ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή Εργασία

«Ανάπτυξη Εφαρμογής Διαχείρισης Παραγωγής»

της φοιτήτριας

Χαραλάμπους Φωτεινής

Αρ. Μητρώου:05/2949

Επιβλέπων Καθηγητής

κ. Γιώργος Πούλακας

Θεσσαλονίκη 2010

Πρόλογος

Η παρούσα εφαρμογή με τίτλο «Διαχείριση Παραγωγής» αναπτύχθηκε στα πλαίσια της πτυχιακής μου εργασίας για το τμήμα Πληροφορικής του Αλεξάνδρειου Τεχνολογικού Εκπαιδευτικού Ιδρύματος Θεσσαλονίκης.

Το πρόβλημα μου παραδόθηκε τον Οκτώβριο του 2009 μετά από αίτηση που είχα κάνει. Από τότε είχα στην διάθεση μου για την ολοκλήρωση της εφαρμογής ένα χρόνο. Αυτό που έπρεπε να κάνω ήταν να φτιάξω μια εφαρμογή που να διαχειρίζεται την παραγωγή ενός εργοστασίου ή μιας βιομηχανίας. Όταν άρχισα να σκέφτομαι το πώς θα το κάνω αυτό είχα πολλές ιδέες. Δεν ήθελα να περιοριστώ μόνο σε αυτό που ζητούσε το πρόβλημα, ήθελα η εφαρμογή να κάνει επιπλέον πράγματα. Στην αρχή νόμιζα ότι αυτά που ήθελα να κάνω ήταν απλά, μικρά πραγματάκια που θα εμπλούτιζαν απλά την εφαρμογή μου. Σιγά – σιγά όμως κατάλαβα ότι αυτό που σκεφτόμουν να κάνω δεν ήταν απλά η διαχείριση της παραγωγής ενός εργοστασίου, αλλά η διαχείριση ολόκληρου του εργοστασίου. Έτσι αποφάσισα ότι θα περιοριστώ μόνο στη λύση του προβλήματος. Δεν σταμάτησα όμως να αφήνω την φαντασία μου ελεύθερη στο τι περισσότερο μπορώ να κάνω. Δυστυχώς ο χρόνος δεν με άφησε να υλοποιήσω όλα όσα είχα σκεφτεί. Με άφησε όμως να ολοκληρώσω το πρόβλημα της εργασίας, αυτό τουλάχιστον θέλω να πιστεύω.

Προσπαθούσα στην εφαρμογή να κάνω όσο το δυνατό περισσότερους ελέγχους γίνεται ώστε να μειώσω όσο μπορώ την δυνατότητα στον οποιονδήποτε χρήστη της να κάνει λάθη. Αυτό όμως πολλές φορές προκάλεσε πολλές δυσκολίες στον κώδικα της εφαρμογής και απαιτούσε χρόνο. Με τον καιρό άρχισα να πιστεύω ότι δεν μπορώ να ελέγξω όλα τα πιθανά λάθη που μπορεί να κάνει κάποιος και να προσπαθήσω μέσω του κώδικα να τα αποφύγω. Έκανα λοιπόν τους πιο σημαντικούς κατά την γνώμη μου ελέγχους ώστε να βάλω κάπου ένα τέλος για να μην χαθώ μέσα σε αυτούς.

Το σημαντικό σε μια εφαρμογή είναι να λειτουργεί σωστά, δηλαδή τα αποτελέσματα που βγάζει και οι υπολογισμοί που τυχόν κάνει να είναι σωστοί. Το πρόβλημα εδώ ήταν ότι δεν ήξερα καλά το πώς λειτουργούσε ένα εργοστάσιο και η παραγωγή του. Προσπάθησα όμως να μαζέψω όσες πληροφορίες

μπορούσα από όσους νόμιζα ότι ήξεραν έστω και λίγα πράγματα για το πώς λειτουργεί μια παραγωγή εργοστασίου. Διάβασα αρκετές φορές το πρόβλημα και την περιγραφή του και προσπάθησα να συνδυάσω αυτά που έμαθα ρωτώντας ώστε να δημιουργήσω την εφαρμογή.

Για να δοκιμάσω κατά πόσο λειτουργούν αυτά που είχα κάνει, έτρεχα την εφαρμογή και προσπαθούσα να δημιουργήσω λάθη ώστε να δω τι προβλήματα δημιουργούνται και να τα διορθώσω. Πολλές φορές όμως όταν ξέρεις πώς λειτουργεί μια εφαρμογή δεν μπορείς να κάνεις λάθη που θα μπορούσε να κάνει ένας χρήστης. Αποφάσισα λοιπόν να δώσω σε φίλους μου την εφαρμογή να την δοκιμάσουν και παρατήρησα ότι αυτοί «μπορούσαν» να βρουν λάθη και προβλήματα όταν εγώ δεν μπορούσα. Οι δοκιμές σταμάτησαν όταν πλέον δεν μπορούσαν ούτε αυτοί να βρουν λάθη. Οι ατέλειες και τα προβλήματα μιας εφαρμογής όμως εμφανίζονται κατά την χρήση της σε πραγματικά γεγονότα. Φυσικά υπάρχουν πράγματα που αν τα έφτιαχνα ξανά ίσως να τα έκανα διαφορετικά, κάποια θα μπορούσα να τα φτιάξω καλύτερα και πράγματα που δεν έγινα και μπορούσαν να γίνουν.

Περίληψη

Ένα από τα προβλήματα που αντιμετωπίζει ένα εργοστάσιο είναι η διαχείριση των πρώτων υλών του και η παραγωγή των προϊόντων του. Αυτό ακριβώς το πρόβλημα είχε σαν στόχο να επιλύσει η εφαρμογή αυτή.

Η διαχείριση μιας παραγωγής θα πρέπει να προσφέρει ένα πλήθος από λειτουργίες. Θα πρέπει να διαχειρίζεται την αποθήκη του εργοστασίου και τις πρώτες ύλες του. Για κάθε προϊόν που παράγεται από το εργοστάσιο πρέπει να δίνεται η δυνατότητα διαχείρισης της συνταγής παραγωγής του. Επιπλέον οι εντολές παραγωγής που δημιουργούνται πρέπει να παρακολουθούνται από τους χρήστες της εφαρμογής. Τέλος για την έγκαιρη παράδοση από τους προμηθευτές των πρώτων υλών θα πρέπει να παρουσιάζονται καταστάσεις ελλείψεων των πρώτων υλών.

Το πρόβλημα της διαχείρισης παραγωγής προσπάθησε να λύσει η δημιουργία της εφαρμογής αυτής. Οι λειτουργίες που απαιτεί μια διαχείριση παραγωγής έχουν υλοποιηθεί ενώ παράλληλα υλοποιήθηκαν και κάποιες άλλες μικρές λειτουργίες για να προσφέρουν μερική ακόμα βοήθεια στους χρήστες της.

Abstract

One of the problems of a factory is the management of raw materials and production of products. This is exactly what the problem was intended to solve the application.

A production management should offer a multitude of functions. Should be managing the raw materials of factory's warehouse. For each product that factory produce should allow management of prescription production. Also, the production orders generated should be monitored by users of the application. Finally for the timely delivery by suppliers of raw materials should be presented situations shortages of raw materials.

This application tried to solve the problem of production management. The functions required of a production management have been implemented as some other small features too, to offer some more assistance to users.

Ευχαριστίες

Με την ολοκλήρωση της πτυχιακής μου εργασίας θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου, κ. Γιώργο Πούλακα, για την βοήθεια του στην κατανόηση του προβλήματος και για τις παρατηρήσεις ως προς την εφαρμογή. Κυρίως όμως θα ήθελα να τον ευχαριστήσω για την καθοδήγηση του στη σύνταξη του κειμένου της εφαρμογής.

Επιπλέον νοιώθω την ανάγκη να ευχαριστήσω το φίλο και συμφοιτητή μου, Ιωαννίδη Σάββα, για το χρόνο που αφιέρωσε στη δοκιμή της εφαρμογής και τα λάθη που σχεδόν πάντα «έκανε» την εφαρμογή να δημιουργεί.

Τέλος, ένα μεγάλο ευχαριστώ στους γονείς μου, που με πρόσφεραν την ευκαιρία να σπουδάσω. Χωρίς αυτούς δεν θα μπορούσα να φτάσω στο σημείο να ολοκληρώσω μια πτυχιακή εργασία.

Περιεχόμενα

Εισαγωγή

Το πρόβλημα που είχε σαν στόχο να επιλύσει η εφαρμογή αυτή ήταν η διαχείριση της παραγωγής ενός εργοστασίου ή μιας βιομηχανίας. Το κύριο πρόβλημα που έχει η παραγωγή ενός εργοστασίου είναι να μπορέσει να διαχειριστεί τις πρώτες ύλες της ώστε να παράγει προϊόντα.

Η εφαρμογή αυτή δεν θα προσπαθήσει να διαχειριστεί ολόκληρη την λειτουργία μιας βιομηχανίας αλλά θα δώσει έμφαση σε ένα συγκεκριμένο κομμάτι, στην παραγωγή. Για να μπορέσει μια εφαρμογή να θεωρηθεί ότι διαχειρίζεται μια παραγωγή θα πρέπει να εκτελεί τουλάχιστον κάποιες σημαντικές λειτουργίες. Θα πρέπει να διαχειρίζεται την αποθήκη, να διαχειρίζεται τα προϊόντα που παράγονται και τις συνταγές παραγωγής τους. Επιπλέον θα πρέπει να μπορεί να δημιουργήσει εντολές παραγωγής και να εμφανίσει καταστάσεις ελλείψεων στα είδη της αποθήκης.

Όλα αυτά μπορούν να εκτελεστούν και διαχειριστούν από την παρούσα εφαρμογή. Ο χρήστης μπορεί να «παρακολουθήσει» τα διάφορα είδη του εργοστασίου, τις πρώτες ύλες και τα παραγόμενα προϊόντα, και να δει διάφορα χαρακτηριστικά για αυτά. Μπορεί να δημιουργήσει κάποιο είδος ή και να διαγράψει κάποιο ήδη υπάρχον. Έχει την δυνατότητα να παραλάβει πρώτες ύλες και να ενημερώσει την διαθεσιμότητα τους στην αποθήκη.

Κάθε είδος που ανήκει στην κατηγορία των παραγόμενων έχει την δυνατότητα να διαθέτει συνταγή παραγωγής. Αυτή την συνταγή παραγωγής κάθε είδους μπορεί ο χρήστης της εφαρμογής να την διαχειριστεί, να προσθέσει και να αφαιρέσει πρώτες ύλες ή να τροποποιήσει τις ποσότητες των υλών που χρειάζονται για να παραχθεί το είδος αυτό.

Για να μπορέσει ένα προϊόν να παραχθεί θα πρέπει να σταλεί στην παραγωγή μια εντολή παραγωγής του προϊόντος, η οποία θα εμφανίζει τα προϊόντα που ζητά ο χρήστης να παραχθούν μαζί με την ποσότητα κάθε προϊόντος. Επιπλέον εμφανίζει τις ύλες που χρειάζονται για την παραγωγή του κάθε προϊόντος καθώς και την συνολική ποσότητα τους. Ο χρήστης εκτός από το να δημιουργήσει μια

εντολή παραγωγής μπορεί επίσης να την τροποποιήσει ή και να την ακυρώσει αν αυτή δεν έχει εκτυπωθεί για να σταλεί στην παραγωγή.

Με κάθε εντολή παραγωγής οι πρώτες ύλες που υπάρχουν στην αποθήκη μειώνονται και έτσι μπορούν να δημιουργηθούν διάφορες ελλείψεις. Οι ελλείψεις αυτές μπορούν να εμφανιστούν από την εφαρμογή, και να εκτυπωθούν αν χρειαστεί, ώστε να μπορέσει η εταιρεία να προμηθευτεί έγκαιρα τις πρώτες ύλες της.

Τέλος, εκτός από τα παραπάνω που είναι βασικά, η εφαρμογή δίνει την δυνατότητα στο χρήστη να δημιουργήσει τις δικές του κατηγορίες ειδών στις οποίες θα ανήκουν τα είδη που θα καταχωρήσει, τις δικές του μονάδες μέτρησης με τις οποίες θα μετρούνται τα είδη και τέλος μπορεί να δημιουργήσει τις δικές του καταστάσεις στις οποίες μπορεί να βρίσκεται μια εντολή παραγωγής ώστε να έχει καλύτερη οπτική της παραγωγής.

Η εφαρμογή αυτή υλοποιήθηκε με την βοήθεια κάποιων εργαλείων. Στα εργαλεία αυτά γίνεται αναλυτική αναφορά μέσα στο κείμενο που ακολουθεί. Επιπλέον, εκτός από τα εργαλεία, για να μπορέσει η εφαρμογή να λειτουργήσει και να αποθηκεύει τα διάφορα δεδομένα που χρειάζονται, χρησιμοποίησε μια βάση δεδομένων, η οποία θα παρουσιαστεί και αυτή στη συνέχεια του κειμένου.

Κεφάλαιο 1
Εργαλεία Υλοποίησης της Εφαρμογής

Για την υλοποίηση της εφαρμογής χρειάστηκε να χρησιμοποιηθούν κάποια εργαλεία. Το ένα από αυτά ήταν το εργαλείο για την σύνταξη του κώδικα της εφαρμογής και το άλλο ήταν το εργαλείο για την διαχείριση της βάσης δεδομένων της. Στο κεφάλαιο αυτό θα γίνει αναφορά στα εργαλεία αυτά και στις δυνατότητες τους.

1.1 Visual Studio 2008 Professional

Το εργαλείο του Visual Studio 2008 είναι το εργαλείο που χρησιμοποιήθηκε για την σύνταξη του κώδικα της εφαρμογής. Το Visual Studio είναι ένα εργαλείο, το οποίο δημιουργήθηκε από την Microsoft, με το οποίο μπορείς να δημιουργήσεις διάφορες εφαρμογές, όπως εφαρμογές για Windows, για διαδίκτυο και ιστοσελίδες με κύριο χαρακτηριστικό τους το γραφικό και φιλικό περιβάλλον προς τον χρήστη (GUI). Το Visual Studio χαρακτηρίζεται σαν ένα ολοκληρωμένο περιβάλλον ανάπτυξης (integrated development environment - IDE) και οι εφαρμογές που δημιουργούνται σε αυτό μπορούν να λειτουργήσουν σε όλες τις πλατφόρμες που υποστηρίζονται από Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework και Microsoft Silverlight.

Σαν περιβάλλον το Visual Studio παρέχει από μόνο του ένα μεγάλο αριθμό από εργαλεία και δυνατότητες διευκολύνοντας τη δημιουργία γραφικών και φιλικών προς τον χρήστη εφαρμογών . Μια από τις δυνατότητες του είναι να δέχεται plug-ins τα οποία βελτιώνουν την λειτουργικότητα σχεδόν σε όλα τα επίπεδα καθώς επίσης και να δέχεται επιπρόσθετα διάφορες ομάδες εργαλείων όπως είναι οι editors και οι visual designer σε συγκεκριμένους τομείς ή γλώσσες ή εργαλεία για όλες τις πτυχές του κύκλου ανάπτυξης του λογισμικού.

Το Visual Studio δεν χρησιμοποιείται για την σύνταξη κώδικα μόνο σε μια γλώσσα προγραμματισμού. Ο προγραμματιστής εδώ μπορεί να επιλέξει σε ποια γλώσσα προγραμματισμού θέλει να αναπτύξει την εφαρμογή του χωρίς η μια

γλώσσα να μειονεκτεί από την άλλη αφού οι δυνατότητες που παρέχονται από το Visual Studio συμπεριλαμβάνονται σε όλες τις γλώσσες. Το γεγονός αυτό επιτυγχάνεται μέσω των γλωσσικών υπηρεσιών. Οι γλωσσικές αυτές υπηρεσίες επιτρέπουν στον επεξεργαστή κειμένου (Coder) και στον debugger να υποστηρίξουν (σε διαφορετικό βαθμό) σχεδόν κάθε γλώσσα προγραμματισμού με την προϋπόθεση ότι η υπηρεσία της συγκεκριμένης γλώσσας υπάρχει. Συνοπτικά οι γλώσσες προγραμματισμού που υποστηρίζονται από το Visual Studio είναι: η Visual C/C++, η Visual Basic.NET και η Visual C#. Επιπλέον σε επίπεδο διαδικτυακών εφαρμογών υποστηρίζονται η γλώσσες: HTML/XHTML, JavaScript και CSS. Τέλος από την πλευρά των βάσεων δεδομένων υποστηρίζεται η XML. Για την υλοποίηση της συγκεκριμένης εφαρμογής χρησιμοποιήθηκε η γλώσσα προγραμματισμού Visual Basic.NET.

Έχοντας πάρει μια πρώτη ιδέα για το Visual Studio στη συνέχεια θα προσπαθήσουμε να δούμε το Visual Studio λίγο πιο βαθιά.

1.1.1 Αρχιτεκτονική του Visual Studio

Είχε αναφερθεί παραπάνω ότι μέσω του Visual Studio ένας προγραμματιστής μπορεί να συντάξει κώδικα σε αρκετές γλώσσες. Στην ουσία το Visual Studio δεν υποστηρίζει από μόνο του καμία γλώσσα προγραμματισμού. Για να μπορέσει να υποστηρίξει αυτές τις γλώσσες προγραμματισμού συνδυάζει διάφορους τύπους λειτουργιών. Οι ειδικές αυτές λειτουργίες κωδικοποιούνται ως VSPackage. Όταν γίνει εγκατάσταση στο Visual Studio οι λειτουργίες αυτές είναι πλέον διαθέσιμες σαν υπηρεσία. Το Visual Studio σαν ένα integrated development environment – IDE παρέχει τρείς υπηρεσίες. Την υπηρεσία SVsSolution, η οποία παρέχει την

δυνατότητα στον προγραμματιστή να απαριθμεί τα έργα (projects) και τις λύσεις (Solutions) του. Την υπηρεσία SVsUIShell η οποία παρέχει την λειτουργικότητα του User Interface (UI) και των διάφορων παραθύρων που εμφανίζονται καθώς επίσης και των καρτελών, των γραμμών και των παράθυρων εργαλείων. Τέλος η υπηρεσία SVsShell η οποία ασχολείται με την καταγραφή των VSPackages. Το IDE δεν παρέχει μόνο αυτές τις τρείς υπηρεσίες αλλά είναι και υπεύθυνο για τον συντονισμό και την επικοινωνία μεταξύ των υπηρεσιών αυτών. Όλοι οι editors, οι designers, οι τύποι των Projects και τα υπόλοιπα εργαλεία είναι υλοποιημένα στο VSPackage. Το Visual Studio για να έχει πρόσβαση στο VSPackage, συνεπώς και σε όλα αυτά που υλοποιεί, χρησιμοποιεί το COM. Επιπλέον το Visual Studio SDK περιλαμβάνει και τη διαχείριση του πακέτου του Framework (Managed Package Framework – MPF) το οποίο είναι ένα σύνολο από διαχειριζόμενα wrappers γύρω από τις διασυνδέσεις του COM. Οι διασυνδέσεις αυτές επιτρέπουν στο VSPackage να είναι γραμμένο σε οποιανδήποτε συμβατή γλώσσα (CLI). Παρόλα αυτά όμως το MPF δεν παρέχει όλες τις λειτουργίες των διασυνδέσεων του Visual Studio COM. Όσες υπηρεσίες δεν παρέχονται χρησιμοποιούνται για την δημιουργία άλλων πακέτων τα οποία προσθέτουν λειτουργικότητα στο Visual Studio IDE.

Το VSPackage το οποίο χρησιμοποιεί το Visual Studio δεν είναι μόνο ένα για όλες τις γλώσσες προγραμματισμού. Αντιθέτως για κάθε γλώσσα προγραμματισμού υπάρχει ένα ειδικό VSPackage το οποίο ονομάζεται Language Service. Αυτό που κάνει ένα Language Service είναι να ορίζει διάφορα Interfaces τα οποία μπορεί να υλοποιήσει το VSPackage ώστε να παρέχει επιπρόσθετη υποστήριξη για διάφορες λειτουργίες. Οι λειτουργίες αυτές μπορούν να είναι διαθέσιμες για κάποια γλώσσα αν το interface έχει υλοποιηθεί (implemented) για την γλώσσα αυτή. Οι διάφορες εφαρμογές των Interfaces μπορούν να επαναχρησιμοποιούν κώδικα από τον parser ή τον compiler. Τα Language Services μπορούν να υλοποιηθούν είτε σε τοπικό κώδικα είτε σε διαχειριζόμενο κώδικα. Για τον τοπικό κώδικα μπορούν να χρησιμοποιηθούν είτε τα τοπικά interfaces του COM είτε το Babel Framework (ένα μέρος από το Visual Studio SDK). Για τον διαχειριζόμενο κώδικα μπορεί να χρησιμοποιηθεί το MPF το οποίο περιλαμβάνει wrappers για να διαχειρίζεται την σύνταξη των Language Services.

Ένα άλλο χαρακτηριστικό του Visual Studio είναι ότι υποστηρίζει την εκτέλεση πολλών στιγμιότυπων ενός περιβάλλοντος. Το κάθε ένα στιγμιότυπο από αυτά έχει το δικό του σύνολο VSPackages. Για να μπορέσει να λειτουργήσει όλο αυτό, τα διάφορα στιγμιότυπα, χρησιμοποιούν διαφορετικές ομάδες μητρώου ώστε να μπορέσουν να αποθηκεύσουν την δική τους διαμόρφωση και διαφοροποιούνται το ένα από το άλλο μέσω του Application ID τους. Πώς λειτουργεί τώρα όλο αυτό. Τα στιγμιότυπα που ξεκινούν από ένα συγκεκριμένο AppID.exe, το οποίο διαλέγει το Application ID τους, ορίζει μια κυψέλη της ρίζας και δρομολογεί το IDE. Τα VSPackages τα οποία καταχωρούνται για ένα AppID ενσωματώνονται σε άλλα VSPackages τα οποία καταχωρήθηκαν για το εν λόγω AppID. Οι διάφορες εκδόσεις του Visual Studio δημιουργούνται χρησιμοποιώντας διαφορετικά AppID. Για παράδειγμα το Visual Studio Express εγκαθίσταται δημιουργώντας ένα δικό του AppID ενώ οι υπόλοιπες εκδόσεις του Visual Studio έχουν τα ίδια AppID. Αυτό σημαίνει ότι κάποιος μπορεί να εγκαταστήσει την έκδοση Express του Visual Studio παράλληλα με κάποια άλλη έκδοση του. Δεν μπορεί όμως να εγκαταστήσει δύο άλλες εκδόσεις του (εκτός της Express) παράλληλα γιατί χρησιμοποιούν την ίδια εγκατάσταση.

1.1.2 Χαρακτηριστικά

Έχοντας παρακολουθήσει εν μέρει την αρχιτεκτονική του Visual Studio παρακάτω θα δούμε και μερικά από τα κυριότερα χαρακτηριστικά του.

Επεξεργαστής κώδικα (Code Editor)

Ο επεξεργαστής κώδικα που περιλαμβάνει το Visual Studio, όπως και οι επεξεργαστές που περιλαμβάνουν όλα τα IDE εργαλεία, υποστηρίζει τον τονισμό (επισήμανση) στην σύνταξη του κώδικα καθώς και την ολοκλήρωση του (αυτόματα) με την χρήστη του IntelliSense όχι μόνο για τις μεταβλητές, της συναρτήσεις και τις μεθόδους αλλά και για την δομή της γλώσσας προγραμματισμού όπως είναι τα διάφορα ερωτήματα και οι βρόχοι. Το IntelliSense παρέχεται όχι μόνο για τις γλώσσες προγραμματισμού που

περιλαμβάνονται αλλά και για το XML, το CSS, την JavaScript και γενικά όλα όσα χρησιμοποιούνται για την δημιουργία ιστοσελίδων και εφαρμογών διαδικτύου.

Ο επεξεργαστής κώδικα επιπλέον υποστηρίζει την τοποθέτηση διάφορων σελιδοδεικτών μέσα στον κώδικα ώστε να επιτρέπει την εύκολη πλοήγηση μέσα σε αυτόν. Επιπρόσθετα βοηθήματα για πλοήγηση μέσα στον κώδικα είναι τα διάφορα blocks κώδικα τα οποία επιτρέπονται καθώς και μια στοιχειώδη αναζήτηση. Ο προγραμματιστής μπορεί να αποθηκεύσει τμήματα κώδικα σαν πρότυπα τα οποία μπορεί να ξαναχρησιμοποιήσει τοποθετώντας τα στον κώδικα του και προσαρμόζοντας τα στο project το οποίο εργάζεται. Επιπλέον υπάρχει και ένα εργαλείο για την διαχείριση του κώδικα. Τα εργαλεία αυτά μπορούν να εμφανίζονται σαν ένα επιπρόσθετο παράθυρο, μπορούν να είναι κρυφά ή να είναι δεσμευμένα σε μια επιφάνεια της οθόνης.

Τέλος ο επεξεργαστής κώδικα μπορεί να παρέχει βοήθεια στον προγραμματιστή κατά την σύνταξη του κώδικα αφού αυτόματα αναγνωρίζει τυχόν λάθη και του τα επισημαίνει υπογραμμίζοντας τα με μια κόκκινη γραμμή. Επιπλέον υπογραμμίζει και κάποιες προειδοποιήσεις με μια πράσινη γραμμή. Έτσι μπορούν λάθη και απροσεξίες να διορθωθούν άμεσα.

Debugger

Ο Debugger (πρόγραμμα εντοπισμού σφαλμάτων) που περιλαμβάνει το Visual Studio λειτουργεί τόσο σε επίπεδο πηγαίου όσο και σε επίπεδο μηχανής. Επιπλέον λειτουργεί σε διαχειριζόμενο και τοπικό κώδικα και μπορεί να χρησιμοποιηθεί σε εφαρμογές που είναι γραμμένες σε οποιανδήποτε από τις γλώσσες που υποστηρίζει το Visual Studio. Παρέχεται επίσης η δυνατότητα καθώς τρέχουν κάποιες διεργασίες να παρακολουθούνται και να εντοπίζονται σφάλματα. Όταν ο πηγαίος κώδικας για την διεργασία που τρέχει είναι διαθέσιμός τότε ο κώδικας αυτό εμφανίζεται όταν η διεργασία τρέχει. Αν ο κώδικας της διεργασίας δεν είναι διαθέσιμος τότε μπορεί ο debugger να τον δείξει αλλά μόνο αποσυναρμολογημένο (disassembly). Ο Debugger του visual studio μπορεί να υποστηρίξει και πολυνηματικά προγράμματα καθώς επίσης και να ρυθμιστεί

ώστε να τρέξει όταν μια εφαρμογή που τρέχει έξω από το περιβάλλον του Visual Studio προκαλέσει σφάλμα.

Μια από της δυνατότητες που παρέχει ο debugger είναι να επιτρέπει στον προγραμματιστή τον καθορισμό κάποιων σημείων διακοπής του κώδικα (breakpoint). Τα σημεία αυτά διακόπτουν τον κώδικα καθώς τρέχει ακριβώς στην εντολή στην οποία έχουν τοποθετηθεί. Με την βοήθεια των breakpoint και των παραθύρων watches ο προγραμματιστής μπορεί να παρακολουθεί τις τιμές που παίρνουν οι διάφορες μεταβλητές κατά την εκτέλεση της εφαρμογής. Ο debugger υποστηρίζει την δυνατότητα του «Επεξεργάζομαι και Συνεχίζω», δηλαδή επιτρέπει την επεξεργασία του κώδικα καθώς αυτός περνά από το πρόγραμμα εντοπισμού σφαλμάτων. Κατά την διάρκεια της αποσφαλμάτωσης αν ο προγραμματιστής περάσει με το ποντίκι πάνω από μια μεταβλητή αυτόματα η τρέχουσα τιμή της μεταβλητής αυτής εμφανίζεται μέσα σε ένα πλαίσιο και αν ο προγραμματιστής το επιθυμεί έχει την δυνατότητα να την αλλάξει.

1.2 .NET Framework

Στην αρχή της αναφοράς μας για το Visual Studio είχε γραφτεί ότι οι εφαρμογές που υλοποιούνται με την βοήθεια του Visual Studio μπορούν να λειτουργήσουν σε πλατφόρμες που μεταξύ άλλων διαθέτουν και .NET Framework. Στο κεφάλαιο αυτό θα γίνει λόγος για το τι ακριβώς είναι το .NET Framework και για τον τρόπο με τον οποίο λειτουργεί.

Το .NET Framework έχει δημιουργηθεί και αυτό (όπως και το Visual Studio) από την Microsoft και είναι ένα πλαίσιο λογισμικού το οποίο μπορεί να εγκατασταθεί σε υπολογιστές που διαθέτουν λειτουργικό σύστημα της Microsoft. Μέσα στο .NET Framework περιλαμβάνεται μια μεγάλη βιβλιοθήκη με κωδικοποιημένες λύσεις σε κοινά προβλήματα προγραμματισμού και μια εικονική μηχανή η οποία διαχειρίζεται την εκτέλεση των προγραμμάτων που γράφτηκαν ειδικά για το πλαίσιο αυτό. Ένα από τα δυνατά σημεία του .NET Framework είναι ότι υποστηρίζει πολλές γλώσσες προγραμματισμού με τέτοιο τρόπο που να επιτρέπει στην γλώσσα της διαλειτουργικότητας, κατά την οποία κάθε γλώσσα

μπορεί να χρησιμοποιήσει κώδικα που είναι γραμμένος σε άλλες γλώσσες. Συγκεκριμένα οι γλώσσες οι οποίες υποστηρίζονται από την βιβλιοθήκη του .NET είναι οι γλώσσες που περιέχονται στο ίδιο το .NET.

