού η 3-tier αρχιτεκτονική της εφαρμογής διασφαλίζει την αξιόπιστη λειτουργία της εφαρμογής όταν αυτή χρησιμοποιείται ταυτόχρονα απο πολλους χρήστες.

Έτσι η εταιρία αποκομίζει απο την εφαρμογη τα παρακάτω οφέλη:

Η διαδικασία των λογιστικών εγγραφών γίνεται απλή, εύκολη, και πολύ γρήγορη.

Αξιοποιούνται οι λύσεις που προσφέρει η εφαρμογή, εξασφαλίζοντας μείωση του χρόνου που απαιτείται για την ολοκλήρωση απλών, καθημερινών εργασιών.

Απλοποιείται η έκδοση των τιμολογίων, καθώς ο τρόπος λειτουργίας του συστήματος προστατεύει τον χρήστη από ενδεχόμενα λάθη κατά την καταχώρηση των στοιχείων. Εάν δεν υπάρχει ένα είδος ή εάν κάποιο από τα δεδομένα είναι λάθος, η ίδια η εφαρμογή σταματάει την επεξεργασία και δεν εκδίδει το τιμολόγιο.

Εξοικονόμηση πόρων υλικών πόρων, γιατί οι databases, όπως και η πλειονότητα των ψηφιακών εφαρμογών, δεν κοστίζουν (πέρα από το κόστος των αδειών λογισμικού). Αυτό σημαίνει ότι κάποιος μπορεί να δημιουργήσει μια τεράστια βάση δεδομένων χωρίς να δαπανήσει ούτε ένα cent για την αγορά χαρτικών, μελανιών, διορθωτικών και συναφών

αναλώσιμων. Τα bits (το δομικό υλικό του ψηφιακού κόσμου) δεν κοστίζουν, είναι δεκτικά άμεσων αλλαγών και χωρούν σε ορισμένα εκατοστά του μέτρου, σε αντίθεση με τα φυσικά δεδομένα που αφενός κοστίζουν και αφετέρου η διόρθωσή τους είναι δύσκολη και επίπονη, η δε αποθήκευσή τους καταλαμβάνει πολύ χώρο.

5 ΣΥΜΠΕΡΑΣΜΑΤΑ

Η ανάπτυξη πλατφόρμων ηλεκτρονικού εμπορίου με τη χρήση της τεχνολογίας asp.net τριών επιπέδων που βασίζεται στα components ενέχει τα εξής οφέλη:

Επαναχρησιμοποιούμενος κώδικας Ένα component μπορεί εύκολα να χρησιμοποιηθεί από τον προγραμματιστή που το ανέπτυξε ή από κάποιον άλλο. Ένα component σχεδιασμένο για πρόσβαση σε μια βάση δεδομένων παραδείγματος χάριν, μπορεί να χρησιμοποιηθεί από πολλές διαφορετικές εφαρμογές, αν έχουν περίπου τις ίδιες ανάγκες πρόσβασης σε δεδομένα.

Μειωμένη πολυπλοκότητα Δημιουργώντας components, ένας προγραμματιστής 'κρύβει' την πολυπλοκότητα του κώδικά του. Η μόνη γνώση που χρειάζεται να έχει κάποιος άλλος προγραμματιστής για να χρησιμοποιήσει το component του είναι γνώση των μεθόδων (methods – functions) του component. Πρέπει δηλαδή να ξέρει μόνο τι πληροφορίες πρέπει να δώσει στο component και τι πληροφορίες να περιμένει από το component.

Ευκολότερη συντήρηση Χρησιμοποιώντας components, γίνεται πιο εύκολη η συντήρηση του κώδικα. Αν παραδείγματος χάριν κάποιος προγραμματιστής φτιάξει ένα component που υλοποιεί κάποιους επιχειρησιακούς κανόνες και οι κανόνες αυτοί κάποια στιγμή αλλάξουν, αρκεί η ενημέρωση αυτού του συγκεκριμένου component. Η υπόλοιπη εφαρμογή μένει άθικτη.

Ευκολότερη ανάπτυξη Διαχωρίζοντας τον κώδικα μιας εφαρμογής σε components, δίνει την δυνατότητα ανάπτυξης και ελέγχου (testing) μικρών, ανεξάρτητων κομματιών της εφαρμογής. Αυτό έχει σαν αποτέλεσμα και την διευκόλυνση των προγραμματιστών αλλά και την ευκολότερη συνεργασία μεταξύ τους.

Απεξάρτηση από συγκεκριμένες γλώσσες προγραμματισμού Σε οποιαδήποτε γλώσσα κι αν γραφτεί ένα component, διατηρεί την ικανότητα να επικοινωνεί με άλλα components, που ίσως είναι γραμμένα σε διαφορετική γλώσσα.

Ευελιξία εφαρμογής Με την ανοικτή αρχιτεκτονική είναι εύκολη η συνεργασία της εφαρμογής με άλλα πιθανά συστήματα λογισμικού (π.χ. ERP, CRM, λογιστικά πακέτα, κ.λπ.) που χρησιμοποιεί η επιχείρηση.

6 ΒΙΒΛΙΟΓΡΑΦΙΑ

1) ASP.NET Step by Step, Microsoft Press, Εκδόσεις Microsoft Press
2) ASP.NET Web Developer's Guide, Mesbah Ahmed-Chris Garrett-Jeremy Faircloth-Chris Payne-DotThatCom.com-Wei Meng Lee-Jonothon Ortiz, e-book http://www.syngress.com/
3) Beginning ASP.Net 1.1 E-Commerce Cristian Darie, Karli Watson, Εκδόσεις APress
4) Building Web Solutions with ASP.NET and ADO.NET, Dino Esposito, Εκδόσεις Microsoft Press
5) Microsoft SQL Server 2000: A Guide to Enhancements and New Features, Rahul Sarma, Εκδόσεις Addison Wesley Professional
6) Professional ASP.NET 1.0, Special Edition, Εκδόσεις WROX
7) Programming SQL Server 2000 with Visual Basic.NET, Rick Dobson, Εκδόσεις Microsoft Press
8) http://msdn.microsoft.com/
9) http://support.microsoft.com/
10) http://www.e-bi.gr/
11) http://www.ebusinessforum.gr/
12) http://www.go-online.gr/