Η κλάση της βασικής βιβλιοθήκης του .NET Framework παρέχει ένα μεγάλο φάσμα από χαρακτηριστικά τα οποία μεταξύ άλλων συμπεριλαμβάνουν διεπαφές χρήστη, πρόσβαση σε δεδομένα, σύνδεση με βάση δεδομένων, κρυπτογραφία, ανάπτυξη ιστοσελίδων, αριθμητικούς αλγορίθμους και δίκτυα επικοινωνίας. Η κλάση της βιβλιοθήκης χρησιμοποιείται από τους προγραμματιστές, οι οποίοι συνδυάζοντας αυτά που παρέχει η βιβλιοθήκη με τον δικό τους κώδικα δημιουργούν διάφορες εφαρμογές.

Τα προγράμματα που έχουν γραφτεί για το .NET Framework μπορούν να εκτελεστούν σε περιβάλλοντα λογισμικού τα οποία μπορούν να διαχειριστούν τις απαιτήσεις για τον χρόνο εκτέλεσης του προγράμματος. Για τον λόγο αυτό το περιβάλλον χρόνου εκτέλεσης αποτελεί μέρος του .NET Framework και είναι γνωστό σαν Common Language Runtime (CLR). Το CLR παρέχει την εμφάνιση μιας εικονικής μηχανής εφαρμογών με τέτοιο τρόπο ώστε οι προγραμματιστές να μην χρειάζεται να γνωρίζουν τις δυνατότητες του επεξεργαστή που θα εκτελεστεί το πρόγραμμα. Το CLR παρέχει επίσης και άλλες σημαντικές υπηρεσίες, όπως η ασφάλεια, η διαχείριση μνήμης και ο χειρισμός εξαιρέσεων. Με λίγα λόγια η βιβλιοθήκη και το CRL μαζί αποτελούν το .NET Framework.

Το .NET Framework έχει δημιουργηθεί από το Microsoft όπως έχει ήδη αναφερθεί και σκοπός του είναι να χρησιμοποιείται από τις περισσότερες νέες εφαρμογές που δημιουργούνται για την πλατφόρμα των Windows. Στη συνέχεια θα δούμε περισσότερες λεπτομέρειες για το .NET Framework.

1.2.1 Κύρια Χαρακτηριστικά Σχεδιασμού

Στο υποκεφάλαιο αυτό θα γίνει μια σύντομη περιγραφή των κύριων χαρακτηριστικών του σχεδιασμού του .NET Framework.

Διαλειτουργικότητα

Τα συστήματα πληροφορικής απαιτούν συνήθως να υπάρχει αλληλεπίδραση μεταξύ των νέων και των παλαιών εφαρμογών. Για τον λόγο αυτό το .NET Framework παρέχει τα μέσα ώστε να υπάρχει πρόσβαση στη λειτουργικότητα των προγραμμάτων που εκτελούνται έξω από το περιβάλλον του. Η πρόσβαση στα στοιχεία του COM παρέχεται στους χώρους ονομάτων System.Runtime.InteropServices και System.EnterpriseServices του framework.

Κοινή Μηχανή Χρόνου Εκτέλεσης (Common Runtime Engine)

Η Common Language Runtime (CLR) είναι η εικονική μηχανή, στοιχείο του .NET Framework. Όλα τα προγράμματα που χρησιμοποιούν το .NET Framework εκτελούνται κάτω από την εποπτεία της CLR, η οποία εγγυάται ορισμένες ιδιότητες και συμπεριφορές στον τομέα της διαχείρισης μνήμης, της ασφάλειας και στον χειρισμό των εξαιρέσεων.

Ανεξαρτησία της γλώσσας

Το .NET Framework εισάγει ένα κοινό τύπο συστήματος (Common Type System – CTS), ο οποίος σύμφωνα με τις προδιαγραφές του καθορίζει όλους τους πιθανούς τύπους δεδομένων και τις προγραμματιστικές δομές που υποστηρίζονται από το CLR καθώς και το πώς αυτά μπορούν ή όχι να αλληλεπιδράσουν μεταξύ τους σύμφωνα πάντα με τις προδιαγραφές της Κοινής Γλώσσας Υποδομών (Common Language Infrastructure – CLI). Εξαιτίας της δυνατότητας αυτής το .NET Framework επιτρέπει την ανταλλαγή των τύπων και των στιγμιότυπων των αντικειμένων μεταξύ των βιβλιοθηκών και των εφαρμογών που χρησιμοποιούν οποιονδήποτε γραπτό σύμφωνο με γλώσσα .NET.

Βασική Κλάση Βιβλιοθήκης

Η βασική κλάση βιβλιοθήκης (Base Class Library – BCL) είναι ένα μέρος από ολόκληρη την κλάση βιβλιοθήκης του framework (Framework Class Library – FCL). Μέσα στην βιβλιοθήκη αυτή παρέχονται διάφορες λειτουργίες. Οι λειτουργίες αυτές είναι διαθέσιμες σε όλες τις γλώσσες που χρησιμοποιούνται από το .NET Framework. Η βασική αυτή βιβλιοθήκη προσφέρει διάφορες κλάσεις οι οποίες ενσωματώνουν μια σειρά από κοινές λειτουργίες, συμπεριλαμβανομένου των αρχείων γραφής και ανάγνωσης, την απόδοση των γραφικών, τον χειρισμό των XML εγγράφων, την αλληλεπίδραση των βάσεων δεδομένων κτλ.

Απλοποιημένη Εγκατάσταση

Το .NET Framework περιλαμβάνει διάφορα χαρακτηριστικά σχεδίασης και εργαλεία τα οποία βοηθούν στη διαχείριση της εγκατάστασης του λογισμικού ώστε να εξασφαλιστεί ότι δεν έρχεται σε αντίθεση με το προηγούμενο εγκατεστημένο λογισμικό και ότι ανταποκρίνεται στις απαιτήσεις ασφαλείας.

Ασφάλεια

Ο σχεδιασμός του .NET Framework έχει ως στόχο του να αντιμετωπίσει ορισμένα από τα τρωτά σημεία, όπως για παράδειγμα οι υπερχειλίσεις μνήμης, τα οποία είχαν γίνει αντικείμενο εκμετάλλευσης από κακόβουλο λογισμικό. Τέλος το .NET Framework παρέχει ένα κοινό μοντέλο ασφαλείας για όλες τις εφαρμογές του.

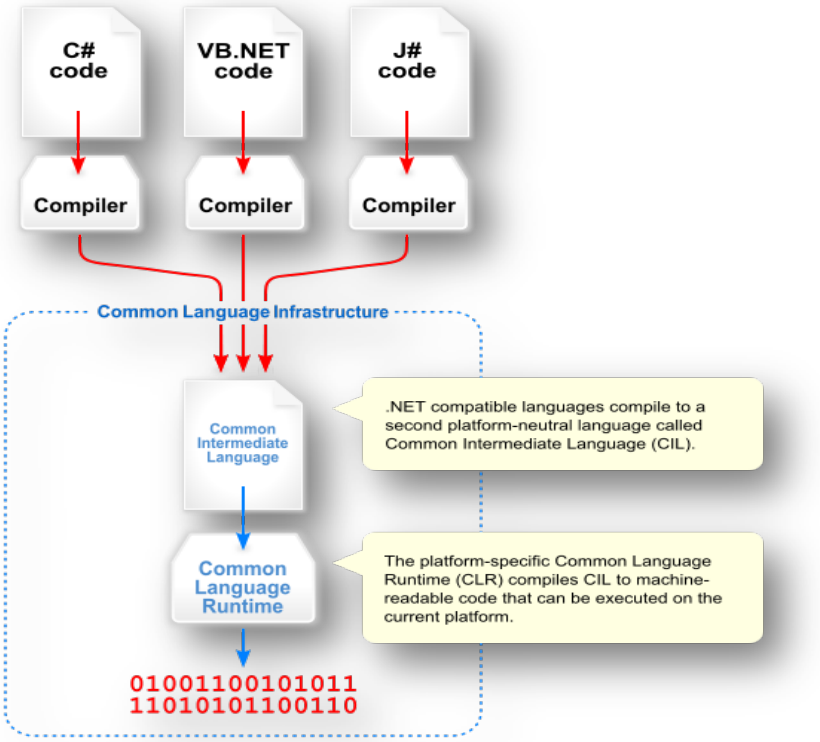
1.2.2 Αρχιτεκτονική του .NET Framework

Στο προηγούμενο υποκεφάλαιο είχε γίνει αναφορά για τα κύρια χαρακτηριστικά του σχεδιασμού του .NET Framework. Σε αυτό το υποκεφάλαιο θα αναφερθούν τα κύρια χαρακτηριστικά της αρχιτεκτονικής του.

Common Language Infrastructure (CLI)

Είχε αναφερθεί στο προηγούμενο κεφάλαιο ότι το Visual Studio επιτρέπει την σύνταξή κώδικα σε περισσότερες από μια γλώσσα προγραμματισμού. Για να το πετύχει αυτό χρησιμοποιεί στοιχεία από την αρχιτεκτονική του .NET Framework. Συγκεκριμένα το χαρακτηριστικό αυτό που χρησιμοποιεί είναι το Common Language Infrastructure (CLI).

Αυτό που κάνει ουσιαστικά το CLI είναι να προσφέρει μια πλατφόρμα ουδέτερης γλώσσας. Στην πλατφόρμα αυτή θα μπορούν να δημιουργούνται και να εκτελούνται εφαρμογές σε συνδυασμό με διάφορες λειτουργίες. Τέτοιες λειτουργίες αφορούν τον χειρισμό εξαιρέσεων, την συλλογή απορριμμάτων , την ασφάλεια και την διαλειτουργικότητα. Όλες αυτές οι δυνατότητες παρέχονται με την υλοποίηση των βασικών πτυχών του .NET Framework στο πεδίο εφαρμογής της Common Language Runtime. Με αυτό τον τρόπο οι λειτουργίες δεν θα είναι διαθέσιμες μόνο για μια γλώσσα προγραμματισμού αλλά σε όλες τις γλώσσες που υποστηρίζονται από το Framework.



Assemblies

Τα Assemblies του .NET χρησιμοποιούνται για να αποθηκεύουν τον κώδικα του CLI. Σύμφωνα με τις προδιαγραφές τους τα διάφορα Assemblies είναι αποθηκευμένα σε φορητή εκτελέσιμη μορφή (Portable Executable – PE) η οποία είναι κοινή για όλα τα αρχεία DLL και EXE που υπάρχουν στην πλατφόρμα των Windows. Ένα Assembly αποτελείται από ένα ή περισσότερα αρχεία εκ των οποίων το ένα από αυτά πρέπει να περιέχει το δηλωτικό, δηλαδή τα metadata για το Assembly. Ένα πλήρες όνομα ενός Assembly περιέχει το όνομα του κειμένου του, τον αριθμό έκδοσης, τον πολιτισμό και το ενδεικτικό δημόσιο κλειδί. Το ενδεικτικό δημόσιο κλειδί είναι μοναδικό και παράγεται όταν το Assembly γίνει compiled, αυτό έχει σαν αποτέλεσμα δύο Assembly με το ίδιο ενδεικτικό δημόσιο κλειδί, να είναι εγγυημένα ότι είναι πανομοιότυπα από την άποψη του framework. Ένα κλειδί μπορεί να είναι ιδιωτικό και να το γνωρίζει μόνο ο δημιουργός του assembly. Ένα ιδιωτικό κλειδί μπορεί να χρησιμοποιηθεί για μια ισχυρή ονομασία και σαν εγγύηση ότι το assembly είναι από τον ίδιο συγγραφέα όταν μια νέα έκδοση δημιουργηθεί.

Metadata

Μέσα από τα metadata αυτό-περιγράφεται ολόκληρο το CLI. Τα metadata χρησιμοποιούνται ώστε να τα ελέγχει η CLR και να διασφαλίζει ότι έχει καλεστεί η σωστή μέθοδος. Συνήθως τα metadata δημιουργούνται από τους compilers των γλωσσών. Παρόλα αυτά metadata με προσαρμοσμένα χαρακτηριστικά μπορούν να δημιουργηθούν και από τους προγραμματιστές. Αυτό που περιέχεται μέσα στα metadata είναι διάφορες πληροφορίες σχετικά με τα assemblies.

Ασφάλεια

Το .NET διαθέτει δικό του μηχανισμό ασφαλείας, ο οποίος έχει δύο γενικά χαρακτηριστικά. Τον κωδικό ασφαλείας για πρόσβαση (Code Access Security – CAS) και την επικύρωση και επαλήθευση. Ο κωδικός ασφαλείας CAS βασίζεται σε αποδεικτικά στοιχεία τα οποία συνδέονται με ένα συγκεκριμένο assembly. Συνήθως τα αποδεικτικά αυτά στοιχεία είναι η πηγή του assembly. Με τον όρο πηγή του assembly εννοείται από πού προέρχεται το assembly. Αν είναι εγκατεστημένο στον τοπικό υπολογιστή ή αν έχει κατέβει μέσω Internet. Μέσα από τα στοιχεία που παίρνει από το assembly ο Code Access Security καθορίζει τα δικαιώματα που χορηγούνται στον κάθε κωδικό. Για παράδειγμα μπορεί να ζητείται άλλος κωδικός αν για την κλήση ενός κώδικα απαιτείται μια συγκεκριμένη άδεια. Η ζήτηση της άδειας αυτής προκαλεί την CRL να εκτελέσει μια στοίβα κλήσεων. Κάθε assembly μιας μεθόδου στην στοίβα των κλήσεων ελέγχεται για τα απαιτούμενα δικαιώματα που χρειάζεται. Αν για κάποιο assembly δεν παρέχονται όλα τα δικαιώματα τότε «ρίχνεται» μια εξαίρεση ασφαλείας.

Όταν φορτωθεί ένα assembly η CRL εκτελεί διάφορες δοκιμές. Από αυτές τις δοκιμές, δύο αφορούν πιστοποίηση και επαλήθευση. Η CRL κατά την διάρκεια των δοκιμών που αφορούν την πιστοποίηση ελέγχει αν το assembly περιέχει έγκυρα metadata και κατά πόσο οι εσωτερικοί πίνακες είναι σωστοί. Κατά την επαλήθευση οι δοκιμές δεν είναι συγκεκριμένες όπως με την πιστοποίηση. Ειδικοί μηχανισμοί επαλήθευσης ελέγχουν αν ο κώδικας κάνει κάτι επικίνδυνο. Ένας κώδικας που δεν είναι ασφαλής είναι δυνατόν να εκτελεστεί μόνο στην

περίπτωση που το assembly έχει την άδεια «αγνόηση του ελέγχου» πράγμα που σημαίνει ότι ο κώδικας είναι εγκατεστημένος στον τοπικό υπολογιστή.

Class Library

Το .NET Framework περιλαμβάνει ένα σύνολο από κλάσεις βιβλιοθηκών. Μια κλάση βιβλιοθήκης είναι οργανωμένη σε μια ιεραρχία ονομάτων. Οι περισσότερες από τις βιβλιοθήκες που βρίσκονται μέσα στο API του .NET Framework είναι μέρος του χώρου ονομάτων (namespace) System.* ή Microsoft.*. Οι βιβλιοθήκες αυτές εμφανίζουν ένα μεγάλο αριθμό κοινών λειτουργιών. Μερικές από τις λειτουργίες αυτές είναι η ανάγνωση και η γραφή αρχείων, η γραφική απόδοση, η αλληλεπίδραση των βάσεων δεδομένων και ο χειρισμός ενός αρχείου XML. Οι βιβλιοθήκες που περιλαμβάνει το .NET είναι διαθέσιμες για όλες τις συμβατές γλώσσες προγραμματισμού (CLI). Η κύρια κλάση βιβλιοθήκης του .NET Framework αποτελείται από δύο μέρη. Την βασική κλάση βιβλιοθήκης και την κλάση βιβλιοθήκης του Framework.

Η βασική κλάση βιβλιοθήκης του .NET Framework (Base Class Library – BCL) περιλαμβάνει ένα μικρό υποσύνολο από ολόκληρη την κλάση βιβλιοθήκης. Η βιβλιοθήκη BCL αποτελεί το βασικό σύνολο των κλάσεων που χρησιμεύουν ως το βασικό API του Common Language Runtime.

Η κλάση βιβλιοθήκης Framework (FCL) αποτελεί ένα υπερσύνολο της βασικής κλάσης βιβλιοθήκης (BCL). Αυτό που περιλαμβάνεται μέσα στην FCL είναι ένα εκτεταμένο σύνολο από βιβλιοθήκες. Μερικές από τις βιβλιοθήκες αυτές είναι η Windows Forms, η ADO.NET και η ASP.NET. Η κλάση βιβλιοθήκης Framework είναι αρκετά μεγάλη σε έκταση, πολύ μεγαλύτερη από κάποιες γλώσσες προγραμματισμού όπως είναι η C++, ενώ μπορεί να συγκριθεί με το πλαίσιο των βασικών βιβλιοθηκών της γλώσσα προγραμματισμού Java.

Διαχείριση Μνήμης

Η Common Language Runtime που υπάρχει στο .NET Framework επιτρέπει στον προγραμματιστή μιας εφαρμογής να μην ασχολείται με την διαχείριση της μνήμης, αφού το θέμα της μνήμης έχει «αναλάβει» να το κάνει μόνη της. Για να μπορέσει να γίνει αυτό «βοηθάει» και η διαχείριση σωρού, η οποία μετατρέπει την μνήμη που διατίθεται στα διάφορα στιγμιότυπα των τύπων του .NET, σε ένα απόθεμα μνήμης το οποίο διαχειρίζεται η CLR. Όσο υπάρχει αναφορά σε ένα αντικείμενο, είτε αυτή η αναφορά είναι άμεση είτε όχι, το αντικείμενο αυτό θεωρείται ότι είναι σε χρήση από την CRL. Όταν ένα αντικείμενο δεν μπορεί πλέον να χρησιμοποιηθεί ή δεν υπάρχει πλέον αναφορά για αυτό, τότε το αντικείμενο μετατρέπεται σε σκουπίδια. Η μετατροπή όμως του αντικειμένου σε σκουπίδια δεν απελευθερώνει την μνήμη που χρησιμοποιούσε, αλλά ακόμα εξακολουθεί να κατέχει την μνήμη που του αναλογεί. Για να λυθεί το πρόβλημα αυτό το .NET Framework παρέχει ένα συλλέκτη σκουπιδιών (Garbage Collector). Ο συλλέκτης αυτός τρέχει περιοδικά σε ένα ξεχωριστό νήμα από αυτό της εφαρμογής και αυτό που κάνει είναι να απαριθμεί τα αντικείμενα που είναι πλέον άχρηστα για την εφαρμογή και να διεκδικεί την μνήμη που τους αναλογεί.

Ο Garbage Collector που υπάρχει στο .NET δεν τρέχει οποιανδήποτε στιγμή αλλά, όπως αναφέρθηκε και στην προηγούμενη παράγραφο, τρέχει όταν έχει χρησιμοποιηθεί ένα συγκεκριμένο ποσοστό μνήμης ή όταν υπάρχει αρκετή πίεση για την μνήμη που χρησιμοποιεί το σύστημα. Όταν οι συνθήκες για να διεκδικήσει τη μνήμη αυξάνονται τότε, τα τρεξίματα του GC δεν είναι προκαθορισμένα ή προγραμματισμένα. Κάθε εφαρμογή που δημιουργείται χρησιμοποιώντας το .NET διαθέτει ένα σύνολο από ρίζες, οι οποίες είναι δείκτες προς τα αντικείμενα που χρησιμοποιεί. Οι δείκτες αυτοί χρησιμοποιούνται στην διαχείριση του σωρού (εκεί όπου διαχειρίζονται τα αντικείμενα). Συγκεκριμένα οι δείκτες που υπάρχουν σε μια εφαρμογή περιλαμβάνουν αναφορές σε στατικά αντικείμενα και αντικείμενα που ορίζονται ως τοπικές μεταβλητές ή παράμετροι σε μεθόδους ή ακόμα και αντικείμενα που αναφέρονται από τα μητρώα της CPU. Ο GC χρησιμοποιεί τις ρίζες αυτές και τους δείκτες όταν τρέχει, ώστε να βρει όλα τα αντικείμενα της εφαρμογής. Αυτό που κάνει συγκεκριμένα είναι να προκαλεί

παύσεις στην εφαρμογή και για κάθε αντικείμενο που αναφέρεται στο σύνολο από ρίζες που υπάρχουν στην εφαρμογή, να απαριθμεί αναδρομικά όλα τα αντικείμενα που είναι προσβάσιμα από αυτό και να το σηματοδοτεί ως προσβάσιμο. Για να ανακαλύψει ο GC τα αντικείμενα που είναι έγκλειστα από κάποιο άλλο χρησιμοποιεί metadata του .NET και αντανάκλαση. Η διαδικασία αυτή γίνεται αναδρομικά σε όλα τα αντικείμενα. Στη συνέχεια, χρησιμοποιώντας αντανάκλαση, απαριθμεί όλα τα αντικείμενα που βρίσκονται στον σωρό. Τα αντικείμενα που έχουν περάσει από την διαδικασία αυτή και δεν έχουν οριστεί ως προσβάσιμα τότε θεωρούνται σκουπίδια. Τα σκουπίδια που υπάρχουν στην μνήμη δεν υπολογίζονται αλλά θεωρούνται σαν ελεύθερος χώρος. Το γεγονός αυτό όμως αφήνει ελεύθερο χώρο ανάμεσα στα αντικείμενα που είναι προσβάσιμα με αποτέλεσμα πλέον να μην είναι συνεχόμενα στη μνήμη. Αυτό όμως «διορθώνεται» στη συνέχεια αφού τα αντικείμενα αυτά συμπιέζονται μαζί ώστε να μπορούν να κάνουν χρήση της μνήμης συνεχόμενα. Τέλος αφού η συλλογή σκουπιδιών τελειώσει, συνεχίζεται η εφαρμογή.

1.3 Visual Basic

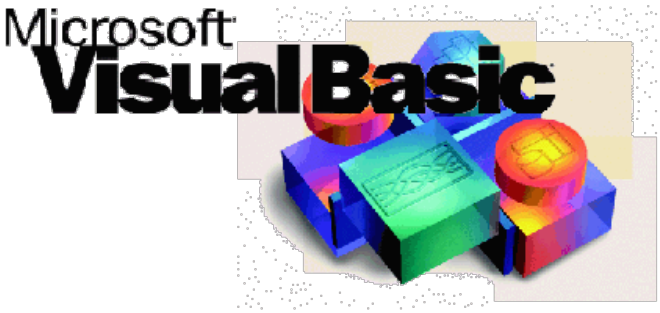
Η υλοποίηση της εφαρμογής έγινε με την βοήθεια του Visual Studio. Το Visual Studio όμως προσφέρει τους «μηχανισμούς» για την υλοποίηση μιας εφαρμογής. Μέσω του Visual Studio όπως αναφέρθηκε πιο αναλυτικά στο παραπάνω υποκεφάλαιο, μπορεί ένας προγραμματιστής να συντάξει κώδικά σε περισσότερες από μια γλώσσες προγραμματισμού. Μια από τις γλώσσες προγραμματισμού αυτές είναι και η Visual Basic η οποία επιλέχτηκε για την σύνταξη του κώδικα της εφαρμογής.

Την Visual Basic σαν γλώσσα προγραμματισμού μπορεί ένας προγραμματιστής να τη χρησιμοποιήσει είτε μέσω του Visual Studio είτε μέσω του δικού της περιβάλλοντος ανάπτυξης. Η Visual Basic η οποία υποστηρίζεται μέσω του Visual Studio αναφέρεται σαν Visual Basic.NET και αυτό είναι που την κάνει να ξεχωρίζει από την «κλασσική» Visual Basic. Αρχικά θα αναφερθούν μερικά

στοιχεία για την «κλασσική» Visual Basic και στην συνέχεια μερικά στοιχεία για την Visual Basic.NET.

Η Visual Basic είναι γλώσσα προγραμματισμού τρίτης γενιάς και ανήκει στην κατηγορία των γλωσσών που κινούνται από συμβάντα (event driven). Η Visual Basic παρέχεται από την Microsoft και έχει ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για το μοντέλο προγραμματισμού COM. Σαν γλώσσα προγραμματισμού, η Visual Basic θεωρείται εύκολη γλώσσα προγραμματισμού, τόσο στην εκμάθηση, όσο και στην χρησιμοποίηση της. Η θεώρηση αυτή προέρχεται από το γεγονός ότι διαθέτει ένα γραφικό περιβάλλον δημιουργίας καθώς επίσης και από το γεγονός ότι διαθέτει αρκετά χαρακτηριστικά από την συγγενική της γλώσσα προγραμματισμού Basic.

Στην ουσία η Visual Basic είναι απόγονος της γλώσσας προγραμματισμού Basic. Το χαρακτηριστικό «Visual» προέρχεται από το γεγονός ότι η Visual Basic διαθέτει ένα γραφικό περιβάλλον ανάπτυξης. Επιπλέον η Visual Basic επιτρέπει την ταχεία ανάπτυξη γραφικών εφαρμογών (Rapid Application Development – RAD), την σύνδεση των εφαρμογών με βάση δεδομένων μέσω κάποιον αντικειμένων, την δημιουργία στοιχείων ελέγχου ActiveX και διάφορων άλλων αντικειμένων. Τα στοιχεία που παρέχει η Visual Basic είναι αρκετά ώστε ένας προγραμματιστής να μπορέσει να ολοκληρώσει μια εφαρμογή κάνοντας χρήση μόνο των στοιχείων αυτών. Φυσικά υπάρχει και η δυνατότητα με την δήλωση εξωτερικών συναρτήσεων η Visual Basic να χρησιμοποιήσει και το Windows API.



1.3.1 Χαρακτηριστικά της Γλώσσας

Σαν απόγονος της Basic, ο σχεδιασμός της Visual Basic έχει σαν στόχο να προσφέρει στους προγραμματιστές μια εύκολη στην εκμάθηση και στον χειρισμό, γλώσσα προγραμματισμού. Επιπλέον σαν γλώσσα αφήνει τους προγραμματιστές να δημιουργήσουν πολύπλοκες εφαρμογές, ξεφεύγοντας από τις απλές εφαρμογές. Ένας προγραμματιστής που γράφει σε Visual Basic αυτό που έχει να κάνει είναι αρχικά να συνδυάσει οπτικά διάφορα στοιχεία ή ελέγχους πάνω σε μια φόρμα και να προσδιορίζει τα διάφορα χαρακτηριστικά και τις ενέργειες που θα κάνουν τα στοιχεία αυτά. Αν επιθυμεί ο προγραμματιστής τα στοιχεία αυτά να κάνουν και κάποιες επιπλέον λειτουργίες θα πρέπει να γράψει μερικές γραμμές κώδικα. Επειδή για τα διάφορα στοιχεία υπάρχουν κάποια προεπιλεγμένα χαρακτηριστικά και ενέργειες ο προγραμματιστής μπορεί να δημιουργήσει μια απλή εφαρμογή χωρίς να χρειαστεί να γράψει πολλές γραμμές κώδικα.

Ένα από τα πιο σημαντικά στοιχεία στη Visual Basic είναι οι φόρμες. Οι φόρμες είναι ο «χώρος» στον οποίο τοποθετούνται τα διάφορα στοιχεία που θα βλέπει και θα χρησιμοποιεί ο χρήστης όταν θα τρέξει την εφαρμογή. Η διαμόρφωση των φορμών γίνεται με την μέθοδο drag and drop. Δηλαδή τα διάφορα στοιχεία ελέγχου που θέλει ο προγραμματιστής να βάλει στην φόρμα, τα επιλέγει, τα σέρνει στο σημείο της φόρμας που θέλει και τα αφήνει. Για να γίνει αυτό η Visual Basic έχει ένα ειδικό μηχανισμό. Τα στοιχεία ελέγχου που διαθέτει η γλώσσα, όπως αναφέρθηκε και προηγουμένως διαθέτουν τα δικά τους χαρακτηριστικά και χειριστές συμβάντων που είναι συνδεδεμένα με αυτά. Κάθε στοιχείο κατά τη δημιουργία του δίνει και στα χαρακτηριστικά του κάποιες προεπιλεγμένες τιμές. Οι τιμές αυτές μπορούν να τροποποιηθούν από το προγραμματιστή κατά τη δημιουργία της εφαρμογής ή κατά τον χρόνο εκτέλεσης της, σύμφωνα με ενέργειες που έχουν γίνει από τον χρήστη. Η δυνατότητα αυτή παρέχει στον προγραμματιστή τη δυνατότητα να δημιουργεί δυναμικές εφαρμογές.

Οι προγραμματιστές χρησιμοποιούν την Visual Basic κυρίως για να δημιουργούν εφαρμογές για το λειτουργικό σύστημα των Windows και εφαρμογές για την

διασύνδεση συστημάτων βάσεων δεδομένων. Η Visual Basic όμως παρέχει και την δυνατότητα δημιουργίας εκτελέσιμων αρχείων (EXE), στοιχείων ελέγχου ActiveX και αρχείων DLL. Επιπλέον ένα συστατικό της Visual Basic μπορεί να μην έχει γραφικό περιβάλλον αλλά να παρέχει αντικείμενα ActiveX σε άλλα προγράμματα μέσω του Component Object Model (COM). Η δυνατότητα αυτή επιτρέπει επεξεργασία στην πλευρά του διακομιστή (server side processing) ή την δημιουργία επιπλέον μορφωμάτων (add-in module).

Η διαχείριση της μνήμης παύει να ενδιαφέρει τους προγραμματιστές που χρησιμοποιούν την Visual Basic αφού η γλώσσα διαθέτει αυτόματη διαχείριση μνήμης μέσω της τεχνικής της συλλογής σκουπιδιών (garbage collector), η οποία χρησιμοποιεί υπολογισμό αναφορών για να «ανακαλύψει» τα αντικείμενα που δεν χρησιμοποιούνται από την εφαρμογή. Η Visual Basic επιπλέον παρέχει μια μεγάλη βιβλιοθήκη με βοηθητικά αντικείμενα καθώς και βασική αντικειμενοστραφή υποστήριξη.

Σε μια γρήγορη ματιά αυτά ήταν τα κυριότερα χαρακτηριστικά της γλώσσας προγραμματισμού Visual Basic. Η τελευταία έκδοση (έκδοση 6) της γλώσσας βγήκε το 1998. Η υποστήριξη της Microsoft σταμάτησε δέκα χρόνια μετά, το 2008 και σαν διάδοχος της γλώσσας ορίστηκε η Visual Basic.NET. Παρόλα αυτά όμως υπάρχουν πολλοί προγραμματιστές ακόμα που την χρησιμοποιούν.

1.3.2 Visual Basic.NET

Η Visual Basic.NET που διαδέχτηκε, όπως προαναφέρθηκε τη Visual Basic, δεν μπορεί να θεωρηθεί από μόνη της μια ανεξάρτητη γλώσσα προγραμματισμού. Η Visual Basic.NET δεν παρέχει ένα ολοκληρωμένο περιβάλλον ανάπτυξης αλλά είναι πλέον ένα κομμάτι του Visual Studio.

Για να μπορέσει ένας προγραμματιστής να αναπτύξει μια εφαρμογή με την χρήση της Visual Basic πρέπει υποχρεωτικά να χρησιμοποιήσει το Visual Studio, το περιβάλλον του και τα στοιχεία του. Η Visual Basic .NET φεύγει από την κατηγορία των event driven γλωσσών προγραμματισμού, στην οποία ανήκε η

κλασσική Visual Basic, και περνά στην κατηγορία των αντικειμενοστραφών γλωσσών προγραμματισμού.

Μπορεί να τεθεί σαν θέμα συζήτησης αν η Visual Basic.NET αποτελεί μια εκδοχή της Visual Basic ή αποτελεί μια εντελώς διαφορετική γλώσσα προγραμματισμού. Το θέμα αυτό πηγάζει από το γεγονός ότι παρατηρήθηκαν πολλές αλλαγές μεταξύ της Visual Basic και της Visual Basic.NET. Παρά τις διάφορες αλλαγές που έχουν παρατηρηθεί αυτό που μένει το ίδιο μεταξύ δύο αυτών γλωσσών ή εκδόσεων, είναι η σύνταξη του κώδικα. Η Visual Basic.NET έχει κρατήσει τον τρόπο σύνταξης του κώδικα που διέθετε η Visual Basic.

Η Visual Basic.NET όπως και ολόκληρο το Visual Studio είναι στενά συνδεδεμένη με το Microsoft .NET Framework. Αυτό είναι και ο λόγος που η ονομασία της μετατράπηκε σε Visual Basic.NET. Παρόλο που διαθέτει τους κανόνες σύνταξης και μερικά από τα στοιχεία της Visual Basic, αυτό που χρησιμοποιεί η Visual Basic.NET είναι τη βιβλιοθήκη του .NET Framework και τα στοιχεία που αυτό παρέχει. Αυτό δικαιολογεί και το γεγονός ότι η Visual Basic.NET δεν μπορεί να θεωρηθεί σαν μια ανεξάρτητη γλώσσα, αφού δεν διαθέτει δική της βιβλιοθήκη.

1.4 Microsoft SQL Server

Τα εργαλεία που αναφέρθηκαν στα προηγούμενα κεφάλαιο αποτελούν τα εργαλεία τα οποία χρησιμοποιήθηκαν για να υλοποιηθεί η εφαρμογή κατά το προγραμματιστικό κομμάτι της. Η εφαρμογή όμως για να λειτουργήσει χρειάζεται και μια βάση δεδομένων. Η βάση δεδομένων της εφαρμογής σχεδιάστηκε και υλοποιήθηκε με την βοήθεια του Microsoft SQL Server.

Ο SQL Server είναι ένα από τα προϊόντα της Microsoft και αποτελεί ένα Σχεσιακό Μοντέλο Διαχείρισης Βάσεων Δεδομένων RDBMS. Ο SQL Server διαθέτει και αυτός τις δικές του γλώσσες προγραμματισμού. Οι κυριότερες από αυτές είναι η T-SQL και η ANSI SQL.

Στην κεντρική βάση δεδομένων του SQL Server ο χρήστης έχει την δυνατότητα να αποθηκεύσει διάφορους τύπους δεδομένων. Εκτός από τους βασικούς τύπους δεδομένων που υποστηρίζει επιτρέπει στον χρήστη να καθορίσει δικούς του σύνθετους τύπους. Οι σύνθετοι αυτοί τύποι όμως θα πρέπει να βασίζονται στους βασικούς τύπους αλλά έχουν την δυνατότητα να τροποποιηθούν.

Τα στοιχεία που κάποιος θέλει να αποθηκεύσει στη βάση δεδομένων με την χρήση του SQL Server αποθηκεύονται μέσα σε αρχεία με την επέκταση «.mdf». Κάποια δευτερεύοντα στοιχεία που πρέπει να κρατηθούν στη βάση δεδομένων ο SQL Server τα αποθηκεύει σε ένα αρχείο με την επέκταση «.ndf». Ενώ όλες οι πρόσφατες αλλαγές που έχουν γίνει στη βάση δεδομένων αποθηκεύονται στο λεγόμενο αρχείο καταγραφής το οποίο θα τα βρει κανείς με την κατάληξη «.ldf». Μια βάση δεδομένων κρατάει ένα χώρο ώστε να μπορέσει να αποθηκευτεί . Ο χώρος αυτός δεν είναι ποτέ ενιαίος αλλά χωρίζεται σε σελίδες αποθήκευσης κάθε μια από τις οποίες έχει μέγεθος 8 KB.

Οι σελίδες αυτές κρατούν τα δεδομένα της βάσης στην ενδιάμεση μνήμη RAM. Ο λόγος που αποθηκεύονται εκεί είναι για να μειωθεί η μεταφορά τους από και προς τον σκληρό δίσκο, κάνοντας έτσι την απόκριση του SQL Server πιο γρήγορη. Στην μνήμη RAM δεν μπορούν να είναι όλες οι σελίδες αποθήκευσης, ειδικά όταν έχει να κάνει με μια μεγάλη βάση δεδομένων. Το σύνολο των σελίδων που είναι αποθηκευμένες στη μνήμη RAM ονομάζεται λανθάνουσα μνήμη ή μνήμη cache. Ο αριθμός των σελίδων που μπορούν να αποθηκευτούν στη λανθάνουσα μνήμη εξαρτάται από το ποσοστό της μνήμης που έχει διαθέσιμο ο SQL Server, δηλαδή δεν είναι ένας σταθερός αριθμός. Για να μπορέσει ο SQL Server να διαχειριστεί σωστά τη λανθάνουσα μνήμη έχει διάφορους αλγορίθμους, οι οποίοι εξασφαλίζουν την καλύτερη απόδοση της μνήμης αυτής.

Ο κεντρικός διακομιστής του SQL Server χρησιμοποιεί τις λεγόμενες «συναλλαγές» για να εξασφαλίσει ότι οποιαδήποτε λειτουργία γίνει πάνω στη

βάση δεδομένων έχει αποτύχει ή έχει ολοκληρωθεί συνολικά. Σε καμία περίπτωση δεν αφήνει τη βάση δεδομένων σε μια ενδιάμεση κατάσταση. Οι «συναλλαγές» αυτές μπορεί να αποτελούνται από πολλές εντολές SQL οι οποίες θα κάνουν μόνο μόνιμη αλλαγή στη βάση δεδομένων, εφόσον η τελευταία εντολή της συναλλαγής, η λεγόμενη COMMIT εντολή, ολοκληρωθεί με επιτυχία.

Τυχόν αλλαγές που έγιναν σε οποιαδήποτε σελίδα, θα ενημερώσουν την προσωρινή μνήμη της σελίδας. Ταυτόχρονα όλες οι εργασίες που εκτελούνται θα καταγράφονται σε ένα αρχείο καταγραφής, μαζί με το αναγνωριστικό συναλλαγής. Κάθε καταχώρηση του αρχείου καταγραφής προσδιορίζεται από έναν αύξοντα αριθμό καταχώρησης (Log Sequence Number - LSN) ο οποίος χρησιμοποιείται για να εξασφαλίσει ότι όλες οι αλλαγές καταγράφονται στα αρχεία δεδομένων. Επίσης, κατά τη διάρκεια μιας επαναφοράς καταχώρησης ο αριθμός αυτός χρησιμοποιείται για να ελέγξει ότι δεν υπάρχουν διπλές καταχωρήσεις ή καταχωρήσεις που παραλείπονται. Θα πρέπει επίσης να εξασφαλιστεί ότι όλες οι εργασίες σε μια συναλλαγή είναι καταγεγραμμένες στο αρχείο καταγραφής πριν από κάθε COMMIT εργασία που αναφέρεται ως ολοκληρωμένη.

Ο SQL Server παρέχει την δυνατότητα σε πολλούς χρήστες ταυτόχρονα να έχουν πρόσβαση στα δεδομένα μιας βάσης. Αυτό όμως προϋποθέτει μια σειρά από ελέγχους που πρέπει να γίνουν από τον κεντρικό διακομιστή ώστε να εξασφαλιστεί η ακεραιότητα των δεδομένων. Ο κεντρικός διακομιστής του SQL Server παρέχει δυο τρόπους ελέγχου τις ταυτόχρονης πρόσβασης στα δεδομένα (ή αλλιώς του ταυτοχρονισμού). Ο ένας είναι ο αισιόδοξος ταυτοχρονισμός και ο άλλος ο απαισιόδοξος.

Κατά τον απαισιόδοξο ταυτοχρονισμό, ο διακομιστής του SQL Server, ελέγχει την ταυτόχρονη πρόσβαση στη βάση δεδομένων με την χρήση κλειδώματος (Locks). Τα κλειδώματα μπορεί να διαμοιράζονται είτε να είναι αποκλειστικά. Ένα αποκλειστικό κλείδωμα δίνει στον χρήστη την αποκλειστική πρόσβαση στα δεδομένα. Αυτό σημαίνει ότι κανένας άλλος χρήστης δεν μπορεί αν έχει πρόσβαση στα στοιχεία αυτά όσο υπάρχει αυτό το κλείδωμα. Για να μπορούν να έχουν πρόσβαση περισσότεροι από ένας χρήστες σε κάποια στοιχεία θα πρέπει να έχουν ένα κοινό κλείδωμα. Κοινό κλείδωμα όμως δίνεται μόνο όταν οι χρήστες θέλουν μόνο να διαβάσουν τα δεδομένα. Σε διαφορετική περίπτωση το κλείδωμα

είναι αποκλειστικό. Τα κλειδώματα, είτε είναι αποκλειστικά είτε κοινά μπορούν να εφαρμοστούν σε διαφορετικά επίπεδα. Σε ολόκληρους πίνακες, σελίδες αποθήκευσης ή ακόμα και σε διαφορετικές γραμμές.

Σύμφωνα με τον αισιόδοξο ταυτοχρονισμό, ο μηχανισμός του SQL Server, επιτρέπει την δημιουργία μιας νέας έκδοσης μιας γραμμής όταν η γραμμή ενημερώνεται. Οι διάφορες εκδόσεις που μπορεί να υπάρχουν για κάθε γραμμή διαθέτουν ένα αυξανόμενο ID συσχετισμένο με αυτή. Τόσο οι νέες εκδόσεις της κάθε γραμμής όσο και οι παλαιές εκδόσεις της αποθηκεύονται και διατηρούνται. Ωστόσο οι παλαιές εκδόσεις τις κάθε γραμμής δεν αποθηκεύονται στην ίδια βάση δεδομένων στην οποία βρίσκονται οι καινούργιες αλλά αποθηκεύονται σε μια βάση δεδομένων του συστήματος, την λεγόμενη «tempdb». Όταν μια γραμμή βρίσκεται σε μια φάση ενημέρωσης, οποιαδήποτε αίτηση για την γραμμή αυτή απορρίπτεται και το αίτημα εκτελείται στην προηγούμενη έκδοση της γραμμής. Οι δύο εκδόσεις που θα δημιουργηθούν για τη γραμμή αυτή θα αποθηκευτούν στην βάση δεδομένων και θα ξεχωρίζει η μια από την άλλη με τη βοήθεια του ID τους.

Ο κύριος τρόπος για να ανακτήσει δεδομένων από μια βάση είναι με την χρήση ερωτημάτων (queries). Μια ερώτηση διευκρινίζει μέσα από την σύνταξή της αυτό που θέλει να ανακτήσει από τη βάση δεδομένων. Κάθε ερώτηση που υποβάλλεται επεξεργάζεται από τον «επεξεργαστή ερώτησης», ο οποίος υπολογίζει την σειρά βημάτων που πρέπει να γίνουν ώστε να ανακτηθούν τα δεδομένα που ζητά η ερώτηση. Η ακολουθία των απαραίτητων ενεργειών που χρειάζεται να εκτελεστούν για μια ερώτηση καλείται «σχέδιο ερώτησης».

Ο SQL Server εκτός από τις ερωτήσεις που μπορούν να του υποβληθούν έχει την δυνατότητα να αποθηκεύει κάποιες ερωτήσεις. Οι ερωτήσεις που είναι αποθηκευμένες στον SQL Server ονομάζονται Stored Procedures. Οι αποθηκευμένες ερωτήσεις μπορούν να δεχτούν τιμές που στέλνονται από τον πελάτη ως παράμετροι εισαγωγής, και να στείλουν τα αποτελέσματα τους ως παραμέτρους παραγωγής. Μπορούν να καλέσουν Functions, ή και άλλες Stored Procedures. Οι Stored Procedures χρησιμοποιούνται γιατί είναι γρηγορότερες σε σχέση με τις απλές ακολουθίες SQL.

1.5 SQL Server Management Studio

Το SQL Server Management Studio είναι ένα εργαλείο το οποίο δημιουργήθηκε από την Microsoft για να βοηθήσει τους χρήστες του SQL Server να μπορέσουν να τον διαχειριστούν καλύτερα. Το εργαλείο αυτό περιέχει ένα script editor και διάφορα γραφικά εργαλεία τα οποία εργάζονται με τα αντικείμενα και τα χαρακτηριστικά του SQL Server.

Ένα από τα βασικά εργαλεία του SQL Server Management Studio είναι ο εξερευνητής που διαθέτει. Ο εξερευνητής αυτός επιτρέπει στον χρήστης να αναζητήσει, να επιλέξει και να δράσει πάνω σε οποιονδήποτε αντικείμενο του SQL Server.

Το SQL Server Management Studio αποτελεί το κύριο εργαλείο για την διαχείριση μιας βάσης δεδομένων η οποία βρίσκεται σε κάποιον SQL Server. Το εργαλείο αυτό αποτελεί σημαντικό βοήθημα για τους διαχειριστές μιας βάσης δεδομένων αφού οι διάφορες εργασίες που έχουν να εκτελέσουν μπορούν να εκτελεστούν με την βοήθεια ενός γραφικού περιβάλλοντος.

Για κάθε έκδοση του SQL Server η Microsoft έχει δημιουργήσει και το αντίστοιχο Server Management Studio, το οποίο και διαθέτει δωρεάν.

Κεφάλαιο 2
Βάση Δεδομένων Εφαρμογής

Σε μια εφαρμογή η σχεδίαση και η δημιουργία μιας βάσης δεδομένων είναι πολύ σημαντικό κομμάτι. Η βάση δεδομένων αποτελεί ένα από τα κύρια μέρη της εφαρμογής. Υπάρχουν φυσικά και εφαρμογές οι οποίες δεν στηρίζονται ούτε χρησιμοποιούν κάποια βάση δεδομένων. Τέτοιες εφαρμογές όμως συνήθως είναι απλές και μικρές εφαρμογές. Σε εφαρμογές όμως που χρησιμοποιούν βάση δεδομένων είναι σημαντικό η βάση δεδομένων αυτή να είναι καλά και σωστά σχεδιασμένη. Μια καλά σχεδιασμένη βάση δεδομένων μπορεί εύκολα να διαχειριστεί από την εφαρμογή και εύκολα να συντηρηθεί.

Στο κεφάλαιο αυτό θα δούμε αναλυτικά τη βάση δεδομένων που έχει δημιουργηθεί για την συγκεκριμένη εφαρμογή. Μια βάση δεδομένων αποτελείται από κάποιους πίνακες οι οποίοι ομαδοποιούν τα δεδομένα της εφαρμογής και συνδέονται μεταξύ τους ώστε να δώσουν περισσότερες πληροφορίες στον χρήστη. Η βάση δεδομένων όμως μπορεί να αποτελείται και από διάφορες συναρτήσεις, λειτουργίες και triggers. Στη συνέχεια θα παρουσιαστή ολόκληρη η βάση δεδομένων της εφαρμογής.

2.1 Πίνακες

Μια βάση δεδομένων όπως έχει αναφερθεί και προηγουμένως κατά κύριο λόγο αποτελείται από πίνακες μέσα στους οποίους θα αποθηκεύονται τα διάφορα δεδομένα και πληροφορίες που χρειάζεται η εφαρμογή για να λειτουργήσει.

2.1.1 Πίνακας Μονάδων Μέτρησης

Ο συγκεκριμένος πίνακας είναι αυτός στον οποία καταχωρούνται οι διάφορες μονάδες μέτρησης που έχει δημιουργήσει ο χρήστης της εφαρμογής. Μέσα στη βάση δεδομένων ο πίνακας αυτός έχει την ονομασία «TblMonMetr». Η μονάδα

μέτρησης είναι ένα από τα χαρακτηριστικά που διαθέτει ένα είδος/προϊόν. Ο πίνακας αυτός έχει δύο στήλες, όπως αυτές φαίνονται παρακάτω.

- Κωδικός Μονάδας Μέτρησης: μέσα στο περιβάλλον ανάπτυξης της βάσης δεδομένων ονομάζεται χαρακτηριστικά «KodMonMetr» και ο τύπος δεδομένων που δέχεται είναι χαρακτήρες. Η στήλη αυτή αποτελεί το κύριο κλειδί του πίνακα και σε αυτό καταχωρείται ο κωδικός που έχει δώσει ο χρήστης στη μονάδα μέτρησης.
- Περιγραφή Μονάδας Μέτρησης: η στήλη αυτή έχει το όνομα «DescMonMetr» και δέχεται αυτή δεδομένα με την μορφή χαρακτήρων (ένα πρώτο όριο στους χαρακτήρες που μπορεί να δεχτεί είναι 50). Στη στήλη αυτή ο χρήστης καταχωρεί την περιγραφή που έχει δώσει σε μια μονάδα μέτρησης.

2.1.2 Πίνακας Κατηγοριών

Ο πίνακας αυτός χρησιμεύει ώστε ο χρήστης να αποθηκεύει τις διάφορες κατηγορίες προϊόντων που έχει δημιουργήσει. Κάθε ένα από τα είδη/προϊόντα που ο χρήστης θα καταχωρήσει στην εφαρμογή θα ανήκει σε μια κατηγορία από αυτές που υπάρχουν στον συγκεκριμένο πίνακα. Το χαρακτηριστικό όνομα του πίνακα αυτού είναι «TblKatEidos». Οι στήλες που εμφανίζονται στον πίνακα είναι:

- Κωδικός Κατηγορίας: η στήλη αυτή είναι το κύριο κλειδί του πίνακα. Εδώ καταχωρείται ο κωδικός που έχει δώσει ο χρήστης της εφαρμογής σε κάθε κατηγορία ειδών που έχει δημιουργήσει. Δέχεται δεδομένα με τύπο χαρακτήρες και μέσα στον πίνακα έχει την ονομασία «KodKatEidos».
- Περιγραφή Κατηγορίας: η στήλη αυτή, με όνομα «DescKatEidos», περιέχει την περιγραφή της κάθε κατηγορίας που υπάρχει στον πίνακα αυτό. Τα δεδομένα που δέχεται είναι χαρακτήρες.
- Συνταγή Παραγωγής: το πεδίο/στήλη αυτή έχει σημαντικό ρόλο. Αυτό που κάνει είναι να καθορίσει αν η κατηγορία αυτή δέχεται ή αλλιώς χρειάζεται συνταγή παραγωγής. Δηλαδή καθορίζει αν τα είδη/προϊόντα που ανήκουν σε αυτή την κατηγορία είναι ύλες ή είδη που παράγονται αν συνδυαστούν κάποιες ύλες. Η στήλη αυτή έχει το όνομα «Sintagi» και δέχεται το

δεδομένα smallint. Συγκεκριμένα αν μια κατηγορία χρειάζεται συνταγή παραγωγής τότε το πεδίο αυτό παίρνει την τιμή «1» αλλιώς παίρνει την τιμή «0».

2.1.3 Πίνακας Ειδών

Ο πίνακας των ειδών είναι ουσιαστικά ο κύριος πίνακας της βάσης δεδομένων. Σε αυτόν τον πίνακα καταχωρούνται όλα τα είδη/προϊόντα της εφαρμογής. Ολόκληρη η λειτουργία της εφαρμογής είναι γύρω από τα είδη/προϊόντα και την διαχείριση τους. Ο πίνακας αυτός έχει το όνομα «TblEidos» και οι στήλες που περιέχει είναι οι ακόλουθες.

- Κωδικός Είδους: το πεδίο αυτό αποτελεί το κύριο κλειδί του πίνακα και μέσα στον πίνακα βρίσκεται με το όνομα «KodEidos». Αποτελεί τον κωδικό του είδους/προϊόντος που του έχει δώσει ο χρήστης της εφαρμογής και δέχεται δεδομένα χαρακτήρων.
- Περιγραφή Είδους: στη στήλη αυτή ο πίνακας κρατάει την περιγραφή του κάθε είδους που υπάρχει καταχωρημένο σε αυτόν. Έχει το όνομα «DescEidos» και τα δεδομένα που μπορούν να καταχωρηθούν εδώ είναι χαρακτήρες.
- Κατηγορία είδους: η κατηγορία στην οποία ανήκει το κάθε προϊόν φαίνεται στη στήλη αυτή. Τα δεδομένα που υπάρχουν στη στήλη αυτή είναι ο κωδικός της κατηγορίας του προϊόντος και αποτελεί ξένο κλειδί από τον πίνακα των Κατηγοριών. Έχει το όνομα «KodKatEidos».
- Μονάδα Μέτρησης: στη στήλη «KodMonMetr» ο πίνακας κρατάει την μονάδα μέτρησης του είδους/προϊόντος. Αυτό που κρατάει στην στήλη αυτή είναι ο κωδικός της μονάδας μέτρησης που υπάρχει στον πίνακα των μονάδων μέτρησης.
- Όριο Αναπαραγγελίας: κάθε ένα από τα είδη/προϊόντα που είναι καταχωρημένα στον πίνακα των ειδών διαθέτει και ένα όριο αναπαραγγελίας. Το όριο αυτό βοηθάει στον εντοπισμό των ελλείψεων των ειδών και κυρίως των υλών. Με έλεγχο που γίνεται από την εφαρμογή, όριο αναπαραγγελίας μπορεί να έχουν μόνο τα είδη που δεν

έχουν εντολή παραγωγής. Η στήλη αυτή δέχεται ακέραιους αριθμούς και έχει όνομα «OrioAnapar».

2.1.4 Πίνακας Αποθήκης

Στην αποθήκη της βάσης δεδομένων αποθηκεύεται η ποσότητα των ειδών/προϊόντων που είναι διαθέσιμη καθώς και η ποσότητα του κάθε είδους που είναι κλειδωμένη. Ο πίνακας αυτός ονομάζεται «TblApothiki» και περιέχει τις τρεις παρακάτω στήλες:

- Κωδικός είδους/προϊόντος: ο κωδικός είδους αποτελεί το κύριο κλειδί του πίνακα και είναι ξένο κλειδί από τον πίνακα των ειδών. Εδώ καταχωρείται ο κωδικός του είδους και έχει το όνομα «KodEidos».
- Διαθέσιμη ποσότητα: μέσα στον πίνακα έχει το όνομα «AvailPosot» και μέσα σε αυτήν είναι κρατημένες οι διαθέσιμες ποσότητες κάθε είδους. Με βάση την στήλη αυτή και σε συνδυασμό με το όριο αναπαραγγελίας του είδους εντοπίζονται οι ελλείψεις των ειδών.
- Κλειδωμένη Ποσότητα: όταν μια εντολή παραγωγής εκτυπωθεί από την εφαρμογή, οι ποσότητες των υλών που χρειάζονται για την παραγωγή των ειδών της, αφαιρούνται από την διαθέσιμη ποσότητα των υλών και κλειδώνονται έως ότου ολοκληρωθεί η εντολή παραγωγής. Μέσα στον πίνακα η στήλη έχει το όνομα «LockPosot».

2.1.5 Πίνακας Συνταγών

Οι συνταγές παραγωγής καθορίζουν τις ποσότητες των υλών που χρειάζονται για την παραγωγή μιας μονάδας προϊόντος. Στη βάση δεδομένων ο πίνακας αυτός είναι αποθηκευμένος με το όνομα «TblSintagi». Με βάση τον πίνακα αυτό ενημερώνεται ο πίνακας της αποθήκης με τις διαθέσιμες και τις κλειδωμένες ποσότητες των υλών που χρειάζεται ένα προϊόν για να παραχθεί. Στον πίνακα υπάρχουν οι στήλες:

- Κωδικός Παραγόμενου Είδους: είναι μέρος τους σύνθετου κλειδιού που έχει ο πίνακας αυτός. Αποθηκεύει τον κωδικό του παραγόμενου είδους και έχει το όνομα «KodParagEidos».
- Κωδικός Είδους: είναι το δεύτερο μέρος του σύνθετου κλειδιού. Κρατάει για κάθε παραγόμενο είδος, τους κωδικούς των υλών του και έχει το όνομα «KodEidos».
- Ποσότητα Είδους: έχει το όνομα «PosotEidos» και κρατάει την ποσότητα της κάθε ύλης που χρειάζεται ένα παραγόμενο είδος για να δημιουργηθεί.

2.1.6 Πίνακας Κατάστασης Εντολών Παραγωγής

Κάθε εντολή παραγωγής από την στιγμή που θα δημιουργηθεί μέχρι την στιγμή που θα ολοκληρωθεί βρίσκεται σε κάποια κατάσταση. Οι διάφορες καταστάσεις που μπορεί να βρεθεί μια εντολή παραγωγής αποθηκεύεται στον πίνακα αυτό. Ονομάζεται «TblKatastEntolis» και διαθέτει μόνο δύο στήλες.

- Κωδικός Κατάστασης: είναι το κύριο κλειδί του πίνακα και δέχεται δεδομένα χαρακτήρων και συμβολοσειρών. Έχει το όνομα «KodKatast» και σε αυτήν ο χρήστης μπορεί να καταχωρήσει τον κωδικό που έχει δώσει σε κάθε κατάσταση.
- Περιγραφή Κατάστασης: στη στήλη αυτή υπάρχει η περιγραφή της κατάστασης που αντιστοιχεί σε κάθε κωδικό που υπάρχει στη στήλη «κωδικός κατάστασης». Στον πίνακα βρίσκεται με το όνομα «DescKatast».

2.1.7 Πίνακας Εντολών Παραγωγής - Header

Οι εντολές παραγωγής που δημιουργεί ο χρήστης διαθέτουν κάποιες πληροφορίες. Οι πληροφορίες αυτές που αφορούν την κάθε εντολή παραγωγής αποθηκεύονται στον συγκεκριμένο πίνακα που στη βάση δεδομένων έχει την ονομασία «TblEntoliParagogisH».Οι πληροφορίες που αποθηκεύονται στον πίνακα αυτό είναι:

- Αριθμός Εντολής Παραγωγής: αποτελεί το κύριο κλειδί του πίνακα και έχει αυτόματη αρίθμηση. Κάθε φορά που ο χρήστης δημιουργεί μια εντολή παραγωγής τότε αυτόματα η εντολή παίρνει ένα αριθμό, ο οποίος δεν είναι τυχαίος αλλά δίνεται αυξανόμενα. Στον πίνακα η στήλη αυτή έχει το όνομα «ArEntolisParag».
- Αριθμός Παραγγελίας: ο αριθμός αυτός καταχωρείται από τον χρήστη και αποτελεί τον αριθμό της παραγγελίας για την οποία γίνεται η εντολή παραγωγής. Το τι τοποθετεί ο χρήστης στο πεδίο αυτό δεν είναι ελεγχόμενο, φτάνει να είναι αριθμός. Θα ήταν ελεγχόμενο αν η εφαρμογή διαχειριζόταν και τις παραγγελίες. Το όνομα της στήλης είναι «ArParaggelias».
- Πελάτης: ο πελάτης είναι αυτός που έχει δώσει την παραγγελία. Ούτε τα δεδομένα που καταχωρούνται σε αυτήν την στήλη ελέγχονται. Δηλαδή δεν ελέγχει αν ο πελάτης είναι πελάτης της εταιρείας γιατί δεν υπάρχει κάποιος πίνακας πελατών. Στη στήλη αυτή μπορούν να καταχωρηθούν δεδομένα με χαρακτήρες μεγέθους 200.
- Ημερομηνία: στη στήλη της ημερομηνίας καταχωρείται η ημερομηνία στην οποία δημιουργήθηκε η εντολή παραγωγής. Η ημερομηνία δεν δίνεται από τον χρήστη αλλά είναι από το σύστημα. Δηλαδή έχει την ημερομηνία και την ώρα του υπολογιστή από τον οποίο καταχωρήθηκε η εντολή παραγωγής.
- Κωδικός Κατάστασης: όπως αναφέρθηκε και προηγουμένως δημιουργήθηκε ένας πίνακας στον οποίο υπάρχουν οι καταστάσεις στις οποίες μπορεί να βρίσκεται μια εντολή παραγωγής. Η στήλη αυτή είναι ξένο κλειδί από τον πίνακα των καταστάσεων και εμφανίζεται στον πίνακα των εντολών παραγωγής σαν «KodKatast».
- Έχει Εκτυπωθεί: το πεδίο αυτό είναι σαν ένας δείκτης που δείχνει αν μια εντολή παραγωγής έχει εκτυπωθεί ή όχι. Η χρησιμότητα του είναι, αν μια εντολή παραγωγής έχει εκτυπωθεί να μην μπορεί να τροποποιηθεί αφού θεωρητικά έχει σταλεί στην παραγωγή. Οι τιμές που παίρνει το πεδίο αυτό είναι «1» (ένα) αν η εντολή έχει εκτυπωθεί και «0» (μηδέν) αν δεν έχει ακόμα εκτυπωθεί.

2.1.8 Πίνακας Εντολών Παραγωγής – Details

Στον πίνακα αυτό καταχωρούνται τα προϊόντα και η ποσότητα των προϊόντων μιας εντολής παραγωγής. Στη βάση δεδομένων ο πίνακας αυτό εμφανίζεται με το όνομα «TblEntoliParagogisD».

- Αριθμός Εντολής Παραγωγής: είναι μέρος του κύριου κλειδιού του πίνακα αυτού και ξένο κλειδί από τον πίνακα «Εντολές Παραγωγής – Header». Εδώ αποθηκεύεται ο αριθμός της κάθε εντολής παραγωγής που θα συσχετιστεί με τα προϊόντα που περιέχει.
- Κωδικός Είδους: το δεύτερο μέρος του κύριου κλειδιού και ξένο κλειδί από τον πίνακα των ειδών. Στη στήλη αυτή αποθηκεύεται ο κωδικός κάθε προϊόντος που αντιστοιχεί σε κάθε εντολή παραγωγής.
- Ποσότητα: στη στήλη αυτή μπορούν να καταχωρηθούν αριθμητικά δεδομένα. Τα δεδομένα που καταχωρούνται εδώ είναι ο αριθμός της ποσότητας κάθε είδους που υπάρχει σε μια εντολή παραγωγής.

2.1.9 Πίνακας Γενικών Παραμέτρων

Ο πίνακας αυτός δεν έχει σχεδόν καμία σχέση με τους υπόλοιπους πίνακες. Τα δεδομένα που αποθηκεύονται σε αυτόν τον πίνακα δεν αφορούν τα είδη και την παραγωγή τους αλλά εξυπηρετούν στην καλύτερη λειτουργία της εφαρμογής. Ο πίνακας αυτός έχει μια γραμμή και τρείς στήλες.

- Προεπιλεγμένη κατάσταση νέας εντολής παραγωγής: όπως αναφέρεται και στο όνομα της η στήλη αυτή περιέχει την προεπιλεγμένη κατάσταση που θα έχει μια εντολή παραγωγής μόλις δημιουργηθεί. Η στήλη αυτή όπως και οι επόμενες είναι κενές και παίρνουν τιμή όταν ο χρήστης θέσει προεπιλεγμένες τιμές. Φυσικά στη στήλη αυτή καταχωρείται ο κωδικός της επιλογής του χρήστη.
- Κατάσταση ολοκληρωμένης εντολής παραγωγής: όταν ο χρήστης δώσει τιμή στη στήλη αυτή, αυτόματα σημαίνει ότι κάθε εντολή παραγωγής που βρίσκεται σε αυτή την κατάσταση θεωρείται ολοκληρωμένη και δεν μπορεί

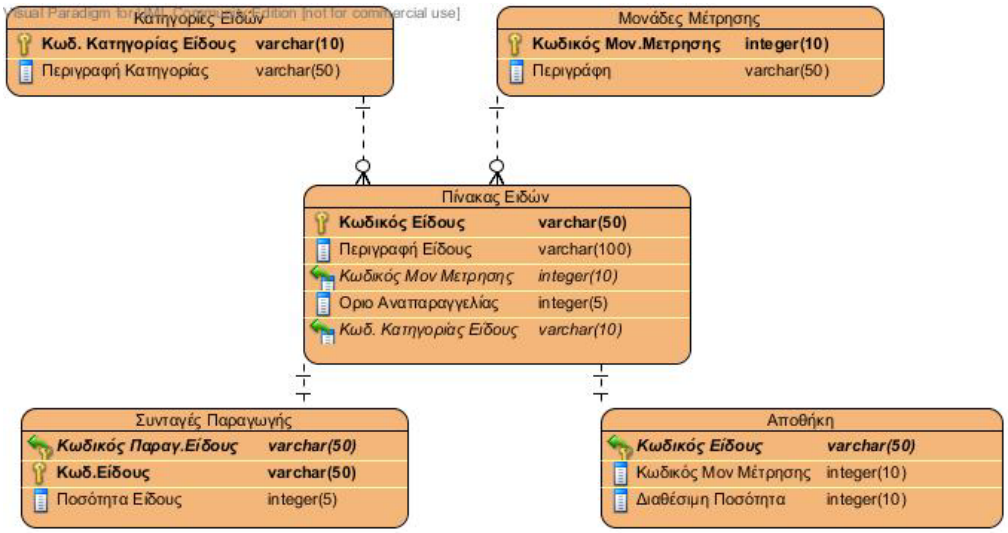
πλέον ο χρήστης να την διαχειριστεί. Όταν το πεδίο αυτό είναι κενό τότε ο χρήστης μπορεί να διαχειριστεί όλες τις εντολές παραγωγής αφού η εφαρμογή δεν μπορεί να γνωρίζει ποία κατάσταση είναι αυτή που έχουν οι ολοκληρωμένες εντολές παραγωγής.

- Κατάσταση ακυρωμένης εντολής παραγωγής: και αυτή η στήλη αντίστοιχα με την προηγούμενη ορίζει ποίες εντολές παραγωγής είναι ακυρωμένες ώστε να μην μπορούν ούτε αυτές να διαχειρίζονται από τον χρήστη. Εφόσον ο χρήστης δεν έχει δώσει τιμή σε αυτή την στήλη τότε η εφαρμογή αφήνει τον χρήστη να διαχειρίζεται όλες τις εντολές παραγωγής, μη μπορώντας να ξεχωρίσει ποίες είναι η ακυρωμένες.

2.2 Διάγραμμα ER και συσχετίσεις πινάκων

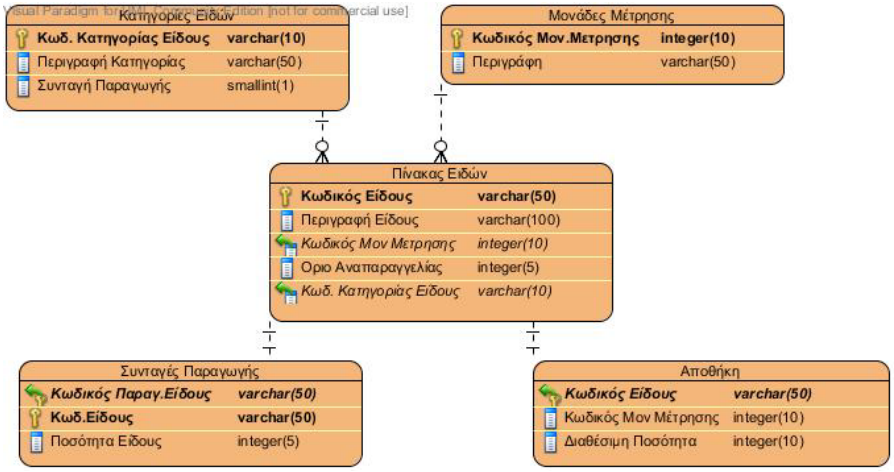
Μια βάση δεδομένων για να δημιουργηθεί πρέπει πρώτα να αναλυθεί το πρόβλημα που θα επιλύει η εφαρμογή και να σχεδιαστούν οι αρχικοί πίνακες. Στη συνέχεια θα πρέπει αυτοί οι πίνακες να μελετηθούν και να σχεδιαστούν όσο καλύτερα γίνεται ώστε το πρόβλημα να λύνεται ποίο εύκολα με αυτούς. Έχει αναφερθεί και στην αρχή του κεφαλαίου ότι είναι πολύ σημαντικό να υπάρχει μια καλά σχεδιασμένη και υλοποιημένη βάση δεδομένων.

Σε μια πρώτη επαφή με το πρόβλημα της εφαρμογής και μετά από λίγη σκέψη δημιουργήθηκε το πρώτο διάγραμμα ER της βάσης και υλοποιήθηκαν οι σχετικοί πίνακες. Το διάγραμμα αυτό είχε την μορφή που φαίνεται παρακάτω.



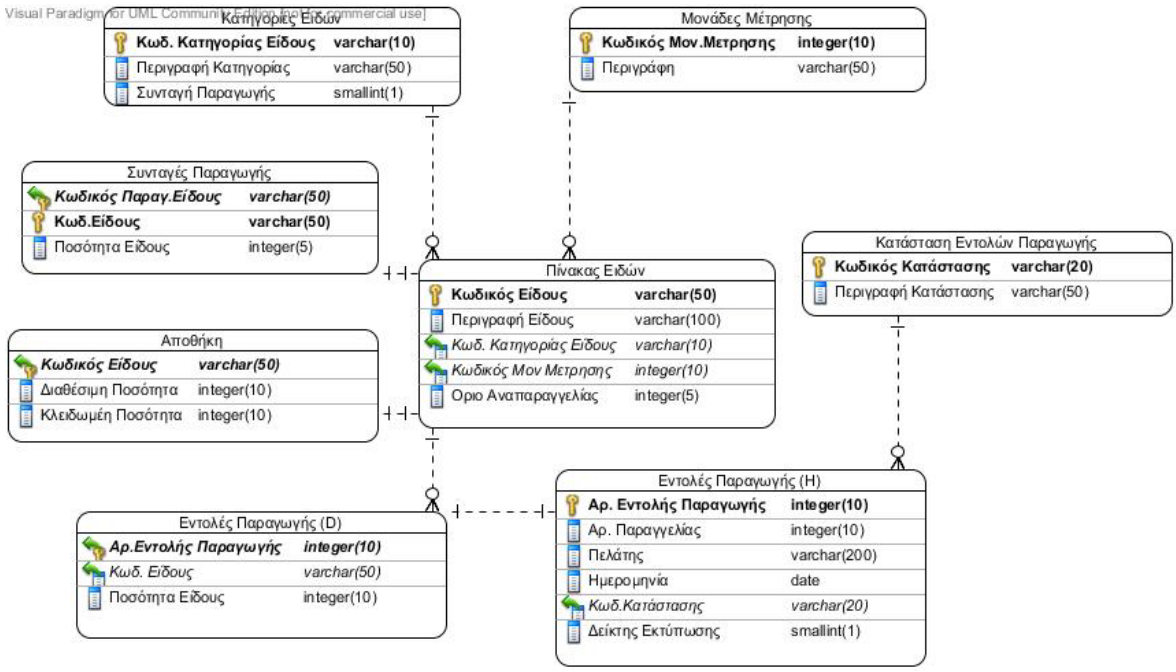
Το διάγραμμα της βάσης αυτό είχε στην αρχή μόνο τους βασικούς πίνακες που αφορούν κυρίως τα είδη και τα χαρακτηριστικά τους. Παρατηρώντας το διάγραμμα αυτό και την υλοποίηση που τελικά είχαν αυτοί οι πίνακες (όπως παρουσιάστηκαν στο προηγούμενο κεφάλαιο) βλέπουμε ότι οι διαφορές από το τελικό είναι σχεδόν ανύπαρκτες.

Καθώς άρχισε να αναπτύσσεται η εφαρμογή παρατηρήθηκε ότι θα έπρεπε να γίνει μια μικρή αλλαγή στον υπάρχοντα σχεδιασμό των πινάκων για να μπορέσει να λειτουργήσει η εφαρμογή. Έτσι στον πίνακα των κατηγοριών των ειδών προστέθηκε άλλη μια στήλη η οποία δείχνει αν η κατηγορία του είδους δέχεται συνταγή παραγωγής. Συνεπώς το διάγραμμα πλέον μετατρέπεται όπως φαίνεται ακολούθως.

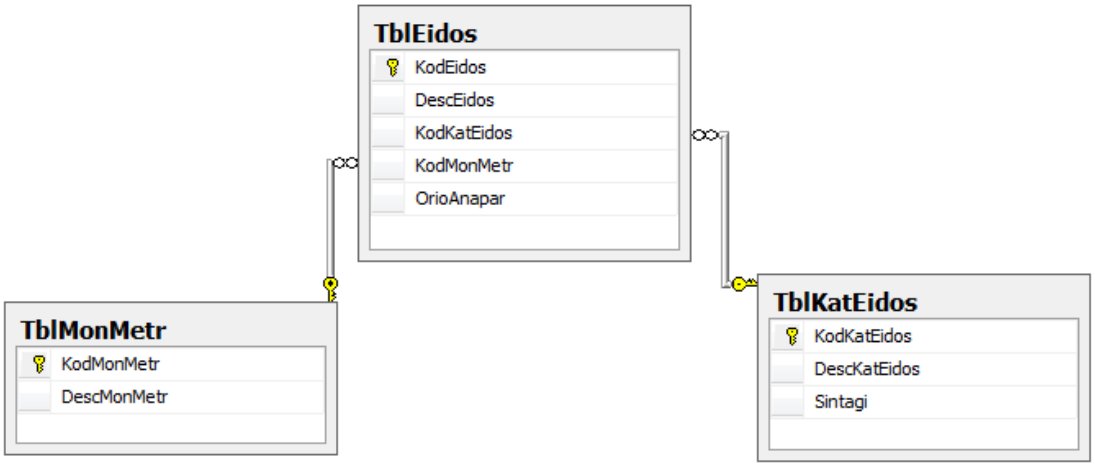


Η εφαρμογή συνέχισε να αναπτύσσεται μέχρι που έφτασε στο σημείο να πρέπει να δημιουργούνται εντολές παραγωγής. Οι εντολές αυτές θα έπρεπε κάπου να καταχωρούνται και να αποθηκεύονται ώστε να μπορεί ο χρήστης να τις διαχειριστεί. Για τον λόγο αυτό δημιουργήθηκαν οι δύο πίνακες που περιγράφηκαν στο προηγούμενο κεφάλαιο, οι οποίοι αποθηκεύουν τα στοιχεία των εντολών παραγωγής. Εκτός από το να αποθηκεύονται οι εντολές παραγωγής γεννήθηκε η ανάγκη να μπορεί ο χρήστη να γνωρίζει σε ποια κατάσταση βρίσκονται ώστε να μπορεί να τις διαχειριστεί καλύτερα. Έτσι προστέθηκε στη βάση δεδομένων και ο πίνακας με τις διάφορες καταστάσεις που μπορεί να βρεθεί μια εντολή παραγωγής.

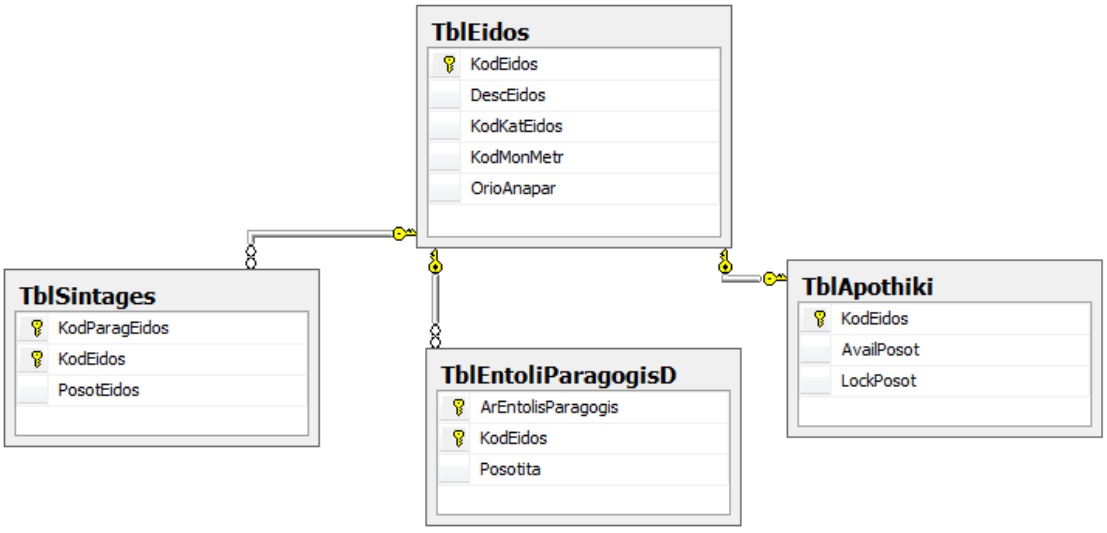
Με το τέλος της εφαρμογής η βάση δεδομένων έχει το παρακάτω τελικό σχήμα.



Το παραπάνω είναι το σχήμα όπως εμφανίζεται σχεδιασμένο. Για να μπορέσει να υλοποιηθεί το σχήμα αυτό έπρεπε, εκτός από την δημιουργία των πινάκων, να δημιουργηθούν και οι συσχετίσεις που υπάρχουν μεταξύ τους. Οι συσχετίσεις όπως δημιουργήθηκαν στον SQL server παρουσιάζονται στην συνέχεια.

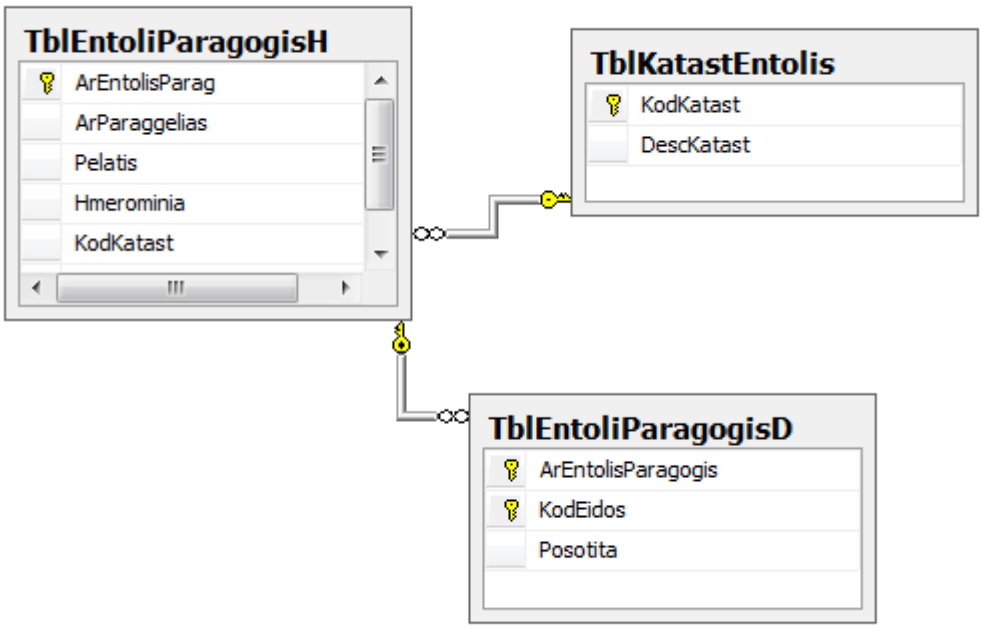


Στο πιο πάνω διάγραμμα φαίνονται οι συσχετίσεις που υπάρχουν μεταξύ του πίνακα των ειδών και των πινάκων «Κατηγορία Είδους» και «Μονάδα Μέτρησης». Οι συσχετίσεις αυτές έχουν οριστεί έτσι ώστε κάθε αλλαγή που μπορεί να γίνεται στους κωδικούς της κατηγορία ή της μονάδας μέτρησης να ενημερώνεται αυτόματα και το αντίστοιχο πεδίο στον πίνακα των ειδών.



Οι συσχετίσεις που εμφανίζονται στο παραπάνω διάγραμμα αφορούν τον πίνακα των ειδών και όλους τους πίνακες που έχουν σαν ξένο κλειδί τους τον κωδικό του

είδους. Η λογική που έχουν και αυτές οι συσχετίσεις είναι με κάθε αλλαγή στον κωδικό του είδους να ενημερώνονται και όλοι οι άλλοι πίνακες.



Αυτό είναι και το τελευταίο διάγραμμα συσχετίσεων που δημιουργήθηκε στη βάση δεδομένων. Αφορά τις εντολές παραγωγής και τον πίνακα των καταστάσεων τους. Κάθε αλλαγή που γίνεται στον κωδικό μιας κατάστασης ενημερώνεται αυτόματα και ο πίνακας των εντολών παραγωγής «Header». Αντίστοιχα το ίδιο θα συνέβαινε και αν μπορούσε να αλλαχθεί ο αριθμός της εντολής παραγωγής, θα ενημερωνόταν αυτόματα και ο πίνακας των εντολών «Details».

2.3 Προγραμματισμός στη Βάση Δεδομένων

Ο SQL Server σε συνδυασμό με το Microsoft SQL Server Management Studio δίνει τη δυνατότητα στους διαχειριστές μιας βάσης δεδομένων να γράψουν κάποια κομμάτια κώδικα μέσα στη βάση. Τα κομμάτια κώδικα αυτά χωρίζονται σε κάποιες κατηγορίες ανάλογα με το τι ακριβώς κάνουν και το πότε εκτελούνται.

Για την συγκεκριμένη βάση δεδομένων και για την συγκεκριμένη εφαρμογή αποφασίστηκε να δημιουργηθούν κάποιες Stored Procedures. Ο κύριος σκοπός που δημιουργήθηκαν αυτές οι Stored Procedures είναι για να εκτελούν κάποιους

ελέγχους που χρειάζεται η εφαρμογή σχετικά με τα δεδομένα. Οι έλεγχοι αυτοί θα μπορούσαν να γίνουν και μέσα στον κώδικα της εφαρμογής αλλά αυτό θα σήμαινε αντίστοιχα περισσότερο φόρτο της βάσης δεδομένων και της εφαρμογής και μεγάλος όγκος πληροφοριών από τη βάση δεδομένων.

Κάποιες Stored Procedures από αυτές δημιουργήθηκαν για να καλούνται από την εφαρμογή πριν γίνει κάποια εισαγωγή ή κάποια ενημέρωση σε οποιονδήποτε από τους πίνακες της βάσης δεδομένων. Πιο συγκεκριμένα αυτό που κάνουν είναι να ελέγχουν είναι αν υπάρχει κάποιος κωδικός ή κάποια περιγραφή σε ένα συγκεκριμένο πίνακα της βάσης δεδομένων. Σκοπός είναι να αποφεύγονται οι καταχωρήσεις με τον ίδιο κωδικό (αυτό βέβαια δεν θα το άφηνε ούτε η βάση δεδομένων αφού συνήθως ο κωδικός είναι και κύριο κλειδί) και οι καταχωρήσεις με την ίδια περιγραφή. Αν υπάρχει ήδη καταχωρημένος ίδιος κωδικός ή περιγραφή τότε η stored procedure που καλέστηκε επιστρέφει τον αριθμό «1» (ένα) αλλιώς επιστρέφει τον αριθμό «0» (μηδέν). Ο αριθμός επιστρέφεται στη εφαρμογή οι οποία με βάση το αποτέλεσμα εμφανίζει τα ανάλογα μηνύματα και μεταφέρεται σε διαφορετικά σημεία του κώδικα.

Στην εικόνα που ακολουθεί εμφανίζεται ο κώδικας μιας Stored Procedure ο οποίος είναι παρόμοιος με τις υπόλοιπες που κάνουν του ίδιου τύπου έλεγχο. Στο μόνο που διαφοροποιείται είναι στα δεδομένα που δέχεται σαν όρισμα και στον πίνακα στον οποίο ψάχνει για να κάνει τον έλεγχο.

```
ALTER PROCEDURE [dbo].[CheckEidosKod]
    -- Add the parameters for the stored procedure here
    (@Kod nvarchar(30),
     @RetVal smallint output)
     --RetVal=0 den xrisimopoieitai aftos o kodikos
     --RetVal=1 o kodikos xrisimopoieitai

 AS
if exists (SELECT * FROM TblEidos   WHERE TblEidos.KodEidos  =@Kod)
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    SET @RetVal =1
    RETURN @RetVal

END
 ELSE
BEGIN
    SET @RetVal =0
    RETURN @RetVal
END
```

Όπως μπορείτε να παρατηρήσετε στη εικόνα ο έλεγχος γίνεται με το «exists», το οποίο επιστρέφει θετική τιμή αν ισχύει το select που υπάρχει μέσα στην παρένθεση. Δηλαδή αν υπάρχει γραμμή που να έχει κωδικό ίδιο με τον κωδικό που έχει δοθεί σαν όρισμα.

Με την ίδια λογική λειτουργούν και οι υπόλοιπες Stored Procedures που κάνουν τους αντίστοιχους ελέγχους. Παρακάτω εμφανίζεται μια λίστα με όλες τις Stored Procedures και μια σύντομη περιγραφή για το τι κάνουν.

- CheckEidosKod: ελέγχει αν υπάρχει ήδη ο κωδικούς του είδους, που θέλει ο χρήστης να καταχωρήσει, σε κάποιο άλλο είδος.
- CheckEidosDesc: γίνεται έλεγχος αν η περιγραφή που θέλει ο χρήστης να δώσει σε ένα είδος ανήκει σε κάποιο άλλο.
- CheckKatEidosKod: ελέγχει αν ο κωδικός της κατηγορίας που του δόθηκε σαν όρισμα είναι διαθέσιμος ή όχι.
- CheckKatEidosDesc: γίνεται έλεγχος αν η περιγραφή που δόθηκε σαν όρισμα υπάρχει ήδη στον πίνακα των κατηγοριών των ειδών.

- CheckMonMetrKod: γίνεται έλεγχος αν στον πίνακα των μονάδων μέτρησης υπάρχει ήδη καταχωρημένος ο κωδικός που δίνεται.
- CheckMonMetrDesc: ελέγχει αν η περιγραφή που δίνεται ανήκει σε κάποια άλλη μονάδα μέτρησης.
- CheckKatastKod: ο έλεγχος εδώ γίνεται στον πίνακα των καταστάσεων των εντολών παραγωγής και συγκεκριμένα στον αν υπάρχει ήδη καταχωρημένος ο κωδικός που δίνεται.
- CheckKatastDesc: αυτό που ελέγχετε σε αυτή την Stored Procedure είναι αν υπάρχει καταχωρημένη κατάσταση εντολής που να έχει περιγραφή ίδια με αυτήν που θέλει ο χρήστης να καταχωρήσει.
- CheckSistatiko: ο έλεγχος εδώ γίνεται στον πίνακα των συνταγών παραγωγής. Εδώ η Stored Procedure ελέγχει αν ένα είδος υπάρχει ήδη σαν συστατικό κάποιου άλλου είδους.
- CheckEidosEntoliParagogis: ελέγχει αν μέσα σε μια συγκεκριμένη εντολή παραγωγής είναι ήδη καταχωρημένο ένα είδος. Ο έλεγχος γίνεται με βάση τον κωδικό του είδους και τον αριθμό της εντολής παραγωγής.

Εκτός από τις Stored Procedures που καλούνται όταν ο χρήστης προσπαθήσει να εισάγει ή να ενημερώσει κάποιο πίνακα υπάρχουν και κάποιες που καλούνται από την εφαρμογή όταν ο χρήστης προσπαθήσει να κάνει διαγραφή δεδομένων από ένα πίνακα. Αυτές οι Stored Procedures ελέγχουν αν αυτό που θέλει ο χρήστης να διαγράψει χρησιμοποιείται από κάποιον άλλο πίνακα ώστε να τον αποτρέψει. Φυσικά ο χρήστης αποτρέπεται και από τη βάση δεδομένων λόγο των συσχετίσεων που έχουν καταχωρηθεί (βλ. υποκεφάλαιο 4.2). Παρόλα αυτά όμως γίνεται ο έλεγχος αυτός ώστε να αποτραπεί η δημιουργία και η εμφάνιση λάθους από την βάση δεδομένων.

Τέτοιες Stored Procedures έχουν τη λογική που εμφανίζεται στην εικόνα παρακάτω. Θα παρατηρήσουμε ότι ο έλεγχος γίνεται πάλι με την χρήση του «exists».

```
ALTER PROCEDURE [dbo].[CheckIfEidosUsing]
    -- Add the parameters for the stored procedure here

    (@Kod nvarchar(30),
     @RetVal smallint output)
     --RetVal=0 den xrisimopoieitai afto to eidos
     --RetVal=1 to eidos  xrisimopoieitai

AS
BEGIN
    SET NOCOUNT ON;

    if exists (SELECT * FROM TblSintages  WHERE KodEidos =@Kod)
        BEGIN
            SET @RetVal =1
            RETURN @RetVal

        END
   ELSE If exists (SELECT * FROM TblEntoliParagogisD WHERE KodEidos =@Kod)
        BEGIN
            SET @RetVal =1
            RETURN @RetVal
        END
    ELSE
        BEGIN
            SET @RetVal =0
            RETURN @RetVal
        END
END
```

Οι Stored Procedures που κάνουν τέτοιου είδους έλεγχο εμφανίζονται παρακάτω και μαζί τους εξηγούν τον έλεγχο που κάνουν.

- CheckIfEidosUsing: ελέγχει αν ένα συγκεκριμένο είδος χρησιμοποιείται, σε συνταγές παραγωγής ή εντολές παραγωγής, με βάση τον κωδικό του.
- CheckIfKatUsing: ελέγχει αν μια συγκεκριμένη κατηγορία είδους χρησιμοποιείται σε κάποιο είδος. Ο έλεγχος γίνεται με βάση τον κωδικό της κατηγορίας.
- CheckIfMonMetrUsing: με βάση τον κωδικό της μονάδας μέτρησης που δίνεται ελέγχεται αν αυτή η μονάδα μέτρησης εμφανίζεται σε κάποιο είδος.
- CheckIfKatastUsing: εδώ ελέγχεται με βάση τον κωδικό της, αν η κατάσταση εντολής παραγωγής που δίνεται χρησιμοποιείται σε κάποια εντολή παραγωγής.

Επιπλέον υπάρχει μια Stored Procedure η οποία δεν κάνει κάποιον έλεγχο όπως τους προηγούμενος. Καλείται από την εφαρμογή για να δει αν ένα είδος, με βάση τον κωδικό του, έχει συνταγή παραγωγής. Δηλαδή αν υπάρχει κάποια γραμμή στον πίνακα των συνταγών που να έχει κωδικό παραγόμενου είδους ίσο με αυτόν που δίνεται. Η Stored Procedure έχει το όνομα «CheckIfEidosHasRecipe» και ο κώδικας της φαίνεται στην ακόλουθη εικόνα.

```
ALTER PROCEDURE [dbo].[CheckIfEidosHasRecipe]
    -- Add the parameters for the stored procedure here
    (@KodParag nvarchar(30),
     @RetVal smallint output)
     --RetVal=0 to prohon den exei sintagi paragogis
     --RetVal=1 to prohon exei sintagi paragogis

AS
if exists (SELECT * FROM TblSintages  WHERE KodParagEidos=@KodParag)
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    SET @RetVal =1
    RETURN @RetVal

END
ELSE
BEGIN
    SET @RetVal =0
    RETURN @RetVal
END
```

Ο προγραμματισμός μέσα στη βάση δεδομένων τελειώνει με την Stored Procedure «NeaEntoliParagogis», η οποία καλείται όταν θέλουμε να δημιουργήσουμε μια νέα εντολή παραγωγής. Δεν δέχεται κανένα όρισμα αλλά επιστρέφει τον αριθμό της εντολής παραγωγής που μόλις δημιούργησε. Χρειάστηκε να γίνει η μέθοδος αυτή για να μπορούμε να πάρουμε την ημερομηνία από το σύστημα του υπολογιστή μας με την μορφή ακριβώς που την δέχεται ο SQL Server. Με αυτόν τον τρόπο αποφεύγουμε τροποποιήσεις της ώρας μέσα στον κώδικα της εφαρμογής ώστε να την μετατρέψουμε στην μορφή που θέλουμε. Επιπλέον η μέθοδος αυτή μας επιστρέφει τον κωδικό της εντολής παραγωγής που δίνεται αυτόματα από τη βάση δεδομένων. Έτσι έχουμε τον

κωδικό της εντολής στη εφαρμογή ώστε να μπορέσουμε να κάνουμε τις προσθήκες που θέλει ο χρήστης στην εντολή παραγωγής και στα δεδομένα της. Η Stored Procedure αυτή φαίνεται παρακάτω.

```
ALTER PROCEDURE [dbo].[NeaEntoliParagogis]
    -- Add the parameters for the stored procedure here

    (@ArEntolis int output)
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    DECLARE @Hmerominia datetime

     set @Hmerominia =GETDATE()
     INSERT INTO TblEntoliParagogisH (Hmerominia,IsPrinted ) VALUES(@Hmerominia,0)

     set @ArEntolis = (SELECT ArEntolisparag FROM TblEntoliParagogisH WHERE Hmerominia=@Hmerominia)
     return @ArEntolis
END
```

Με αυτή την Stored Procedure τελειώνει και η παρουσίαση της βάσης δεδομένων που διαθέτει η εφαρμογή. Στο κεφάλαιο αυτό εμφανίστηκαν λεπτομερώς οι διάφοροι πίνακες της βάσης δεδομένων καθώς και οι συσχετίσεις που υπάρχουν μεταξύ τους. Επιπλέον παρουσιάστηκαν οι συναρτήσεις – μέθοδοι που δημιουργήθηκαν ώστε να καλούνται από την εφαρμογή και να ελέγχουν κυρίως τα δεδομένα της βάσης. Το πώς οι πίνακες αυτοί και οι συναρτήσεις τους χρησιμοποιούνται από την εφαρμογή θα αναφερθεί σε επόμενο κεφάλαιο.

Κεφάλαιο 3
Η Εφαρμογή Προγραμματιστικά

Στο κεφάλαιο αυτό θα γίνει αναφορά στην εφαρμογή από την πλευρά του προγραμματιστή. Σκοπός του κεφαλαίου αυτού είναι να παρουσιάσει τον τρόπο με τον οποίο υλοποιήθηκαν οι διάφορες φόρμες τις εφαρμογής και οι λειτουργίες τους.

Για να μπορέσει να γίνει κατανοητό το κεφάλαιο αυτό θα προσπαθήσουμε να πάρουμε μια-μια τις φόρμες της εφαρμογής και να εξηγούμε τα σημαντικότερα και ουσιώδη μέρη του κώδικα.

3.1 Φόρμα «Διαχείρισης Κατηγοριών»

Η φόρμα αυτή έχει σκοπό να διαχειρίζεται τις διάφορες κατηγορίες ειδών που θέλει ο χρήστης να εισάγει στην εφαρμογή. Είναι άμεσα συσχετισμένη με τον πίνακα «Κατηγορίες Ειδών» (TblKatEidos) και έχει το όνομα «FrmKatEidos».

Στη φόρμα αυτή μέσα σε ένα Data Grid View εμφανίζονται τα δεδομένα του πίνακα «TblKatEidos». Το Data Grid View δεν είναι συνδεδεμένο με τον πίνακα αυτό αλλά τα δεδομένα του πίνακα εμφανίζονται μέσω κώδικα ο οποίος δημιουργεί γραμμές στο Data Grid View και περνά σε αυτές τα στοιχεία που υπάρχουν στον πίνακα.

```
DataGridView1.Rows.Clear()

Dim cmdSelect As String
cmdSelect = "SELECT KodKatEidos,DescKatEidos,Sintagi FROM TblKatEidos"

If SearchWithFilter Then
    Dim filter As String = ToolStripTxtSearch.Text
    filter = filter.Replace("*", "%")
    cmdSelect &= " WHERE DescKatEidos LIKE '" & filter & "'"
End If

Dim GridViewColumns As Integer = DataGridView1.Columns.Count
Dim cmd As SqlCommand
Dim rdr As SqlDataReader = Nothing

Try

    cmd = New SqlCommand(cmdSelect, Conn)
```

```
    rdr = cmd.ExecuteReader

    While rdr.Read()
        DataGridView1.Rows.Add()
        Dim c As Integer
        For c = 0 To GridViewColumns - 1
            Dim timi As String = rdr.Item(c).ToString
            DataGridView1.Item(c, DataGridView1.Rows.Count - 1).Value =
timi
         Next
      End While
      rdr.Close()
      rdr = Nothing

      DataLoaded = True
      Catch ex As Exception
      If Not (rdr Is Nothing) Then
            rdr.Close()
      End If
      MsgBox(ex.Message)
End Try
```

Στο πάνω μέρος της φόρμας υπάρχει ένα Tool Strip στο εμφανίζοναται κάποια κουμπιά με τα οποία ο χρήστης μπορεί να κάνει κάποιες λειτουργίες. Ένα από αυτά είναι το κουμπί της αναζήτησης με το οποίο δίνεται η δυνατότητα στον χρήστη να κάνει αναζήτηση με βάση την περιγραφή της κατηγορίας. Στην αναζήτηση μπορεί να χρησιμοποιηθεί και ο χαρακτήρας αστεράκι «*» ο οποίος συμβολίζει το οτιδήποτε. Ο κώδικας της αναζήτησης είναι ο ίδιος κώδικας με αυτόν που φορτώνονται τα δεδομένα μόλις ανοίξει η φόρμα (κώδικας που εμφανίζεται παραπάνω), με την μόνη διαφορά ότι στο Select που δημιουργείται δυναμικά προστίθεται και ένα Where τμήμα που έχει την αναζήτηση του χρήστη.

Ο χρήστης έχει τη δυνατότητα να προσθέσει νέες κατηγορίες στη βάση δεδομένων ή να τροποποιήσει υπάρχουσες. Για να προσθέσει νέα γραμμή πρέπει να πατήσει το κυμπί «+» από το Tool Strip. Αυτό που θα γίνει είναι να εμφανιστεί μια νέα γραμμή στο Data Grid View στην οποία ο χρήστης θα περάσει δεδομένα. Ο δείκτης της γραμμή στο Data Grid View θα αποθηκευτεί μέσα σε ένα array ο οποίος κρατά τους δείκτες των καινούργιων γραμμών. Μέσα σε διαφορετικό array αποθηκεύονται οι δείκτες των γραμμών στις οποίες ο χρήστης έχει κάνει αλλαγές, δηλαδή σε εγγραφές που είναι καταχωρημένες στον πίνακα. Όταν ο χρήστης αλλάξει μια ήδη υπάρχουσα γραμμή τότε γίνεται ένας έλεγχος ποίο στοιχείο της εγγραφής τροποποίησε. Αν τροποποιήθηκε ο κωδικός της κατηγορίας υπάρχει ένας άλλος πίνακας (array) στον οποίο καταχωρείται και ο

δείκτης της γραμμής που δέχτηκε την αλλαγή και ο προηγούμενος κωδικός που είχε. Το ίδιο συμβαίνει και όταν αλλάζει η περιγραφή της κατηγορίας.

Ο χρήστης μπορεί να κάνει όσες αλλαγές και όσες νέες εγγραφές θέλει. Για να μπορέσει να τις αποθηκεύσει όμως, θα πρέπει να πατήσει το πλήκτρο «αποθήκευση». Πατώντας το πλήκτρο αποθήκευση καλείται μια μέθοδος με το όνομα «SaveMethod» η οποία κάνει τους κατάλληλους ελέγχους και καλεί τις κατάλληλες μεθόδους για να καταχωρήσει τις καινούργιες γραμμές στον πίνακα και να ενημερώσει τις προϋπάρχουσες.

```
Private Sub SaveMethod()

  Dim ProblemNew As ArrayList = New ArrayList

  Dim Row As Integer
  Dim Desc As String
  Dim Kod As String
  Dim Sintagi As Short

  For i As Short = 0 To NewRowsCount

     Row = NewRowsIndex(i)
     Desc = DataGridView1.Item(ColumnDescKat.Index, Row).Value
     Kod = DataGridView1.Item(ColumnKodKat.Index, Row).Value
     Sintagi = DataGridView1.Item(ColumnSintagi.Index, Row).Value

      If Desc <> "" AndAlso Kod <> "" Then
         If CheckKatEidosDesc(Desc) = 0 AndAlso CheckKatEidosKod(Kod) =
0 Then
              InsertKatEidos(Kod, Desc, Sintagi, Row)
         Else
              MessageBox.Show("Η κατηγορία με κωδικό '" & Kod & "' και
περιγραφή '" & Desc & "' δεν μπορεί να καταχωρηθεί")
              ProblemNew.Add(Row)
         End If
      Else
         MessageBox.Show("Δεν έχετε καταχωρήσει όλα τα δεδομένα για την
εγγραφή στη γραμμή '" & Row + 1 & "'")
         ProblemNew.Add(Row)
       End If
  Next


  For i As Short = 0 To ChangedRowsCount

     Row = ChangedRowsIndex(i)
     Desc = DataGridView1.Item(ColumnDescKat.Index, Row).Value
     Kod = DataGridView1.Item(ColumnKodKat.Index, Row).Value
     Sintagi = DataGridView1.Item(ColumnSintagi.Index, Row).Value

     Dim Saved As Boolean = False
```

```
     If Desc <> "" AndAlso Kod <> "" Then
        Dim found As Boolean = False
        Dim PreKod As String = ""


        For j As Short = 0 To ChangedKodCount   'έλεγχος αν αυτό που
έχει αλλαχθεί είναι ο κωδικός

           If ChangedKod(0, i) = Row Then
              If CheckKatEidosKod(Kod) = 0 Then
                  PreKod = ChangedKod(1, i)
                      If PreKod <> Kod Then
                            found = True
                            Exit For
                       End If
                Else
                   PreKod = ChangedKod(1, i)
                   MessageBox.Show("Ο κωδικός '" & Kod & "'
χρησιμοποιείται ήδη και δεν μπορεί να χρησιμοποιηθεί ξανά")
                   DataGridView1.Item(ColumnKodKat.Index, Row).Value =
PreKod 'βάζουμε τον κωδικό που υπήρχε στη γραμμή πριν προσπαθήσει ο
χρήστης για την αλλαγή
                    GoTo continueHere
                End If
             End If
          Next

          For k As Short = 0 To ChangedDescCount  'Ελεγχος αν έχει
αλλαχθεί η περιγραφή

               If ChangedDesc(k) = Row Then
                   If CheckKatEidosDesc(Desc) = 0 Then
                       If found = True Then 'Αν έχουν αλλαχθεί και τα
δύο
                           UpdateKatEidos(Kod, Desc, Sintagi,
PreKod,Row)
                            Saved = True
                       Else 'έχει αλλαχθεί μόνο η περιγραφή
                          UpdateKatEidos(Kod, Desc, Sintagi, Kod, Row)
                          Saved = True
                           End If
                       Else
                           MessageBox.Show("Η κατηγορία με την
περιγραφή '" & Desc & "' είναι ήδη καταχωρημένη στη βάση δεδομένων")
                           GoTo continueHere
                       End If
                   End If
                Next

                If found = True And Saved = False Then 'έχει αλλαχθεί
τελικά μόνο ο κωδικός
                    UpdateKatEidos(Kod, Desc, Sintagi, PreKod, Row)
                    Saved = True
                End If

            End If

            If Saved = False Then 'αλλάχθηκε μόνο η συνταγή
                UpdateKatEidos(Kod, Desc, Sintagi, Kod, Row)
            End If
```

```
continueHere: Continue For
        Next

        NewRowsCount = -1
        ReDim NewRowsIndex(0)
        If ProblemNew.Count > 0 Then
            NewRowsCount = ProblemNew.Count - 1
            ReDim NewRowsIndex(NewRowsCount)
            NewRowsIndex = ProblemNew.ToArray
            ProblemNew.Clear()
        End If
        ChangedRowsCount = -1
        ReDim ChangedRowsIndex(0)
        ChangedRowsIndex(0) = -1
        ChangedDescCount = -1
        ReDim ChangedDesc(0)
        ChangedKodCount = -1
        ReDim ChangedKod(1, 0)

End Sub
```

Αρχικά στη μέθοδο αυτή ορίζονται κάποιες μεταβλητές. Οι μεταβλητές αυτές είναι τα πεδία της κάθε εγγραφής. Επιπλέον εκτός από τα πεδία ορίζεται και ένα ArrayList στο οποίο θα καταχωρούνται οι δείκτες των καινούργιων γραμμών που δεν μπόρεσαν να καταχωρηθούν στον πίνακα για κάποιο λόγο.

Η μέθοδος ξεκινά με ένα For Loop στο οποίο παίρνει μια-μια τις καινούργιες γραμμές που δημιουργούσε ο χρήστης, με βάση τους δείκτες τους οι οποίοι υπάρχουν μέσα στον πίνακα «NewRowsIndex». Για κάθε νέα γραμμή παίρνει από το Data Grid View τα στοιχεία που καταχώρησε ο χρήστης. Στη συνέχεια ελέγχει αν ο χρήστης έχει καταχωρήσει κάποιο κωδικό και περιγραφή στην εγγραφή. Αν δεν έχει καταχωρήσει τότε εμφανίζεται ένα μήνυμα και η γραμμή με τον δείκτη της προστίθεται στο ArrayList. Αν ο χρήστης έχει καταχωρήσει τα στοιχεία που απαιτούνται τότε καλούνται οι μέθοδοι «CheckKatEidosDesc» και «CheckKatEidosKod» μέσα στις οποίες στέλνονται η περιγραφή και ο κωδικός που έδωσε ο χρήστης αντίστοιχα. Αν το αποτέλεσμα και των δύο αυτών μεθόδων είναι μηδέν τότε σημαίνει ότι δεν υπάρχει μέσα στη βάση δεδομένων κάποια εγγραφή που να έχει ίδιο κωδικό ή περιγραφή. Αφού σιγουρευτεί η μέθοδος ότι τα στοιχεία που θα περαστούν είναι μοναδικά τότε καλείται η μέθοδος InsertKatEidos η οποία κάνει τη εισαγωγή της καινούργιας γραμμής στη βάση δεδομένων.

Στη συνέχεια ένα άλλο For Loop παίρνει μια-μια τις γραμμές που δέχτηκαν αλλαγές. Για κάθε μια γραμμή από αυτές ελέγχει αν βρίσκεται μέσα στον πίνακα

των γραμμών στις οποίες αλλάχθηκε ο κωδικός της κατηγορίας. Ο έλεγχος αυτός γίνεται με ένα δεύτερο For Loop. Αν η γραμμή βρίσκεται μέσα στον πίνακα αυτό τότε καλείται η μέθοδος «CheckKatEidosKod» για να διαπιστωθεί ότι ο κωδικός που έχει δώσει ο χρήστης δεν ανήκει σε κάποια άλλη κατηγορία. Αν ο κωδικός ανήκει σε κάποια κατηγορία τότε εμφανίζεται το κατάλληλο μήνυμα και με ένα «go to» ο κώδικας μεταφέρεται σε ένα άλλο σημείο για να συνεχίσει στην επόμενη τροποποιημένη γραμμή. Αν τελικά ο κωδικός είναι διαθέσιμος τότε μέσα σε μια μεταβλητή κρατάμε τον προηγούμενο κωδικό που είχε η εγγραφή, ώστε να μπορέσουμε μετά να την τροποποιήσουμε, και θέτουμε σε ένα flag την τιμή true για να δηλώσουμε ότι στη γραμμή τροποποιήθηκε ο κωδικός της κατηγορίας. Με ένα τρίτο For Loop αναζητούμε την γραμμή στον πίνακα με τις γραμμές στις οποίες αλλάχθηκε η περιγραφή. Βρίσκοντας την γραμμή αυτή το πρώτο που γίνεται είναι να ελεγχθεί αν η περιγραφή που δόθηκε από τον χρήστη είναι διαθέσιμη και δεν ανήκει σε κάποια άλλη κατηγορία. Ο έλεγχος αυτός γίνεται καλώντας τη μέθοδο «CheckKatEidosDesc». Αφού περάσει από αυτόν τον έλεγχο μετά βλέπουμε αν στην ίδια εγγραφή ο χρήστης είχε αλλάξει και τον κωδικό, αυτό φαίνεται από το flag που αναφέραμε προηγουμένως, ή αν έχει αλλαχθεί μόνο η περιγραφή. Αυτό γίνεται για να αποφασίσουμε ποια δεδομένα θα στείλουμε μέσα στην μέθοδο που θα ενημερώσει την εγγραφή («UpdateKatEidos»). Ενημερώνουμε την εγγραφή και ορίζουμε ένα δεύτερο flag σε true ο οποίο δείχνει ότι η εγγραφή έχει ενημερωθεί. Αν τελικά η εγγραφή δεν έχει ενημερωθεί αλλά έχει τροποποιημένο κωδικό τότε καλείται η «UpdateKatEidos» με διαφορετικά ορίσματα, ενώ αν τελικά αλλάχθηκε μόνο η ένδειξη για συνταγή παραγωγής τότε καλούμε πάλι την μέθοδο «UpdateKatEidos» με τα κατάλληλα ορίσματα.

Αφού φτάσει στο τέλος της η μέθοδος και αποθηκεύσει όλες τις νέες και τροποποιημένες εγγραφές μηδενίζει τους μετρητές που χρησιμοποίησε και αδειάζει όλους τους βοηθητικούς πίνακες που χρησιμοποίησε.

Οι έλεγχοι για το αν ένας κωδικός ή μια περιγραφή είναι διαθέσιμα γίνεται με την βοήθεια των μεθόδων «CheckKatEidosKod» και «CheckKatEidosDesc». Οι μέθοδοι αυτοί στην ουσία περιέχουν κώδικα ο οποίος καλεί τις αντίστοιχες Stored Procedures στον SQL Server. Στο κεφάλαιο που αναφερόταν στη βάση

δεδομένων της εφαρμογής έγινε αναφορά στις Stored Procedures που καλεί και χρησιμοποιεί η εφαρμογή. Ο κώδικας για την κλήση μιας τέτοιας Procedure φαίνεται παρακάτω. Ανάλογα με την μέθοδο που καλείται δίνεται και το αντίστοιχο όρισμα.

```
Private Function CheckKatEidosKod(ByVal Kod As String) As Short
   'RetVal=0 o kodikos den einai idi kataxorimenos
   'RetVal=1 o kodikos einai  kataxorimenos
   Try
     Dim ReturnValue As String

     Dim cmdProc As New SqlCommand("CheckKatEidosKod", Conn)
     cmdProc.CommandType = CommandType.StoredProcedure
     cmdProc.Parameters.AddWithValue("@Kod", Kod)
     cmdProc.Parameters.Add("@RetVal",  SqlDbType.SmallInt).Direction =
ParameterDirection.Output
     cmdProc.ExecuteNonQuery()

     ReturnValue = cmdProc.Parameters("@RetVal").Value

     Return ReturnValue

     Catch ex As Exception
            MessageBox.Show(ex.Message)
     End Try

 End Function
```

Όταν η μέθοδος της αποθήκευσης φτάσει στο σημείο που θέλει να καταχωρήσει μια νέα γραμμή στον πίνακα, καλεί την μέθοδο «InsertKatEidos» στην οποία στέλνει τα κατάλληλα ορίσματα ώστε να γίνει η καταχώρηση στη βάση δεδομένων. Αυτό που κάνει η μέθοδος αυτή είναι να εκτελεί στη βάση δεδομένων ένα Insert. Ο κώδικας της φαίνεται στην συνέχεια.

```
Private Sub InsertKatEidos(ByVal KodKat As String, ByVal Desc As String,
ByVal Sintagi As Short, ByVal Row As Integer)

      Dim cmdInsert As String
      cmdInsert = "INSERT INTO TblKatEidos VALUES('" & KodKat & "','" &
Desc & "'," & Sintagi & ")"
      Try
         Dim cmd As SqlCommand
         cmd = New SqlCommand(cmdInsert, Conn)
          cmd.ExecuteNonQuery()

          DataGridView1.Rows(Row).DefaultCellStyle.BackColor =
Color.White

        Catch ex As Exception
            MessageBox.Show("Δημιουργήθηκε κάποιο πρόβλημα κατά την
καταχώρηση της εγγραφής")
        End Try
    End Sub
```

Παρόμοια είναι και η μέθοδος η οποία ενημερώνει τη βάση για της αλλαγές. Στη μέθοδο «UpdateKatEidos» όπως ονομάζεται αντί για ένα «Insert» εκτελείται μια εντολή «Update» στη βάση δεδομένων. Ο κώδικας της είναι ο ακόλουθος.

```
Private Sub UpdateKatEidos(ByVal NewKod As String, ByVal Desc As String,
ByVal Sintagi As Short, ByVal PreKod As String, ByVal Row As Integer)

    Dim cmdUpdate As String
    cmdUpdate = "UPDATE TblKatEidos SET KodKatEidos='" & NewKod & "',
DescKatEidos='" & Desc & "', Sintagi=" & Sintagi & " WHERE
KodKatEidos='" & PreKod & "'"

     Try
       Dim cmd As New SqlCommand(cmdUpdate, Conn)
       cmd.ExecuteNonQuery()

       DataGridView1.Rows(Row).DefaultCellStyle.BackColor = Color.White

     Catch ex As Exception
        MessageBox.Show(ex.Message)
     End Try
 End Sub
```

Εκτός από της λειτουργίες της εισαγωγής και της ενημέρωσης ο χρήστης έχει στη διάθεση του τη λειτουργία της διαγραφής. Για να διαγράψει μια εγγραφή θα πρέπει πρώτα να την επιλέξει και να πατήσει το κουμπί της διαγραφής από το Tool Strip. Ο χρήστης έχει το δικαίωμα να επιλέξει για διαγραφή περισσότερες από μια εγγραφές. Πατώντας το κουμπί αυτό εκτελούνται για κάθε εγγραφή που επέλεξε ο χρήστης να διαγράψει μια σειρά από εντολές.

Αρχικά μια-μια η γραμμή με τον δείκτη της μπαίνει σε ένα βοηθητικό πίνακα (array) στον οποίο αποθηκεύονται οι γραμμές για διαγραφή. Στην συνέχεια γίνεται έλεγχος αν η γραμμή που θέλει να διαγράψει είναι κάποια γραμμή που μόλις έχει προσθέσει αλλά δεν έχει αποθηκεύσει. Αν η γραμμή είναι καινούργια τότε συνεχίζει για να ελέγξει την επόμενη γραμμή.

```
IndexForDelete(i) = DataGridView1.SelectedRows.Item(i).Index

If isNewRow(IndexForDelete(i)) Then
    Continue For
End If
```

Η μέθοδος «isNewRow» ελέγχει αν ο δείκτης της γραμμής που της έχει δοθεί σαν όρισμα εμφανίζεται μέσα στον βοηθητικό πίνακα στον οποίο αποθηκεύονται οι δείκτες των καινούργιων γραμμών.

Αφού βεβαιωθούμε ότι η γραμμή δεν είναι καινούργια τότε γίνεται ένας δεύτερος έλεγχος αν ο κωδικός της κατηγορίας που έχει η γραμμή προς διαγραφή έχει προηγουμένως αλλαχθεί. Αυτό γίνεται για να διαγραφεί από τη βάση δεδομένων η σωστή εγγραφή. Αν ο κωδικός της γραμμής έχει τροποποιηθεί τότε ο προηγούμενος κωδικός που είχε η γραμμή αποθηκεύεται μέσα σε μια μεταβλητή ώστε να χρησιμοποιηθεί αργότερα.

```
For j As Short = 0 To ChangedKodCount
    If ChangedKod(0, j) = IndexForDelete(i) Then
          KodKat = ChangedKod(1, j)
    End If
Next
```

Ένας τελευταίος έλεγχος που γίνεται είναι αν η εγγραφή που θέλει ο χρήστης να διαγραφεί, χρησιμοποιείται. Δηλαδή αν υπάρχει στον πίνακα των ειδών κάποιο είδος που να έχει σαν κατηγορία την κατηγορία που θέλει ο χρήστης να διαγράψει. Ο έλεγχος αυτός γίνεται με την βοήθεια της μεθόδους «CheckIfKatUsing» και σαν όρισμα παίρνει τον κωδικό της κατηγορίας. Αυτός είναι και ο λόγος που ελέγχεται αν ο κωδικός έχει τροποποιηθεί πριν αποφασίσει ο χρήστης να διαγράψει την εφγγραφή. Αν η κατηγορία χρησιμοποιείται τότε εμφανίζεται ένα μήνυμα που ενημερώνει τον χρήστη ότι η κατηγορία δεν μπορεί να διαγραφεί. Σε αντίθετη περίπτωση καλείται η μέθοδος «DeleteKatEidos» για να διαγράψει την κατηγορία.

```
RetVal = CheckIfKatUsing(KodKat)
If RetVal = 1 Then
   MessageBox.Show("Η κατηγορία ειδών '" & DescKat & "' έχει
καταχωρημένα κάποια είδη/προϊόντα και δεν μπορεί να διαγραφεί",
"Διαγραφή Κατηγορίας", MessageBoxButtons.YesNo,
MessageBoxIcon.Information)

   IndexForDelete(i) = -1
Else
   DeleteKatEidos(KodKat)
End If
```

Στο τέλος της μεθόδου καλείται η μέθοδος «RemoveDeletedRows» η οποία διαγράφει από το Data Grid View τις γραμμές που έχουν διαγραφεί.

Αυτά ήταν κατά κύριο λόγο τα σημαντικότερα σημεία στο προγραμματιστικό κομμάτι της φόρμας αυτής. Ο τρόπος που λειτουργεί η φόρμα αυτή είναι κοινός για τις περισσότερες από τις φόρμες τις εφαρμογής κυρίως σε αυτές που ο χρήστης μπορεί να διαχειριστεί κάποιο πίνακα της βάσης δεδομένων.

3.2 Φόρμα «Διαχείρισης Μονάδων Μέτρησης»

Ο κώδικας της φόρμας αυτής δημιουργήθηκε ώστε να δώσει στο χρήστη της εφαρμογής τη δυνατότητα να διαχειριστεί τον πίνακα των μονάδων μέτρησης («TblMonMetr»). Η φόρμα αυτή έχει την ονομασία «FrmMonMetr» και επιτρέπει στον χρήστη να δημιουργήσει, να τροποποιήσει και να διαγράψει εγγραφές από τον πίνακα των μονάδων μέτρησης.

Ουσιαστικά η φόρμα αυτή εκτελεί τις ίδιες λειτουργίες που εκτελεί και η φόρμα για την διαχείριση των κατηγοριών. Οι μέθοδοι με τις οποίες διαχειρίζεται τις λειτουργίες αυτές έχουν την ίδια λογική με αυτές τις φόρμας «FrmKatEidos». Οι μόνες διαφορές που παρατηρούνται είναι στα ονόματα των μεθόδων και στις Stored Procedures που καλούνται. Για εισαγωγή, ενημέρωση και διαγραφή η εντολές απευθύνονται στον σωστό πίνακα της βάσης δεδομένων.

Επειδή στη φόρμα αυτή δεν υπάρχει κάτι ουσιαστικά διαφορετικό από την φόρμα «Διαχείρισης Κατηγοριών» δεν θα γίνει αναλυτική αναφορά και παρουσίαση του κώδικα της.

3.3 Φόρμα «Διαχείρισης Καταστάσεων Εντολής Παραγωγής»

Η φόρμα «Διαχείρισης Καταστάσεων Εντολής Παραγωγής» ή αλλιώς «FrmKatastasiEntolis» όπως είναι το όνομα της είναι άμεσα συνδεδεμένη με τον πίνακα «TblKatastEntolis» της βάσης δεδομένων.

Σε αυτήν τη φόρμα ο χρήστης έχει την δυνατότητα να διαχειριστεί τον πίνακα «TblKatastEntolis». Οι λειτουργίες που του παρέχονται είναι οι ίδιες με τις φόρμες «Διαχείριση Κατηγοριών» και «Διαχείριση Μονάδων Μέτρησης». Μπορεί να δημιουργήσει τις δικές του καταστάσεις στις οποίες μπορεί να είναι μια εντολή παραγωγής, να τροποποιήσει τις ήδη υπάρχουσες ή να διαγράψει μη χρησιμοποιούμενες καταστάσεις.

Οι λειτουργίες που προσφέρονται, προγραμματιστικά ακολουθούν την ίδια λογική με αυτές στην φόρμα «FrmKatEidos». Το μόνο που διαφοροποιεί τον κώδικα που βρίσκεται στις δύο φόρμες είναι ο πίνακας της βάσης δεδομένων ο οποίος ενημερώνεται και οι Stored Procedures που καλεί.

Η μόνη ουσιαστική διαφορά που υπάρχει ανάμεσα στις δύο φόρμες είναι ότι η παρούσα φόρμα όταν υπάρχει κάποια τροποποίηση στα δεδομένα της, εκτός από το να ενημερώσει τον πίνακα των καταστάσεων, ενημερώνει και τις στήλες του πίνακα των Γενικών Παραμέτρων που αναφέρονται στις προκαθορισμένες τιμές των εντολών παραγωγής. Η μέθοδος καλείται μετά την μέθοδο η οποία ενημερώνει τον πίνακα των καταστάσεων και έχει την ακόλουθη μορφή:

```
Private Sub UpdateGenParameters(ByVal NewKod As String, ByVal PreKod As
String)

        Dim cmdUpdate As String
        Dim cmd As SqlCommand
        cmdUpdate = "UPDATE TblGenParameters SET NeaEntoli='" & NewKod &
"' WHERE NeaEntoli='" & PreKod & "'"

        Try
            cmd = New SqlCommand(cmdUpdate, Conn)
            cmd.ExecuteNonQuery()
        Catch ex As Exception

        End Try

        cmdUpdate = "UPDATE TblGenParameters SET DoneEntoli='" & NewKod
& "' WHERE DoneEntoli='" & PreKod & "'"
        Try
            cmd = New SqlCommand(cmdUpdate, Conn)
            cmd.ExecuteNonQuery()
        Catch ex As Exception

        End Try

        cmdUpdate = "UPDATE TblGenParameters SET CancelEntoli='" &
NewKod & "' WHERE CancelEntoli='" & PreKod & "'"
        Try
            cmd = New SqlCommand(cmdUpdate, Conn)
            cmd.ExecuteNonQuery()
        Catch ex As Exception

        End Try
    End Sub
```

Ο λόγος που γίνεται αυτό είναι γιατί ο πίνακας των γενικών παραμέτρων δεν είναι συνδεδεμένος με κάποιον από τους υπόλοιπους πίνακες και ο λόγος είναι γιατί εκτός του ότι δεν έχει κάποιο κύριο κλειδί δεν υπάρχει κάποιο πεδίο του που να μπορεί να αντιστοιχηθεί με κάποιον πίνακα. Έτσι με έμμεσο τρόπο προσπαθούμε να ενημερώσουμε και τον πίνακα αυτό για τυχόν αλλαγές.

3.4 Φόρμα «Διαχείρισης Γενικών Παραμέτρων»

Η φόρμα των γενικών παραμέτρων είναι η φόρμα στην οποία ο χρήστης θα δώσει τις προκαθορισμένες τιμές που θέλει να υπάρχουν στην εφαρμογή, να επισημάνει και να ορίσει κάποιες τιμές. Στην εφαρμογή αυτή η φόρμα των γενικών παραμέτρων διαθέτει μόνο τρεις παραμέτρους, όσες είναι και οι στήλες στον πίνακα της βάσης δεδομένων «TblGenParameters».

Οι γενικές παράμετροι που χρησιμοποιεί η εφαρμογή αυτή είναι η προκαθορισμένη τιμή που θα παίρνει μια νέα εντολή παραγωγής, η κατάσταση στην οποία θα θεωρούνται οι εντολές παραγωγής ακυρωμένες και η κατάσταση που θα εμφανίζεται στις ολοκληρωμένες εντολές παραγωγής. Στην φόρμα αυτή υπάρχουν τρία Combo Boxes με όλες τις εγγραφές από τον πίνακα των καταστάσεων εντολών παραγωγής που υπάρχει στη βάση δεδομένων. Ο χρήστης επιλέγει την κατάσταση που θέλει να δώσει σε κάθε παράμετρο και πατάει το κουμπί αποθήκευση. Πίσω από το κουμπί αυτό αρχικά ελέγχεται αν οι παράμετροι έχουν διαφορετική τιμή ώστε να αφήσει να γίνει η αποθήκευση στη βάση δεδομένων.

```
If NeaEntoli = DoneEntoli Or NeaEntoli = CancelEntoli Or DoneEntoli =
CancelEntoli Then

MessageBox.Show("Δεν μπορούν δύο γενικές παραμέτροι να έχουν την ίδια
τιμή." & vbNewLine & "Παρακαλώ επιλέξτε διαφορετικές τιμές και
προσπαθήστε ξανά", "Γενικές Παραμέτροι")
      Exit Sub
End If
```

Αφού οι παράμετροι δεν έχουν ίδια τιμή τότε γίνεται η ενημέρωση του πίνακα

```
Dim cmdUpdate As String
cmdUpdate = "UPDATE TblGenParameters SET NeaEntoli='" & NeaEntoli & "',
DoneEntoli='" & DoneEntoli & "', CancelEntoli='" & CancelEntoli & "'"
```

```
Dim cmd As New SqlCommand(cmdUpdate, Conn)

Try
   cmd.ExecuteNonQuery()
   Changed = False
   GetDefaultValues()
Catch ex As Exception
   MessageBox.Show(ex.Message)
End Try
```

3.5 Φόρμα «Παραλαβής Ειδών»

Η φόρμα της παραλαβής των ειδών είναι μια τυπική φόρμα για παραλαβή ειδών. Δεν διαθέτει όλες εκείνες τις πληροφορίες που καταχωρεί μια εταιρεία για κάθε είδος που παραλαμβάνει. Ο σκοπός δημιουργίας της είναι καθαρά τυπικός.

Στη φόρμα αυτή υπάρχουν δύο Combo Boxes, το ένα έχει καταχωρημένο όλους τους κωδικούς των ειδών που ανήκουν σε κάποια κατηγορία η οποία δεν διαθέτει συνταγή παραγωγής και το άλλο τις περιγραφές των αντίστοιχων ειδών. Ο λόγος που ελέγχεται σε ποία κατηγορία ανήκει είναι γιατί η λογική λέει ότι μια εταιρεία μπορεί να παραλάβει μόνο μη παραγόμενα είδη, δηλαδή ύλες.

```
Dim cmdSelect As String = "SELECT KodEidos, DescEidos FROM TblEidos
WHERE KodKatEidos IN (SELECT KodKatEidos FROM TblKatEidos WHERE
Sintagi=0)"
```

Τα δύο αυτά Combo Boxes είναι συγχρονισμένα. Κάθε φορά που ο χρήστης επιλέγει ένα είδος από το ένα Combo Box εμφανίζεται στο άλλοο κωδικός ή η περιγραφή του ανάλογα με το τι έχει επιλέξει.

```
Private Sub CmbxKodEidos_SelectedValueChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles CmbxKodEidos.SelectedValueChanged

      If DataLoaded Then
            CmbxDescEidos.SelectedValue = CmbxKodEidos.Text
      nd If
End Sub


Private Sub CmbxDescEidos_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
CmbxDescEidos.SelectedIndexChanged

      If DataLoaded Then
            CmbxKodEidos.SelectedValue = CmbxDescEidos.Text
      End If
End Sub
```

Επιλέγοντας ο χρήστης το είδος που θέλει πρέπει να καταχωρήσει σε ένα Text Box την ποσότητα του είδους αυτού που έχει παραλάβει. Στο Text Box αυτό ο χρήστης μπορεί να βάλει μόνο αριθμούς και καθόλου γράμματα.

```
Private Sub TxtPosot_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TxtPosot.KeyPress
        'για να αφήνει το χρήστη να καταχωρεί μόνο αριθμούς
    If Not (Char.IsDigit(e.KeyChar) Or Char.IsControl(e.KeyChar)) Then
            e.Handled = True
    End If

End Sub
```

Για να τελειώσει η διαδικασία παραλαβής κάποιου είδους θα πρέπει να πατηθεί το κουμπί «Ενημέρωση Αποθήκης». Με το πάτημα του κουμπιού αυτού εκτελείται μια εντολή «Update» στον πίνακα «Αποθήκη» της βάσης δεδομένων.

```
Dim cmdUpdate As String
cmdUpdate = "UPDATE TblApothiki SET AvailPosot=AvailPosot+" &
TxtPosot.Text & " WHERE KodEidos='" & CmbxKodEidos.Text & "'"
```

Ο χρήστης μπορεί να καταχωρήσει όσες παραλαβές ειδών θέλει χωρίς να κλείνει και να ανοίγει την φόρμα. Κάθε φορά που καταχωρεί μια παραλαβή του γίνεται ερώτηση αν θέλει να καταχωρήσει επιπλέον προϊόντα.

3.6 Φόρμα «Διαχείρισης Ειδών»

Μέσω της φόρμας «Διαχείρισης Ειδών» ο χρήστης έχει την δυνατότητα να παρακολουθεί τα διάφορα είδη που υπάρχουν στην αποθήκη του καθώς και τα είδη που μπορεί να παράγει. Ο χρήστης μπορεί να δει όλες τις πληροφορίες που υπάρχουν στον πίνακα των Ειδών καθώς και επιπλέον την διαθέσιμη ποσότητα του κάθε είδους που από τον πίνακα της Αποθήκης.

Ομοίως με τις προηγούμενες φόρμες και εδώ ο χρήστης μπορεί να τροποποιήσει τα στοιχεία των διάφορων ειδών (όσα από αυτά του δίνεται η δυνατότητα να τροποποιήσει), να διαγράψει κάποια είδη αν δεν τα χρειάζεται πλέον ή και να καταχωρήσει καινούργια. Η τροποποίηση και η διαγραφή μπορεί να γίνεται κατευθείαν από τη φόρμα αυτή ή μέσω της «Καρτέλας Είδους». Η καρτέλα του κάθε είδους εμφανίζεται αν ο χρήστης πατήσει διπλό κλικ πάνω σε μια εγγραφή (είδος) από αυτές που εμφανίζεται στο Data Grid View.

Στη φόρμα αυτή η διαδικασία της αποθήκευσης είναι λίγο διαφορετική από τις φόρμες που αναφέρθηκαν προηγουμένως. Ο χρήστης μπορεί να κάνει αλλαγές στις εγγραφές όπως και στις προηγούμενες φόρμες αλλά σε αυτήν την περίπτωση εκτός από την περίπτωση να έχει αλλαχθεί ο κωδικός ή η περιγραφή κάποιου είδους, λαμβάνεται υπόψη και η περίπτωση αλλαγής της κατηγορίας του είδους. Οι εγγραφές στις οποίες παρατηρήθηκε αλλαγή στην κατηγορία τους αποθηκεύονται σε ένα βοηθητικό πίνακα (array) με όνομα «ChangedKatig». Στον πίνακα αυτό αποθηκεύεται ο δείκτης της γραμμής και η προηγούμενη κατηγορία του είδους. Στη μέθοδο της αποθήκευσης γίνεται ένας έλεγχος όσον αφορά την καινούργια κατηγορία του προϊόντος. Ελέγχεται αν στην καινούργια κατηγορία μπορεί να καταχωρηθεί συνταγή παραγωγής. Αν στην κατηγορία δεν μπορεί να καταχωρηθεί συνταγή παραγωγής τότε ελέγχεται αν το είδος διαθέτει κάποια συνταγή παραγωγής. Αυτό γίνεται για να ενημερώσει τον χρήστη ότι αν αλλάξει την κατηγορία του είδους θα διαγραφή και η συνταγή παραγωγής τους. Η συνέχεια της μεθόδου είναι σχετική ανάλογα με την απόφαση που θα πάρει ο χρήστης.

```
For m As Short = 0 To ChangedKatigCount
      If ChangedKatig(0, m) = Row Then
          Dim KodikosEidous As String
           If Not HaveProductionRecipe(KodKat) Then
               If found Then
                  KodikosEidous = PreKod
               Else
                  KodikosEidous = Kod
               End If
               If CheckIfEidosHasRecipe(KodikosEidous) = 1 Then
                  If MessageBox.Show("Το προϊόν/είδος στην γραμμή '" &
Row & "' διαθέτει συνταγή παραγωγής." & vbCrLf & "Η κατηγορία προϊόντος
που επιλέξατε δεν μπορεί να δεχτεί συνταγή παραγωγής." & vbCrLf & "Αν
κάνετε την αλλαγή η συνταγή παραγωγής του προϊόντος θα διαγραφεί. Θέλετε
να συνεχίσετε;", "Αλλαγή Κατηγορίας Προϊόντος", MessageBoxButtons.YesNo)
= Windows.Forms.DialogResult.Yes Then
                                        'θέλει να συνεχίσει την αλλαγή της
κατηγορίας
                           DeleteSintagi(PreKod)
                      Else 'δεν θέλει να κανει την αλλαγή της κατηγορίας
                           PreKatigoria = ChangedKatig(1, m)
                           DataGridView1.Item(ColumnKat.Index,
Row).Value = PreKatigoria
                           KodKat = PreKatigoria
                       End If
                End If
            End If
        End If
```

```
Next
```

Η διαγραφή κάποιου είδους ακολουθεί την ίδια λογική που ακολουθεί και η διαδικασία διαγραφής κάποιας εγγραφής και στις φόρμες που προαναφέρθηκαν. Για να μπορέσει να διαγραφή ένα είδος πρέπει πρώτα να ελεγχθεί αν δεν εμφανίζεται σε κάποια εντολή παραγωγής ή σε κάποια συνταγή παραγωγής κάποιου άλλου είδους. Ο έλεγχος αυτός γίνεται με την κλήση της Stored Procedure «CheckIfEidosUsing» στον SQL Server.

```
Public Function CheckIfEidosUsing(ByVal Kod As String) As Short

   Try
      Dim ReturnValue As String

      Dim cmdProc As New SqlCommand("CheckIfEidosUsing", Conn)
      cmdProc.CommandType = CommandType.StoredProcedure
      cmdProc.Parameters.AddWithValue("@Kod", Kod)
      cmdProc.Parameters.Add("@RetVal", SqlDbType.SmallInt).Direction =
ParameterDirection.Output
      cmdProc.ExecuteNonQuery()

      ReturnValue = cmdProc.Parameters("@RetVal").Value

      Return ReturnValue

   Catch ex As Exception
            MessageBox.Show(ex.Message)
   End Try

End Function
```

Η μέθοδος αυτή το μόνο που κάνει είναι να καλεί την αντίστοιχη Stored Procedure και να της στέλνει τα κατάλληλα ορίσματα. Παρατηρούμε ότι η μέθοδος αυτή είναι «Public». Ο λόγος που είναι δημόσια είναι γιατί χρησιμοποιείται και από άλλη φόρμα εκτός από την «Φόρμα των Ειδών» και έτσι χρειάστηκε να καταχωρηθεί μέσα σε ένα Module ώστε οι φόρμες που την χρησιμοποιούν να μπορούν να την καλούν.

Για να θεωρηθεί ότι ένα είδος έχει διαγραφεί δεν πρέπει να διαγραφεί μόνο από τον πίνακα των ειδών. Θα πρέπει να διαγραφούν και όλες οι αναφορές του είδους αυτού. Με την έννοια αναφορές εννοούμε ότι το είδος πρέπει να διαγραφεί από τον πίνακα της αποθήκης καθώς επίσης πρέπει να διαγραφεί και συνταγή παραγωγής του αν τυχόν υπάρχει τέτοια.

Η δημιουργία και καταχώρηση ενός νέου είδους δεν γίνεται άμεσα από αυτήν τη φόρμα. Όταν ο χρήστης θέλει να δημιουργήσει ένα νέο είδος και βρίσκεται σε

αυτήν τη φόρμα μπορεί να πατήσει το κουμπί «+». Πατώντας το κουμπί αυτό ανοίγ ε μια άλ λ ηφόρμα, η «Καρτέλ α Είδους» μέσω της οποίας μπορεί να δημιουργήσει το νέο είδος που θέλει.

Εκτός από τις βασικές αυτές λειτουργίες, στη φόρμα αυτή υπάρχει η δυνατότητα για σύνθετη αναζήτηση. Στο κάτω μέρος της φόρμας εμφανίζεται ένα πλαίσιο για αναζήτηση στο οποίο υπάρχουν διάφορες παράμετροι με τις οποίες μπορεί να κάνει αναζήτηση ο χρήστης μέσα στα είδη. Αφού ο χρήστης επιλέξει με ποιο κριτήρια θέλει να κάνει την αναζήτηση του πατά το κουμπί «Αναζήτηση». Το κουμπί αυτό αφού πατηθεί προσπαθεί να δημιουργήσει δυναμικά, με βάση τα κριτήρια αναζήτησης του χρήστη, ένα Select ερώτημα στη βάση δεδομένων και να εμφανίσει στο χρήστη τα είδη που πληρούν τα κριτήρια του.

```
cmdSelectEidos = "SELECT E.KodEidos, DescEidos, KodKatEidos, KodMonMetr,
OrioAnapar,A.AvailPosot FROM TblEidos E ,TblApothiki A WHERE
E.KodEidos=A.KodEidos AND"

If TxtDescS.Text <> "" Then
   Dim filter As String = TxtDescS.Text
   filter = filter.Replace("*", "%")
   cmdSelectEidos &= " DescEidos LIKE '" & filter & "' AND"
End If

If CmbxKatS.SelectedValue <> Nothing Then
   cmdSelectEidos &= " KodKatEidos='" & CmbxKatS.SelectedValue & "' AND"
End If

If CmbxMonMetrS.SelectedValue <> Nothing Then
   cmdSelectEidos &= " KodMonMetr='" & CmbxMonMetrS.SelectedValue & "'
AND"
End If

If TxtLimitFromS.Text <> "" Then
   cmdSelectEidos &= " OrioAnapar >= " & TxtLimitFromS.Text & " AND"
End If

If TxtLimitToS.Text <> "" Then
    cmdSelectEidos &= " OrioAnapar < " & TxtLimitToS.Text & " AND"
End If

If ChBoxSintagi.Checked Then
   cmdSelectEidos &= " KodEidos IN (SELECT KodParagEidos FROM
TblSintages)"
 End If


 If cmdSelectEidos.EndsWith("AND") Then
    cmdSelectEidos = cmdSelectEidos.Remove(cmdSelectEidos.Length - 3, 3)
End If
```

Στη συνέχεια διαβάζονται με ένα «Data Reader» οι εγγραφές που επέστρεψε το ερώτημα στη βάση δεδομένων. Οι εγγραφές αυτές εμφανίζονται μέσα στο Data Grid View ώστε ο χρήστης να τις διαχειριστεί όπως εκείνος θέλει.

3.7 Φόρμα «Καρτέλα Είδους»

Η καρτέλα είδους είναι μια φόρμα μέσα στην οποία εμφανίζονται αναλυτικά τα χαρακτηριστικά ενός είδους, και για όσα είδη διαθέτουν συνταγή παραγωγής εμφανίζεται, και η συνταγή παραγωγής τους. Πρόσβαση σε αυτή την φόρμα υπάρχει κυρίως μέσω της φόρμας «Διαχείριση Ειδών» και μέσω της επιλογής «Νέο Είδος». Η φόρμα αυτή εμφανίζεται όταν ο χρήστης θέλει να δημιουργήσει ένα νέο είδος ή όταν θέλει απλά να δει την καρτέλα ενός είδους.

Η φόρμα αυτή εμφανίζει τα στοιχεία ενός είδους αλλά αν ο χρήστης θέλει να δει στοιχεία και από τα άλλα είδη τότε δεν χρειάζεται να κλείνει και να ανοίγει την φόρμα αυτή. Μπορεί να πλοηγηθεί ανάμεσα σε όλα τα είδη που υπάρχουν στον πίνακα των ειδών με την βοήθεια των πλήκτρων πλοήγησης που βρίσκονται στο πάνω μέρος της φόρμας. Για να μπορέσει να υλοποιηθεί η λειτουργία της πλοήγησης τα διάφορα είδη από τη βάση δεδομένων αποθηκεύονται μέσα σε ένα Data Table, το οποίο γεμίζει με δεδομένα με τη βοήθεια της μεθόδου «GetEidi». Η μέθοδος αυτή παίρνει σαν όρισμα ένα Select ερώτημα με την μορφή string το οποίο εκτελεί στη βάση για να επιστρέψει κάθε φορά τα δεδομένα που θέλει ο χρήστης.

```
Public Sub GetEidi(ByVal cmdSelect As String)
  Dim cmd As SqlCommand
  Dim objDataAdapter As SqlDataAdapter

  Try
     cmd = New SqlCommand(cmdSelect, Conn)
     objDataAdapter = New SqlDataAdapter
     objDataAdapter.SelectCommand = cmd
     DataTableEidi = New DataTable
     objDataAdapter.Fill(DataTableEidi)

     objDataAdapter.Dispose()
     objDataAdapter = Nothing
     cmd.Dispose()
     cmd = Nothing

  Catch ex As Exception
```

```
      MessageBox.Show(ex.Message, "Σφάλμα Βάσης", MessageBoxButtons.OK)
   End Try
End Sub
```

Τα κουμπιά πλοήγησης αυτό που κάνουν είναι να μετακινούνται στις θέσεις του πίνακα και να εμφανίζουν τα δεδομένα που βρίσκονται στην κάθε θέση. Επιπλέον με τη βοήθεια της μεθόδου «CheckForRecipe()» ελέγχουν αν το είδος μιας συγκεκριμένης θέσης έχει συνταγή παραγωγής ώστε να την εμφανίσουν.

```
If currentRow < DataTableEidi.Rows.Count - 1 Then
   currentRow += 1
   GetDetails()
   ToolStripTxtCurrent.Text = currentRow + 1
   CheckForRecipe()
End If


Private Sub GetDetails()
   Try
     TxtKodEidos.Text = DataTableEidi.Rows(currentRow).Item("KodEidos")
     TxtDescEidos.Text =
DataTableEidi.Rows(currentRow).Item("DescEidos")
     CmbxKatigoria.SelectedValue =
DataTableEidi.Rows(currentRow).Item("KodKatEidos")
     CmbxMonMetr.SelectedValue =
DataTableEidi.Rows(currentRow).Item("KodMonMetr")
     TxtOrioAnap.Text =
DataTableEidi.Rows(currentRow).Item("OrioAnapar")
     TxtDiathPosot.Text =
DataTableEidi.Rows(currentRow).Item("AvailPosot")

     KodParagEidos = TxtKodEidos.Text
     DescChanged = False
     KodChanged = False
     KatigoriaChanged = False
     changedRecord = False
   Catch ex As Exception
      MessageBox.Show(ex.Message)
   End Try
End Sub
```

Όταν ο χρήστης θέλει να καταχωρήσει νέα εγγραφή μέσα από αυτή την φόρμα μπορεί να το κάνει αφού πατήσει το κουμπί «+». Αυτόματα αδειάζουν όλα τα κελιά από τις τιμές που υπήρχαν προηγουμένως και περιμένουν από τον χρήστη να δώσει τις δικές του τιμές. Πολύ σημαντικό κομμάτι είναι όταν ο χρήστης επιλέγει κατηγορία για το είδος. Στο σημείο αυτό τρέχει ένα κομμάτι κώδικα το οποίο, ανάλογα με το αν η κατηγορία που επιλέγηκε μπορεί να δεχτεί συνταγή παραγωγής, εμφανίζει το πλαίσιο για τις συνταγές παραγωγής ή το κρύβει. Διαμόρφωση ανάλογα με την κατηγορία δέχεται και το πεδίο για τον όριο αναπαραγγελίας το οποία γίνεται Read Only και αδειάζει αν η κατηγορία μπορεί

να δεχτεί συνταγή και επιτρέπει στον χρήστη να γράψει σε αυτό αν η κατηγορία δεν μπορεί να δεχτεί συνταγή παραγωγής. Ο λόγος που γίνεται αυτό είναι γιατί δεν μπορεί να οριστεί όριο αναπαραγγελίας σε ένα προϊόν το οποίο παράγεται.

```
If HaveProductionRecipe(CmbxKatigoria.SelectedValue.ToString) Then
   GroupBoxSintagi.Visible = True
   TxtOrioAnap.ReadOnly = True
   TxtOrioAnap.Text = ""
   BtnClose.Location = New Point(217, 607)
   Me.Size = New Size(545, 615)
Else
  GroupBoxSintagi.Visible = False
  TxtOrioAnap.ReadOnly = False
  BtnClose.Location = New Point(217, 275)
  Me.Size = New Size(545, 300)
End If
```

Αφού ο χρήστης περάσει όλα τα δεδομένα που χρειάζονται για το καινούργιο είδος που θέλει να καταχωρήσει, ή τελειώσει τις αλλαγές για ήδη υπάρχον είδος πρέπει να πατήσει το πλήκτρο της αποθήκευσης για να καταχωρηθούν οι αλλαγές τους στη βάση δεδομένων. Η μέθοδος της αποθήκευσης ελέγχει για διαθεσιμότητα του κωδικού του είδους και της περιγραφή. Επιπλέον γίνεται έλεγχος και για την κατηγορία του είδους, όταν αυτή αλλαχθεί, ακλουθώντας την λογική που έχει η μέθοδος της αποθήκευσης στη φόρμα «Διαχείριση Ειδών».

Εκτός από εισαγωγή και ενημέρωση ο χρήστη έχει την δυνατότητα να κάνει και διαγραφή κάποιου είδους. Για να μπορέσει να γίνει η διαγραφή πρέπει πρώτα ο χρήστης να απαντήσει θετικά στην ερώτηση αν όντως θέλει να διαγράψει το συγκεκριμένο είδος και ακολούθως να περάσει ο κώδικας από τον έλεγχο για το αν το είδος που θέλει να διαγραφεί χρησιμοποιείται σε κάποια εντολή ή συνταγή παραγωγής. Όταν περάσει από το έλεγχο αυτό και βεβαιωθεί η μέθοδος ότι το είδος δεν χρησιμοποιείται πουθενά τότε καλείται η μέθοδος για την διαγραφή του. Η διαγραφή ενός είδους δεν σημαίνει μόνο της διαγραφή του από τον πίνακα των ειδών. Το είδος πρέπει να διαγραφεί και από την αποθήκη καθώς επίσης πρέπει να διαγραφεί και η συνταγή παραγωγής του αν αυτή υπάρχει.

Όταν ο χρήστης δημιουργήσει ένα καινούργιο είδος το οποία θα ανήκει σε κάποια κατηγορία η οποία δέχεται συνταγή παραγωγής, για να μπορέσει να καταχωρήσει συνταγή παραγωγής για το είδος αυτό, θα πρέπει πρώτα να το αποθηκεύσει ώστε να ενεργοποιηθούν οι λειτουργίες της διαχείρισης της συνταγής παραγωγής του. Για τα είδη που είναι καταχωρημένα στη βάση

δεδομένων οι λειτουργίες διαχείρισης της συνταγής παραγωγής είναι πάντα ενεργοποιημένες.

Σαν λειτουργίες διαχείρισης της συνταγής εννοείται η καταχώρηση κάποιου συστατικού, η τροποποίηση ήδη υπάρχοντος και η διαγραφή του. Τα συστατικά που είναι διαθέσιμα για καταχώρηση σε μια συνταγή παραγωγής είναι όλα τα καταχωρημένα είδη εκτός από το τρέχον είδος (το είδος του οποίου διαχειρίζεται η συνταγή παραγωγής).

```
Private Sub LoadEidi()
   Dim cmd As SqlCommand
   Dim objDataAdapter As SqlDataAdapter
   Dim objDataTable As DataTable

   Dim cmdSelect As String = "SELECT KodEidos, DescEidos FROM TblEidos
WHERE KodEidos <>'" & KodParagEidos & "'"
   Try
       cmd = New SqlCommand(cmdSelect, Conn)
       objDataAdapter = New SqlDataAdapter
       objDataAdapter.SelectCommand = cmd
       objDataTable = New DataTable
       objDataAdapter.Fill(objDataTable)

       ColumnEidos.DataSource = objDataTable
       ColumnEidos.DisplayMember = "DescEidos"
       ColumnEidos.ValueMember = "KodEidos"

       objDataAdapter.Dispose()
       objDataAdapter = Nothing
       cmd.Dispose()
       cmd = Nothing

   Catch ex As Exception
       MessageBox.Show(ex.Message, "Σφάλμα Βάσης", MessageBoxButtons.OK)
   End Try

End Sub
```

Κάθε φορά που θα πατηθεί το κουμπί της αποθήκευσης για καταχώρηση νέου συστατικού ή για τροποποίηση ενός παλαιού γίνεται ένας έλεγχος αν το συστατικό που θέλει να καταχωρηθεί υπάρχει ήδη σαν συστατικό μέσα στη συγκεκριμένη συνταγή παραγωγής. Ο συγκεκριμένος έλεγχος γίνεται με τη βοήθεια της μεθόδου «CheckSistatiko» η οποία με την σειρά της καλεί την ομώνυμη Stored Procedure που βρίσκεται στον SQL Server.

```
Private Function CheckSistatiko(ByVal KodEidos As String) As Short
   Try
      Dim ReturnValue As String

      Dim cmdProc As New SqlCommand("CheckSistatiko", Conn)
```

```
        cmdProc.CommandType = CommandType.StoredProcedure
        cmdProc.Parameters.AddWithValue("@KodParag", KodParagEidos)
        cmdProc.Parameters.AddWithValue("@KodEidos", KodEidos)
        cmdProc.Parameters.Add("@RetVal", SqlDbType.SmallInt).Direction =
ParameterDirection.Output
        cmdProc.ExecuteNonQuery()

        ReturnValue = cmdProc.Parameters("@RetVal").Value

        Return ReturnValue

    Catch ex As Exception
       MessageBox.Show(ex.Message)
    End Try

End Function
```

3.8 Φόρμα «Διαχείρισης Εντολών Παραγωγής»

Για να παραχθεί ένα προϊόν από την παραγωγή θα πρέπει πρώτα να έχει δημιουργηθεί, να έχει εκτυπωθεί και να έχει σταλεί στην παραγωγή, μια εντολή παραγωγής. Στην παραγωγή μπορεί να υπάρχουν διάφορες εντολές παραγωγής οι οποίες μπορεί να βρίσκονται σε διαφορετικές καταστάσεις η κάθε μια. Οι διάφορες εντολές παραγωγής που υπάρχουν στην βάση δεδομένων μπορούν να διαχειριστούν από τον χρήστη μέσω της φόρμας «Διαχείρισης Εντολών Παραγωγής».

Στη φόρμα αυτή χωρίζονται οι τρέχουσες εντολές παραγωγής από τις εντολές που έχουν ολοκληρωθεί ή έχουν ακυρωθεί. Αυτό βοηθάει τον χρήστη να έχει καλύτερη εποπτεία για τις διάφορες εντολές παραγωγής. Για να μπορέσει να γίνει ο διαχωρισμός αυτός η φόρμα χρησιμοποιεί τις γενικές παραμέτρους. Αν ο χρήστης δεν έχει δώσει τιμές στις γενικές παραμέτρους της εφαρμογής τότε δεν μπορεί να γίνει ο διαχωρισμός των εντολών παραγωγής. Ο διαχωρισμός γίνεται με την βοήθεια δύο Data Grid View και ενός Tab Control με δύο σελίδες, σε κάθε μια από τις οποίες υπάρχει και ένα Data Grid View.

```
While rdr.Read()
     Try
        If rdr.Item("KodKatast").ToString = DefaultKatastDoneEntolis Or
rdr.Item("KodKatast").ToString = DefaultKatastCancelEntolis Then

            DGVDoneCancelEntoles.Rows.Add()
            Dim c As Integer
            For c = 0 To GridViewColumns - 1
```

```
                Dim timi As String = rdr.Item(c).ToString
                DGVDoneCancelEntoles.Item(c,
DGVDoneCancelEntoles.Rows.Count - 1).Value = timi
            Next
        Else
            DGVCurrentEntoles.Rows.Add()
            Dim c As Integer
            For c = 0 To GridViewColumns - 1
                Dim timi As String = rdr.Item(c).ToString
                DGVCurrentEntoles.Item(c, DGVCurrentEntoles.Rows.Count
- 1).Value = timi
            Next
        End If
    Catch ex As Exception
        MsgBox(ex.Message)
    End Try
End While
```

Οι τρέχουσες εντολές παραγωγής μπορούν να τροποποιηθούν μόνο αν δεν έχουν εκτυπωθεί, ενώ οι ολοκληρωμένες ή ακυρωμένες εντολές παραγωγής δεν μπορούν να δεχτούν καμία τροποποίηση. Η τροποποίηση μιας εντολής παραγωγής γίνεται με την βοήθεια της φόρμα «Εντολή Παραγωγής». Για να ανοίξει η φόρμα αυτή πρέπει να επιλεγεί μια εντολή παραγωγής και να πατηθεί το κουμπί «επεξεργασία» από το Tool Strip.

Μια εντολή παραγωγής εκτός από το να τροποποιηθεί μπορεί να ακυρωθεί. Με το πάτημα του κουμπιού για ακύρωση της επιλεγμένης εντολής παραγωγής ζητείται από τον χρήστη να γίνει επιβεβαιώσει. Αφού επιβεβαιώσει την ακύρωση τότε γίνεται ένας έλεγχος αν η εντολή παραγωγής έχει εκτυπωθεί. Αν δεν έχει εκτυπωθεί τότε θεωρούμε ότι δεν έχει πάει στην παραγωγή και ότι δεν δεσμεύτηκαν κάποιες ύλες για αυτήν, οπότε το μόνο που γίνεται είναι να οριστεί η εντολή σαν ακυρωμένη. Σε διαφορετική περίπτωση, αν η εντολή έχει εκτυπωθεί, ελευθερώνονται οι ποσότητες των υλών που δεσμεύτηκαν για την εντολή αυτή και στη συνέχεια ορίζεται η κατάσταση της σαν ακυρωμένη.

```
If DGVCurrentEntoles.SelectedRows(0).Cells(ColumnIsPrinted.Index).Value
= 1 Then
        ReleaseRecipeMaterials()
End If

Dim ArEntolis As Integer =
DGVCurrentEntoles.SelectedRows(0).Cells(ColumnArEntolis.Index).Value
SetStateEntolisAsCancel(ArEntolis)
```

Η μέθοδος «ReleaseRecipeMaterials()» παίρνει ένα - ένα τα προϊόντα που υπάρχουν στην εντολή παραγωγής και με βάση την συνταγή παραγωγής τους προσθέτει στην αποθήκη των αριθμό των υλών που δεσμεύτηκαν για τα προϊόντα αυτά.

```
cmdUpdate = "UPDATE TblApothiki SET AvailPosot= AvailPosot + " & Posot &
",LockPosot=LockPosot- " & Posot & " WHERE KodEidos='" & KodEidos & "'"
```

Για να δεσμευτούν οι ποσότητες των πρώτων υλών για κάθε εντολή παραγωγής, όπως έχει αναφερθεί και παραπάνω, θα πρέπει η εντολή παραγωγής να εκτυπωθεί. Κατά την εκτύπωση της εντολής παραγωγής, αν η εντολή δεν εκτυπωθεί ξανά, δεσμεύονται οι ποσότητες των πρώτων υλών που χρειάζονται για την εντολή και ενημερώνεται το πεδίο το οποίο δηλώνει ότι μια εντολή παραγωγής έχει εκτυπωθεί.

```
If DGVCurrentEntoles.SelectedRows(0).Cells(ColumnIsPrinted.Index).Value
= 0 Then

      ReportIsPrinted(DGVCurrentEntoles.SelectedRows(0).Cells(ColumnArEn
      tolis.Index).Value)
      LockRecipeMaterials()
End If
```

Η μέθοδος «LockRecipeMaterials» εκτελεί τις ίδιες ενέργειες με την «ReleaseRecipeMaterials» ώστε να βρει τα συστατικά που χρειάζονται για την εντολή παραγωγής και να τα δεσμεύσει. Η δέσμευση για κάθε συστατικό γίνεται με την εκτέλεση της παρακάτω εντολής στη βάση δεδομένων.

```
Dim cmdUpdate As String
cmdUpdate = "UPDATE TblApothiki SET AvailPosot= AvailPosot - " & Posot &
",LockPosot=LockPosot+ " & Posot & " WHERE KodEidos='" & KodEidos & "'"
```

Για κάθε εντολή η οποία ολοκληρώνεται, τα συστατικά της ,τα οποία δεσμεύτηκαν αφαιρούνται από την αποθήκη και από τις δεσμευμένες ποσότητες

```
For j As Short = 0 To eidiCount - 1
    kodEidos = DGVEidiEntolis.Item(ColumnEidos.Index, i).Value
    PosotEidos = DGVEidiEntolis.Item(ColumnPosotita.Index, i).Value
    FindSistatika_Remove(kodEidos, PosotEidos)
Next
```

Με την μέθοδο «FindSistatika_Remove» αφαιρούνται οι δεσμευμένες ποσότητες των υλών που χρειάστηκαν για να ολοκληρωθεί η εντολή παραγωγής.

```
Dim cmdUpdate As String
```

```
cmdUpdate = "UPDATE TblApothiki SET LockPosot=LockPosot- " & Posot & "
WHERE KodEidos='" & KodEidos & "'"
```

Η μόνη αλλαγή που μπορεί να γίνει στις εντολές παραγωγής μέσω αυτής της φόρμα είναι η κατάσταση της. Τα υπόλοιπα στοιχεία της για να αλλαχθούν πρέπει η εντολή να ανοίξει μέσα στη φόρμα «Εντολή Παραγωγής». Η κατάσταση της εντολής παραγωγής μπορεί να αλλαχθεί μόνο αν αυτή είναι εκτυπωμένη, αλλιώς δεν μπορεί να αλλαχθεί. Αυτό γίνεται για να αποφευχθούν λάθη ή απροσεξίες από τον χρήστη, ο οποίος μπορεί να ορίσει μια εντολή παραγωγής σαν ολοκληρωμένη ενώ αυτή δεν έχει εκτυπωθεί ακόμα.

```
Private Sub DGVCurrentEntoles_CellBeginEdit(ByVal sender As Object,
ByVal e As System.Windows.Forms.DataGridViewCellCancelEventArgs) Handles
DGVCurrentEntoles.CellBeginEdit

 If DataLoaded Then
     If DGVCurrentEntoles.CurrentCell.ColumnIndex = ColumnKatast.Index
Then
          Dim Row As Integer = DGVCurrentEntoles.CurrentCell.RowIndex
          If DGVCurrentEntoles.Item(ColumnIsPrinted.Index, Row).Value =
0 Then
              e.Cancel = True
          End If
      End If
  End If
End Sub
```

Στη φόρμα αυτή ο χρήστης μπορεί να παρακολουθήσει και να διαχειριστεί τις συνταγές παραγωγής. Για να γίνει η παρακολούθηση πιο εύκολη, η φόρμα διαθέτει και μια σύνθετη αναζήτηση. Η αναζήτηση μπορεί να γίνει είτε στις τρέχουσες εντολές παραγωγής είτε στις ολοκληρωμένες/ακυρωμένες, ανάλογα με το ποία καρτέλα έχει ανοικτή ο χρήστης.

3.9 Φόρμα «Εντολή Παραγωγής»

Όταν ο χρήστης θέλει να δημιουργήσει μια νέα εντολή παραγωγής ή να τροποποιήσει μια ήδη υπάρχουσα, όταν αυτή φυσικά δεν έχει εκτυπωθεί, θα πρέπει να χρησιμοποιήσει αυτή τη φόρμα.

Ανάλογα με το τι θέλει να κάνει ο χρήστης καλούνται και οι κατάλληλες μέθοδοι όταν ανοίγει η φόρμα αυτή. Αν ο χρήστης θέλει να δημιουργήσει μια καινούργια

εντολή παραγωγής, με την βοήθεια της μεθόδου «NeaEntoliParagogis» καλείται στον SQL Server η συνώνυμη Stored Procedures ώστε να δημιουργηθεί μια νέα εγγραφή στον πίνακα των εντολών παραγωγής.

```
Dim cmdProc As New SqlCommand("NeaEntoliParagogis", Conn)
cmdProc.CommandType = CommandType.StoredProcedure
cmdProc.Parameters.Add("@ArEntolis", SqlDbType.Int).Direction =
ParameterDirection.Output
cmdProc.ExecuteNonQuery()

ArEntolis = cmdProc.Parameters("@ArEntolis").Value

TxtEntoliNum.Text = ArEntolis
NewRecord = True
DataLoaded = True

If DefaultKatastNeasEntolis = "None" Then
     CmbxKatast.SelectedValue = 0
Else
     CmbxKatast.SelectedValue = DefaultKatastNeasEntolis
End If
```

Σε μια νέα εντολή παραγωγής ο χρήστης θα πρέπει να τοποθετήσει και κάποια προϊόντα τα οποία θέλει να δημιουργηθούν μαζί με τις ποσότητες τους. Τα προϊόντα που μπορεί να επιλέξει για να βάλει σε μια εντολή παραγωγής είναι μόνο τα προϊόντα τα οποία έχουν συνταγή παραγωγής. Δηλαδή αν κάποιο προϊόν ανήκει σε κάποια κατηγορία η οποία δέχεται συνταγή παραγωγής ενώ δεν υπάρχει συνταγή παραγωγής για αυτό δε βρίσκεται στη λίστα με τις επιλογές που μπορεί να έχει ο χρήστης.

```
Dim cmdSelect As String = "SELECT KodEidos, DescEidos FROM TblEidos
WHERE KodEidos IN (SELECT KodParagEidos From TblSintages)"
```

Τα προϊόντα που έχει καταχωρήσει ο χρήστης στην εντολή παραγωγής, μπορεί να τα ελέγξει όσον αφορά την διαθεσιμότητα των συστατικών τους. Ο έλεγχος της διαθεσιμότητας μπορεί να γίνει σε όλα τα προϊόντα ή σε ένα συγκεκριμένο, είτε τα προϊόντα αυτά είναι καταχωρημένα στη βάση δεδομένων είτε όχι. Η διαθεσιμότητα των συστατικών εμφανίζεται αναλυτικά στη φόρμα «Διαθεσιμότητα», η οποία δημιουργείται δυναμικά ανάλογα με τα προϊόντα και τα συστατικά τους.

Για να ελέχθη η διαθεσιμότητα των ειδών η μέθοδος παίρνει ένα – ένα τα προϊόντα που βρίσκονται στο Data Grid View και κρατά τον κωδικό τους και την ποσότητα που ζητάει ο χρήστης. Εμφανίζει για το κάθε ένα τα στοιχεία του και

στη συνέχεια δημιουργεί δυναμικά Data Grid View με μόνο μια γραμμή για κάθε συστατικό τους. Στο Data Grid View κάθε συστατικού εμφανίζεται και η συνολική ποσότητα του συστατικού που χρειάζεται για να παραχθεί η ποσότητα του προϊόντος που θέλει ο χρήστης. Η συνολική ποσότητα αυτή έχει πράσινο χρώμα αν στην αποθήκη υπάρχουν διαθέσιμα τα συστατικά αυτά και κόκκινο χρώμα αν δεν υπάρχουν διαθέσιμα.

Η μέθοδος η οποία δίνει τα συστατικά του κάθε προϊόντος είναι αναδρομική. Αν κάποιο από τα συστατικά του προϊόντος διαθέτει και αυτό συνταγή παραγωγής τότε ξανακαλείται η μέθοδος για να πάρει τα συστατικά του.

```
Private Sub ElegxosDiath(ByVal KodEidos As String, ByVal LocX As
Integer, ByVal LocY As Integer, ByVal PosotEidos As Integer, ByVal
EidosPanel As Panel, ByVal EidosDGV As DataGridView)

  If KodEidos = Nothing Then
     Return
  End If

  Dim Sistatika As Object(,) = SistatikaEidous(KodEidos)

  For i As Short = 0 To Sistatika.GetUpperBound(1)

     If CheckIfEidosHasRecipe(Sistatika(0, i)) = 1 Then
        EidosDetails(Sistatika(0, i), LocX, LocY, PosotEidos *
Sistatika(1, i), True, False, EidosPanel, EidosDGV)
        LocX = LocX + 20
        LocY = LocY + 22
        ElegxosDiath(Sistatika(0, i), LocX, LocY, PosotEidos *
Sistatika(1, i), EidosPanel, EidosDGV)

     Else
         EidosDetails(Sistatika(0, i), LocX, LocY, PosotEidos *
Sistatika(1, i), True, True, EidosPanel, EidosDGV)
          LocY = LocY + 22
     End If
   Next
End Sub
```

Η δυναμική δημιουργία της φόρμας γίνεται στη μέθοδο «EidosDetails» , στην οποία στέλνονται αρκετά ορίσματα ώστε να μπορέσει να διαχειριστεί και να καταλάβει αν αυτό που θα δημιουργήσει είναι Data Grid View που θα περιέχει τα στοιχεία του προϊόντος ή των συστατικών.

```
Private Function EidosDetails(ByVal KodEidos As String, ByVal LocX As
Integer, ByVal LocY As Integer, ByVal PosotEidos As Integer, ByVal
Sistatika As Boolean, ByVal CheckDiath As Boolean, ByVal EidosPanel As
Panel, ByVal EidosDGV As DataGridView) As Boolean

    Dim Values(5) As Object
```

```
    Values(0) = KodEidos

    Dim cmdSelect As String
    cmdSelect = "SELECT
E.KodEidos,DescEidos,KodKatEidos,KodMonMetr,A.AvailPosot,A.LockPosot
FROM TblEidos E ,TblApothiki A WHERE E.KodEidos=A.KodEidos AND
E.KodEidos='" & KodEidos & "'"

    Dim cmd As SqlCommand = New SqlCommand(cmdSelect, Conn)
    Dim rdr As SqlDataReader = Nothing

    Dim Katigoria As String
     Dim MonMetrisis As String
     Dim AvailPosot As Integer

     Try
       Try
          rdr = cmd.ExecuteReader
          rdr.Read()
          Values(1) = rdr.Item("DescEidos")
          Katigoria = rdr.Item("KodKatEidos")
          MonMetrisis = rdr.Item("KodMonMetr")
          AvailPosot = rdr.Item("AvailPosot")
          rdr.Close()
          rdr = Nothing

       Catch ex As Exception

       End Try


       Try
          cmdSelect = "SELECT DescKatEidos FROM TblKatEidos WHERE
KodKatEidos='" & Katigoria & "'"
          cmd = New SqlCommand(cmdSelect, Conn)
          rdr = cmd.ExecuteReader
          rdr.Read()
          Values(2) = rdr.Item("DescKatEidos").ToString
          rdr.Close()
          rdr = Nothing
       Catch ex As Exception

       End Try


       Try
          cmdSelect = "SELECT DescMonMetr FROM TblMonMetr WHERE
KodMonMetr='" & MonMetrisis & "'"
          cmd = New SqlCommand(cmdSelect, Conn)
          rdr = cmd.ExecuteReader
          rdr.Read()
          Values(3) = rdr.Item("DescMonMetr").ToString
          rdr.Close()
          rdr = Nothing

       Catch ex As Exception

       End Try

        Values(4) = AvailPosot
```

```
        Values(5) = PosotEidos

        If Sistatika Then
            Dim DGV As DataGridView = New DataGridView
            Dim newColumn As DataGridViewColumn


            For Each clm As DataGridViewColumn In EidosDGV.Columns
                newColumn = New DataGridViewColumn
                newColumn = clm.Clone
                DGV.Columns.Add(newColumn)
            Next

            DGV.ColumnHeadersVisible = False

            EidosPanel.Controls.Add(DGV)
            DGV.Location = New Point(LocX, LocY)
            DGV.BackgroundColor = Color.White
            DGV.BorderStyle = BorderStyle.None
            DGV.CellBorderStyle = DataGridViewCellBorderStyle.None
            DGV.RowHeadersVisible = False
            DGV.AllowUserToAddRows = False
            DGV.ReadOnly = True
            DGV.Size = New Size(731, 20)

            DGV.Rows.Add(Values)

            If CheckDiath Then
                If AvailPosot >= PosotEidos Then

                  DGV.Columns(5).DefaultCellStyle.BackColor =
                  Color.Green
                  LockPosot(KodEidos, PosotEidos)
                  Return True
                Else
                  DGV.Columns(5).DefaultCellStyle.BackColor = Color.Red
                  LockPosot(KodEidos, PosotEidos)
                  Return False
                End If
           End If

        Else
            EidosDGV.Rows.Add(Values)
        End If

        Return True

     Catch ex As Exception
         MessageBox.Show(ex.Message)
     End Try
End Function
```

Το σημαντικό στη μέθοδο αυτή είναι να δεσμεύονται οι ποσότητες των υλών που χρειάζονται ώστε η διαθεσιμότητα που εμφανίζεται να είναι αντικειμενική. Η δέσμευση γίνεται με τον ίδιο τρόπου που δεσμεύονται οι ύλες όταν εκτυπωθεί η εντολή παραγωγής. Όταν ο χρήστης κλείσει τη φόρμα της διαθεσιμότητας οι ποσότητες που έχουν δεσμευτεί, αποδεσμεύονται.

Μια εντολή παραγωγής μπορεί να εκτυπωθεί και από αυτή την φόρμα. Όπως είχε αναφερθεί και στην φόρμα «Διαχείριση Εντολών Παραγωγής», όταν μια εντολή εκτυπωθεί δεν μπορεί να δεχτεί πλέον τροποποιήσεις, και οι ύλες που χρειάζονται για τα προϊόντα που είναι αποθηκευμένα σε αυτήν την εντολή παραγωγής δεσμεύονται.

Μέσω της φόρμας αυτής ο χρήστης έχει την δυνατότητα να δημιουργήσει καινούργια εντολή παραγωγής ή να διαγράψει μια καινούργια εντολή παραγωγής που δεν έχει εκτυπωθεί ακόμα. Η διαγραφή μιας εντολής παραγωγής φαίνεται παρακάτω.

```
Private Sub DeleteEntoli()
   Dim cmdDelete As String = "DELETE FROM TblEntoliParagogisD WHERE
ArEntolisParagogis=" & ArEntolis
   Dim cmd As SqlCommand
   Try
      cmd = New SqlCommand(cmdDelete, Conn)
      cmd.ExecuteNonQuery()
   Catch ex As Exception
       MsgBox(ex.Message)
        End Try

   cmdDelete = "DELETE FROM TblEntoliParagogisH WHERE ArEntolisParag=" &
ArEntolis
   Try
      cmd = New SqlCommand(cmdDelete, Conn)
      cmd.ExecuteNonQuery()
   Catch ex As Exception
      MsgBox(ex.Message)
   End Try
End Sub
```

Στη βάση δεδομένων δεν επιτρέπει η εφαρμογή να υπάρχουν εντολές παραγωγής χωρίς κάποιο προϊόν ή εντολές παραγωγής που δεν έχουν καταχωρημένα όλα τα στοιχεία τους. Στην περίπτωση που ο χρήστης προσπαθήσει να κλείσει την φόρμα χωρίς να έχει καταχωρήσει τα στοιχεία που απαιτούνται, εμφανίζεται ένα μήνυμα το οποίο ενημερώνει τον χρήστη ότι δεν μπορεί να υπάρξει η εντολή παραγωγής που έχει δημιουργήσει και τον ρωτάει αν θέλει να τη διαγράψει. Ανάλογα με την απόφαση του χρήστη εκτελούνται οι κατάλληλες εντολές.

Προτάσεις – Συμπεράσματα

Μια εφαρμογή έχει πάντα περιθώρια για να αναπτυχθεί περισσότερο και να εκτελεί περισσότερες λειτουργίες. Στην εφαρμογή αυτή θα μπορούσαν να προστεθούν πολλά πράγματα ώστε η λειτουργία της να ήταν πληρέστερη.

Μια πρώτη βελτίωση που θα μπορούσε να γίνει είναι να υπάρχει ένα πίνακας με τους πελάτες της επιχείρησης ώστε, στις εντολές παραγωγής, ο χρήστης να επιλέγει τον πελάτη από την λίστα και όχι να πληκτρολογεί το όνομα του. Ακόμα καλύτερη σκέψη για την συγκεκριμένη ενέργεια θα ήταν να υπάρχουν οι παραγγελίες μέσα σε ένα πίνακα μαζί με τους πελάτες που τις έθεσαν έτσι ώστε ο χρήστης της εφαρμογής, απλά καταχωρώντας τον αριθμό της παραγγελίας, να παίρνει αυτόματα από την εφαρμογή το όνομα του πελάτη.

Η φόρμα της παραλαβής των προϊόντων, έτσι όπως έχει δημιουργηθεί, είναι μια τυπική φόρμα που εξυπηρετεί την λογική της εφαρμογής. Μια τέτοια φόρμα θα έπρεπε να διαθέτει αριθμό τιμολογίου και προμηθευτή. Για κάθε τιμολόγιο θα μπορούσε να καταχωρούνται όλα τα προϊόντα που έχει προμηθευτεί η επιχείρηση καθώς και στοιχεία του τιμολογίου όπως είναι η ημερομηνία.

Η παρακολούθηση των εντολών παραγωγής θα μπορούσε να γίνει πιο σύνθετη. Για παράδειγμα ο χρήστης θα μπορούσε να έχει την δυνατότητα να παρακολουθήσει αναλυτικά όλες τις φάσεις παραγωγής ενός είδους.

Βελτιώσεις μπορούν να γίνουν και στις συνταγές παραγωγής του είδους. Ίσως σε κάποια συστατικά που έχουν μονάδα μέτρησης, για παράδειγμα το μέτρο, να καταχωρούσε ο χρήστης εκτός από την ποσότητα σε μέτρα που χρειάζεται και τις διαστάσεις του συστατικού.

Με μια πρώτη ματιά στην εφαρμογή, αυτές είναι κάποιες από τις βελτιώσεις που μπορούν να γίνουν. Αν η εφαρμογή χρησιμοποιηθεί θα μπορούσαμε πιο εύκολα να βρούμε επιπλέον βελτιώσεις, τις οποίες θα ζητούσαν οι χρήστες της για να κάνουν πιο εύκολη την εργασία τους.

Βιβλιογραφία

Halvorson M. (2008). Microsoft Visual Basic 2008 Βήμα Βήμα. Αθήνα: Κλειδάριθμος.

Wikipedia, the free encyclopedia (15 Απριλίου 2009). Microsoft Visual Studio – Wikipedia, the free encyclopedia. Ανακτήθηκε 14, Ιουλίου 2010, από http://en.wikipedia.org/wiki/Microsoft_Visual_Studio

Wikipedia, the free encyclopedia (20 Μαΐου 2008). .NET Framework – Wikipedia, the free encyclopedia. Ανακτήθηκε 14, Ιουλίου 2010, από http://en.wikipedia.org/wiki/.NET_Framework

Βικιπαίδεια, η ελεύθερη εγκυκλοπαίδεια (29 Απριλίου 2009). Visual Basic – Βικιπαίδεια. Ανακτήθηκε 15, Ιουλίου 2010, από http://el.wikipedia.org/wiki/Visual_Basic

Wikipedia, the free encyclopedia (11 Φεβρουαρίου 2008). Visual Basic – Wikipedia, the free encyclopedia. Ανακτήθηκε 15, Ιουλίου 2010, από http://en.wikipedia.org/wiki/Visual_Basic

Wikipedia the free encyclopedia (11 Απριλίου 2003). Visual Basic.NET – Wikipedia, the free encyclopedia. Ανακτήθηκε 15, Ιουλίου 2010, από http://en.wikipedia.org/wiki/Visual_Basic_.NET

Visual Basic 2008 Tutorial. Visual Basic 2008 Tutorial. Ανακτήθηκε 15, Ιουλίου 2010 από http://www.vbtutor.net/vb2008/vb2008tutor.html

Wikipedia the free encyclopedia (8 Σεπτεμβρίου 2008). Microsoft SQL Server – Wikipedia, the free encyclopedia. Ανακτήθηκε 17, Ιουλίου 2010, από http://en.wikipedia.org/wiki/Microsoft_SQL_Server

Wikipedia the free encyclopedia (12 Αυγούστου 2008). SQL Server Management Studio – Wikipedia the free encyclopedia. Ανακτήθηκε 17, Ιουλίου 2010, από http://en.wikipedia.org/wiki/SQL_Server_Management_Studio

Παράρτημα

Ενδεικτική Εντολή Παραγωγής

Εντολή Παραγωγής

Ημερομηνία: 20/10/2010

Αρ.Εντολής Παρ.:	1024	Πελάτης:	Φωτεινή Χαραλάμπους
Αρ.Παραγγελίας:	35928	Ημ. Καταχώρησης:	13/08/2010 18:09:08

Προϊόντα Εντολής Παραγωγής

Κωδικός Είδους	Περιγραφή	Μονάδα Μέτρησης	Ποσότητα
400	Ηχείο	Τεμάχια	20

Συστατικά Προϊόντων

Κωδ. Παραγ.Είδους	Κωδ. Συστατικού	Περιγραφή Συστατικού	Μον. Μέτρ.	Συν. Ποσότ.
400	100	Πλαστικό	Μέτρα	20
400	300	Καλώδιο	Μέτρα	80
400	500	Διακόπτης	Τεμάχια	20

Page 1 of 1

Ενδεικτική Κατάσταση Ελλείψεων

Κατάσταση Ελλείψεων

Ημερομηνία: 20/10/2010

Κωδικός Είδους	Περιγραφή Είδους	'Οριο Αναπαρ.	Διαθέσιμη Ποσότ.
100	Πλαστικό	200	-20
300	Καλώδιο	310	285
600	Πλήκτρο	1,000	800

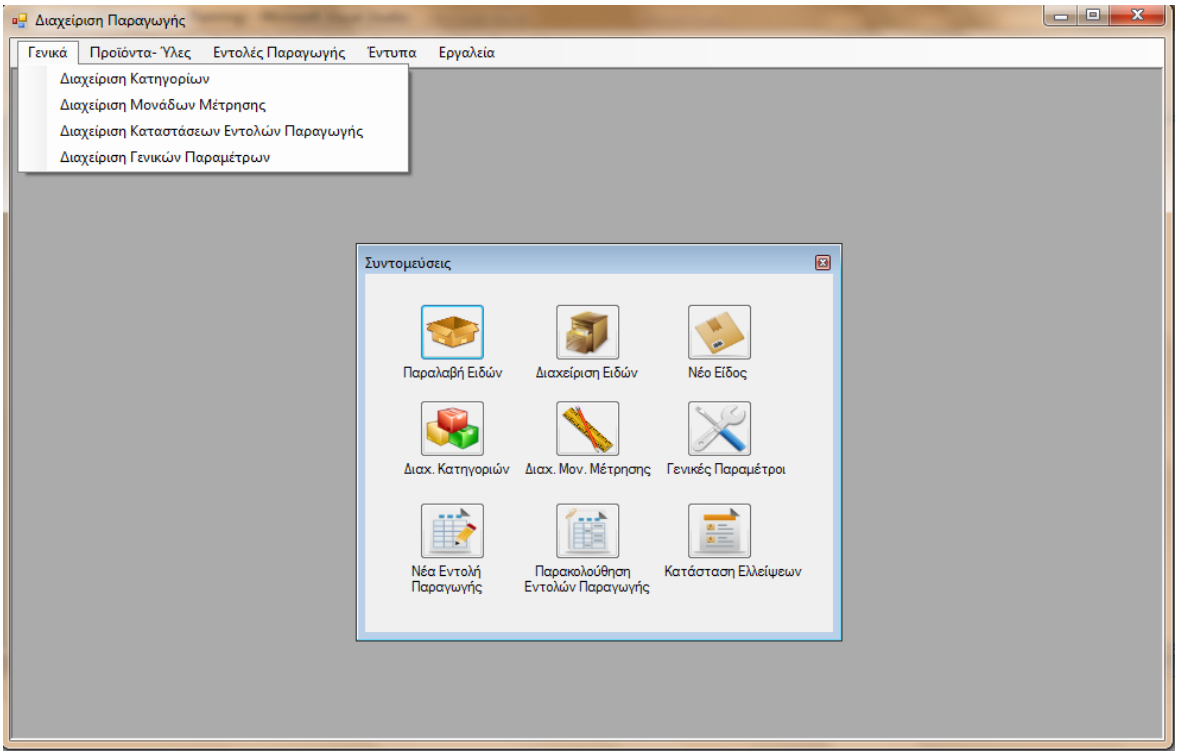
Page 1 of 1

Εφαρμογή Διαχείρισης Παραγωγής

Οδηγός Χρήσης

Η εφαρμογή αυτή είναι ένα εργαλείο που μπορεί να χρησιμοποιήσει μια βιομηχανία ή ένα εργοστάσιο για να μπορέσει να διαχειριστεί τις πρώτες ύλες του ώστε να παράγει διάφορα προϊόντα. Τα διάφορα είδη και οι πρώτες ύλες που διαθέτει η βιομηχανία διαθέτουν κάποια χαρακτηριστικά και ανήκουν σε κάποιες κατηγορίες. Για να παραχθεί ένα προϊόν πρέπει να σταλεί στην παραγωγή μια εντολή παραγωγής. Όλες αυτές οι λειτουργίες που χρειάζονται για την διαχείριση μιας παραγωγής μπορούν να γίνουν μέσω της εφαρμογής αυτής.

Ανοίγοντας την εφαρμογή βλέπετε την παρακάτω οθόνη, που είναι και η κύρια οθόνη μέσα στην οποία μπορούν να ανοίξουν τα υπόλοιπα παράθυρα της εφαρμογής.



Η οθόνη αυτή αποτελεί την κύρια οθόνη της εφαρμογής. Στο κέντρο της οθόνης υπάρχει ένα πλάσιο με συντομεύσεις. Στο πλάσιο αυτό υπάρχουν οι κύριες λειτουργίες της εφαρμογής. Οι λειτουργίες που υπάρχουν στις συντομεύσεις υπάρχουν και στο μενού της εφαρμογής, που βρίσκεται στο πάνω μέρος της οθόνης, μαζί με τις υπόλοιπες λειτουργίες της.

Μπορείτε να επιλέξετε κάποια λειτουργία είτε από το μενού, είτε από το πλαίσιο των συντομεύσεων. Το πλαίσιο των συντομεύσεων μπορείτε αν θέλετε να το κλείσετε, να το μετακινήσετε σε όποια θέση σας βολεύει και να το ανοίξετε από τα «Εργαλεία» στο κεντρικό μενού της εφαρμογής.

Μπάρα Εργαλείων

Οι περισσότερες από τις λειτουργίες που παρέχει η εφαρμογή είναι για τη διαχείριση των διαφόρων στοιχείων που την απαρτίζουν. Στα παράθυρα των λειτουργιών αυτών εμφανίζεται στο πάνω μέρος μια μπάρα εργαλείων. Η μπάρα αυτή είναι κοινή για το μεγαλύτερο μέρος των παραθύρων και μέσω αυτής ο χρήστης διαχειρίζεται τα δεδομένα του.

Το πρώτο μέρος των εργαλείων που υπάρχουν στη μπάρα είναι εργαλεία πλοήγησης. Με τα βελάκια μπορείτε να μεταφερθείτε ανάμεσα στις εγγραφές που υπάρχουν ή μπορείτε να γράψετε κατευθείαν τον αριθμό της γραμμή που θέλετε στο κουτάκι που υπάρχει και να πατήσετε το κουμπί «Enter» από το πληκτρολόγιο σας για να μεταφερθείτε στη γραμμή αυτή.

Στο δεύτερο μέρος των εργαλείων υπάρχουν τα κουμπιά για την διαχείριση των εγγραφών. Με το πρώτο κουμπί μπορείτε να δημιουργήσετε μια νέα εγγραφή στη βάση δεδομένων, το δεύτερο κουμπί είναι για την διαγραφή μιας ή περισσότερων εγγραφών και το τελευταίο για την αποθήκευση των αλλαγών που κάνατε στις εγγραφές και για την καταχώρηση της νέας εγγραφής που δημιουργήσατε.

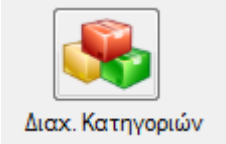
Το τρίτο μέρος της γραμμής εργαλείων περιέχει μια γρήγορη αναζήτηση στις εγγραφές που εμφανίζονται. Η αναζήτηση αυτή γίνεται με την περιγραφή των εγγραφών. Για να κάνετε μια αναζήτηση θα πρέπει να πληκτρολογήσετε αυτό που θέλετε στο κουτάκι δίπλα από το κουμπί της αναζήτησης και να πατήσετε

την αναζήτηση. Σε μια αναζήτηση μπορείτε να χρησιμοποιήσετε και τον χαρακτήρα «*» (αστεράκι). Ο χαρακτήρας αυτός δηλώνει οποιονδήποτε χαρακτήρα και μπορείτε να τον χρησιμοποιήσετε αν για παράδειγμα θέλετε τις εγγραφές που αρχίζουν από «Πα» και τελειώνουν σε οτιδήποτε άλλο. Η σύνταξη για αυτήν την αναζήτηση θα είναι «Πα*».

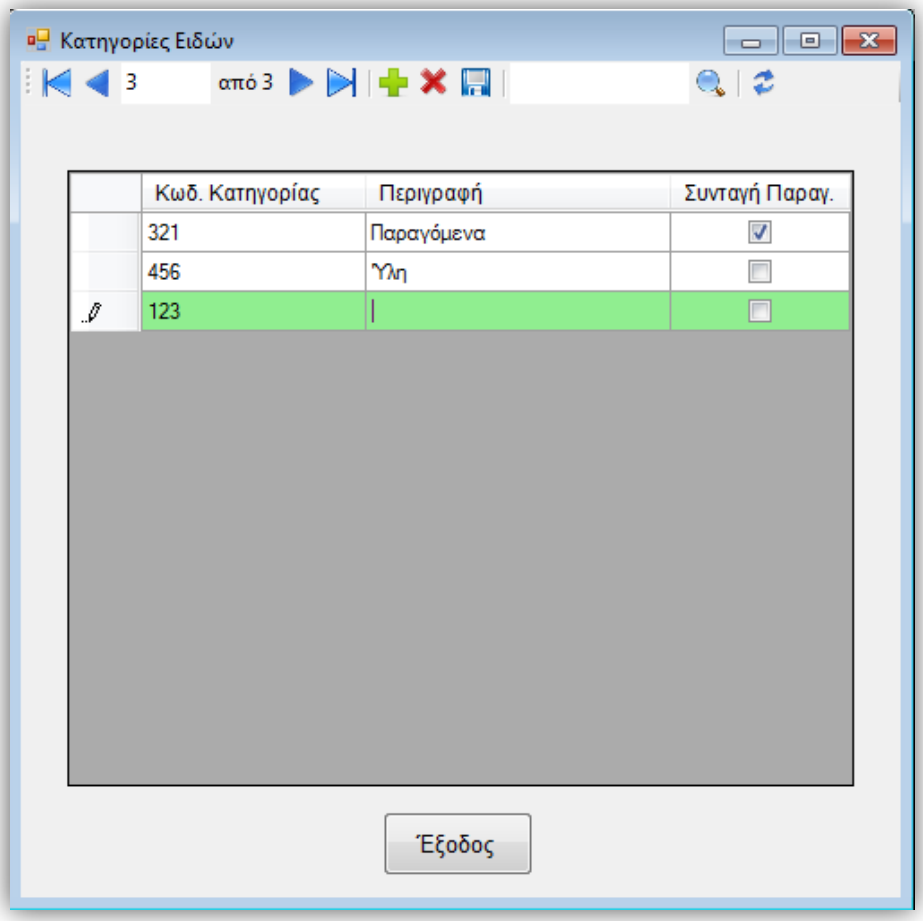
Στο τέταρτο και τελευταίο μέρος της γραμμής εργαλείων είναι η ανανέωση. Πατώντας το πλήκτρο αυτό ανανεώνονται οι εγγραφές που εμφανίζονται στο παράθυρο. Αυτό σας βοηθάει για να δείτε αν οι αλλαγές ή οι καινούργιες εγγραφές σας έχουν καταχωρηθεί.

Διαχείριση Κατηγοριών

Κάθε είδος που θα καταχωρήσετε μέσα στη βάση δεδομένων θα πρέπει να ανήκει σε κάποια κατηγορία. Οι κατηγορίες δεν είναι συγκεκριμένες αλλά πρέπει να τις δημιουργήσετε και να τις διαχειριστείτε ανάλογα με τα δεδομένα της βιομηχανίας σας. Για να δημιουργήσετε τις κατηγορίες στις οποίες θα ανήκουν τα είδη σας θα πρέπει να επιλέξετε από το μενού «Γενικά» την επιλογή «Διαχείριση Κατηγοριών». Η επιλογή μπορεί να γίνει και από το πλαίσιο των συντομεύσεων.



Το παράθυρο που θα ανοίξει έχει την παρακάτω μορφή. Η μορφή αυτή είναι και η μορφή που έχουν τα περισσότερα παράθυρα της εφαρμογής.

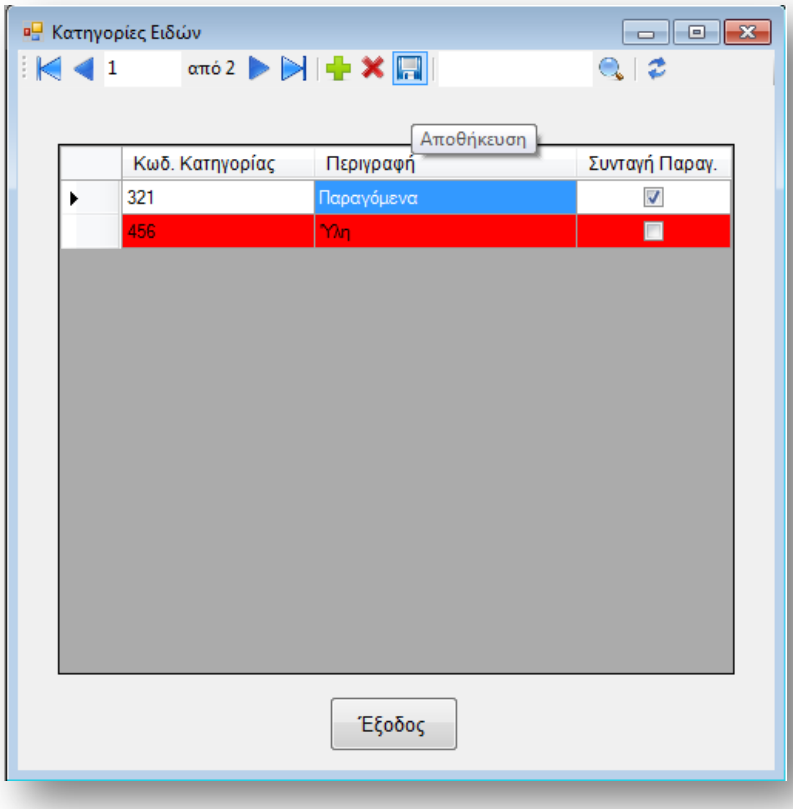


Θα παρατηρήσετε ότι στο πάνω μέρος του παραθύρου υπάρχει η μπάρα εργαλείων που περιγράφηκε προηγουμένως. Οι λειτουργίες της έχουν αναφερθούν αναλυτικά και δεν θα αναφερθούν ξανά.

Στο στιγμιότυπο που βλέπετε πιο πάνω έχει πατηθεί το κουμπί για δημιουργία νέας εγγραφής. Μόλις πατηθεί το κουμπί αυτό εμφανίζεται μια νέα γραμμή στο πίνακα η οποία έχει πράσινο χρώμα. Αυτό που πρέπει να κάνετε είναι να δώσετε ένα κωδικό για τη κατηγορία που θέλετε να δημιουργήστε και μια περιγραφή. Τα στοιχεία αυτά είναι απαραίτητα για την δημιουργία μιας εγγραφής. Το πεδίο «Συνταγή Παραγωγής» το επιλέγετε όταν τα προϊόντα που θα ανήκουν στην κατηγορία αυτή θα πρέπει να διαθέτουν συνταγή παραγωγής για να δημιουργηθούν. Όταν ολοκληρώσετε την εισαγωγή σας θα πρέπει να πατήσετε σε κάποια άλλη γραμμή ώστε να δηλώσετε ότι τελειώσατε (θα πρέπει να φύγει το μολυβάκι που υπάρχει δίπλα στη γραμμή του πίνακα). Αφού τελειώστε με την εισαγωγή σας θα πρέπει να πατήσετε το κουμπί «Αποθήκευση» από την μπάρα εργαλείων για να καταχωρηθεί ή εγγραφή στη βάση δεδομένων.

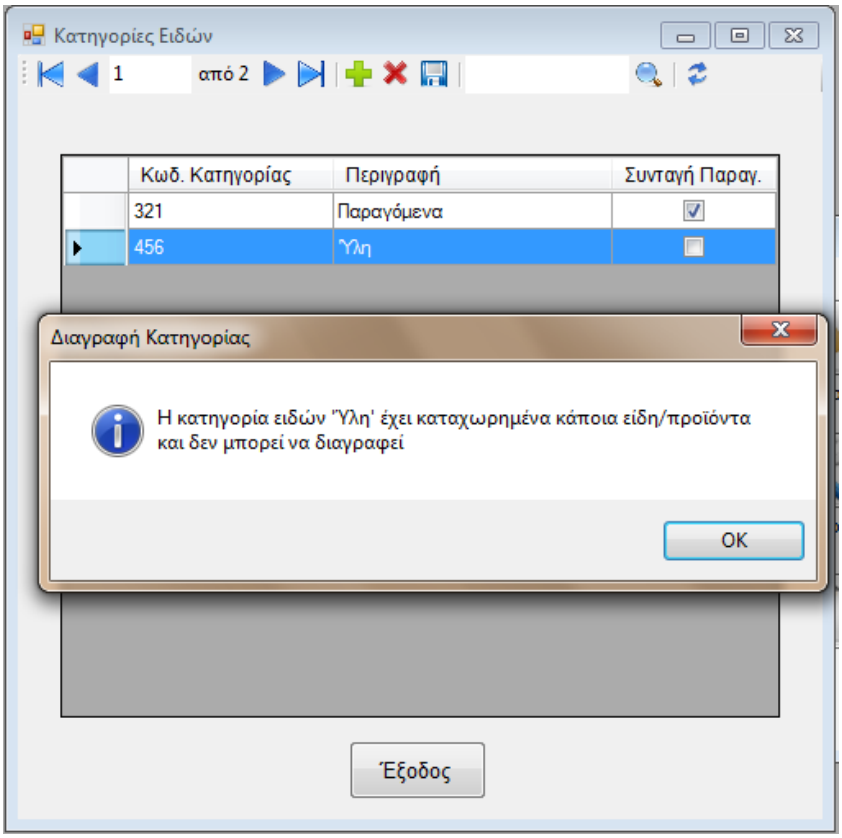
Να θυμάστε ότι δεν μπορείτε να καταχωρήσετε κάποια εγγραφή που να έχει τον ίδιο κωδικό ή την ίδια περιγραφή με κάποια άλλη εγγραφή. Κάτι τέτοιο θα δημιουργούσε πρόβλημα σύγχυσης στο τελικά ποια εγγραφή είναι ποια.

Αν σε κάποια από τις εγγραφές θελήσετε να κάνετε κάποια αλλαγή θα παρατηρήσετε ότι η γραμμή στην οποία κάνατε την αλλαγή απέκτησε κόκκινο χρώμα. Αυτό γίνεται για να μπορείτε να διαχωρίζεται τις καινούργιες γραμμές (έχουν πράσινο χρώμα) από τις γραμμές που είναι τροποποιημένες και από αυτές που δεν δέχτηκαν καμία αλλαγή. Όταν τελειώσετε με τις αλλαγές που θέλετε να κάνετε (να θυμάστε ότι θα πρέπει να φύγετε από το κελί του πίνακα που κάνατε την αλλαγή για να δηλώσετε ότι τελειώσετε με την τροποποίηση) θα πρέπει να πατήστε το πλήκτρο της «Αποθήκευσης» για να καταχωρήστε της αλλαγές σας στη βάση δεδομένων. Ένα στιγμιότυπο από τη διαδικασία της τροποποίησης μιας εγγραφής εμφανίζετε στη εικόνα που φαίνετε παρακάτω.



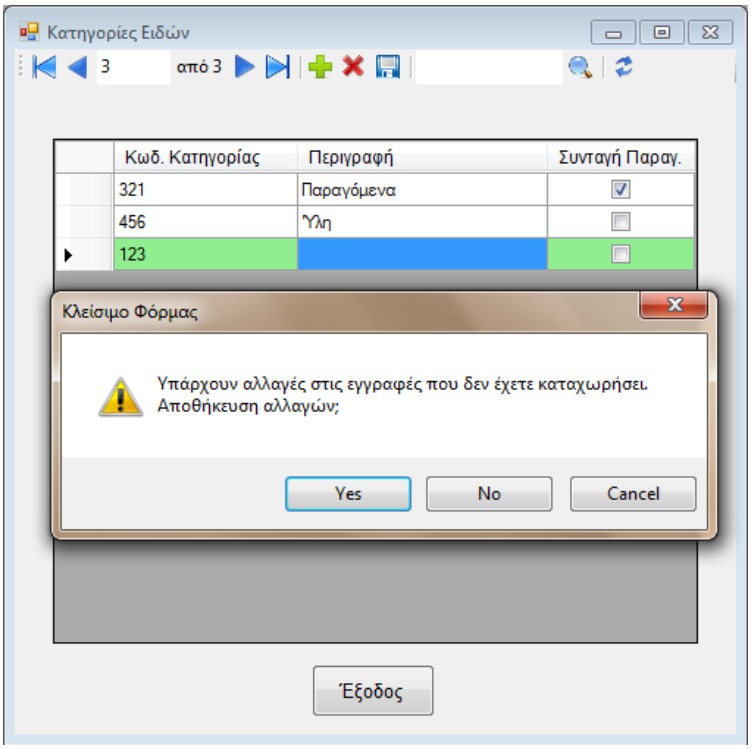
Εκτός από την δημιουργία και την τροποποίηση κάποιας εγγραφής έχετε το δικαίωμα να διαγράψετε και μια ή περισσότερες από αυτές. Για να διαγράψετε κάποια εγγραφή θα πρέπει να την επιλέξετε και να πατήσετε το κουμπί διαγραφή.

Τότε θα σας ζητηθεί μια επιβεβαίωση της διαγραφής και αφού την επιβεβαιώσετε θα συνεχίσει η εφαρμογή την διαδικασία της διαγραφής. Στο σημείο της διαγραφής θα ήταν σημαντικό να γνωρίζετε ότι δεν μπορείτε να διαγράψετε κάποια κατηγορία στην οποία ανήκουν κάποια από τα είδη ή τα προϊόντα σας. Αν για κάποιο λόγο προσπαθήσετε να διαγράψετε μια τέτοια κατηγορία θα λάβετε ένα μήνυμα που θα σας ενημερώνει ότι η εγγραφή που θέλετε δεν μπορεί να διαγραφεί. Για να διαγράψετε μια κατηγορία δεν θα πρέπει να ανήκουν σε αυτή κάποια είδη ή προϊόντα. Αν θέλετε οπωσδήποτε να τη διαγράψετε θα πρέπει να αλλάξετε κατηγορία σε όλα τα είδη που ανήκουν σε αυτήν (αυτό μπορεί να γίνει μέσα από το παράθυρο διαχείρισης των ειδών) και στη συνέχεια να προσπαθήσετε να την διαγράψετε. Το μήνυμα «απόρριψης» μιας αίτησης διαγραφής φαίνεται στην εικόνα που ακολουθεί.



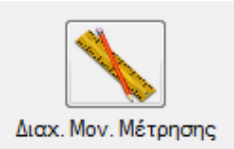
Αν θεωρήσουμε ότι σε κάποιο παράθυρο κάνατε κάποιες αλλαγές, ή δημιουργήσατε κάποιες εγγραφές και δεν τις έχετε αποθηκεύσει ενώ προσπαθήσετε να κλείσετε το παράθυρο, δεν θα χάσετε τις αλλαγές σας αν δεν το θέλετε. Η εφαρμογή μπορεί να καταλάβει αν έχετε κάνει κάποιες

τροποποιήσεις και όταν πάτε να κλείσετε το παράθυρο σας εμφανίζει ένα προειδοποιητικό μήνυμα στο οποίο σας λέει ότι στο παράθυρο υπάρχουν αλλαγές. Εκτός από το μήνυμα υπάρχει και μια ερώτηση για το αν θέλετε να αποθηκεύσετε τις αλλαγές σας ή όχι. Το μήνυμα φαίνετε στην εικόνα που ακολουθεί.

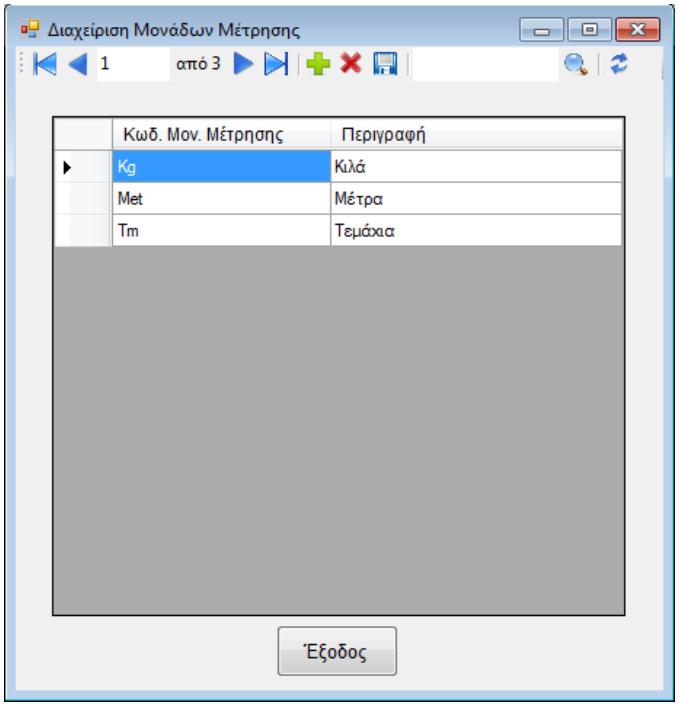


Διαχείριση Μονάδων Μέτρησης

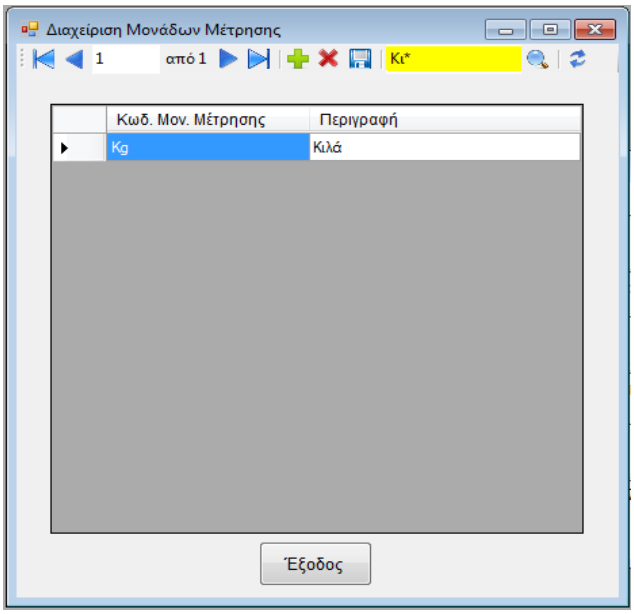
Ένα είδος διαθέτει κάποια χαρακτηριστικά, ένα από τα χαρακτηριστικά του είναι και η μονάδα μέτρησης τους. Τις μονάδες μέτρησης που θέλετε να υπάρχουν στην εφαρμογή θα πρέπει να τις δημιουργήσετε και να τις διαχειριστείτε ανάλογα με τα είδη και τα προϊόντα που διαχειρίζεστε και παράγετε. Για την διαχείριση των μονάδων μέτρησης υπάρχει συγκεκριμένο παράθυρο το οποίο μπορείτε να το ανοίξετε είτε από το κύριο μενού της εφαρμογής, πατώντας «Γενικά» και στη συνέχεια «Διαχείριση Μονάδων Μέτρησης», είτε πατώντας το αντίστοιχο κουμπί από το πλαίσιο των συντομεύσεων.



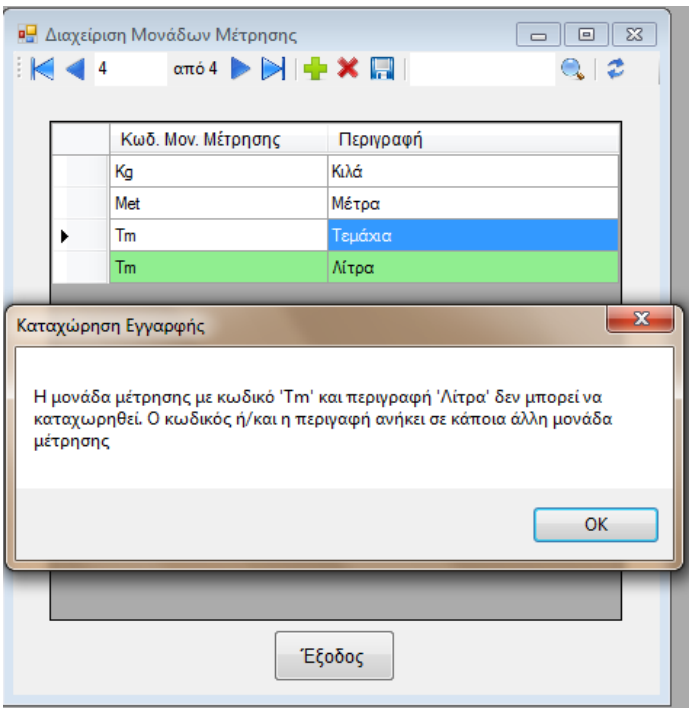
Ανοίγοντας το παράθυρο για την διαχείριση των μονάδων μέτρησης θα παρατηρήσετε ότι έχει την ίδια μορφή με αυτή που έχει το παράθυρο για την διαχείριση των κατηγοριών. Στο πάνω μέρος του περιέχει τη μπάρα εργαλείων, που η χρήση της δείχτηκε στην αρχή του οδηγού χρήσης, και στο υπόλοιπο μέρος εμφανίζονται οι διάφορες μονάδες μέτρησης που είναι αποθηκευμένες στην εφαρμογή.



Μέσω της μπάρας εργαλείων έχετε τη δυνατότητα να διαχειριστείτε τις μονάδες μέτρησης που θέλετε να υπάρχουν στην εφαρμογή σας. Όπως έχει ήδη αναφερθεί με τη βοήθεια της μπάρας εργαλείων μπορείτε να πλοηγηθείτε ανάμεσα στις εγγραφές που έχετε καταχωρημένες, να δημιουργήσετε, να επεξεργαστείτε ή να διαγράψετε κάποιες από αυτές και τέλος μπορείτε να κάνετε μια γρήγορη αναζήτηση ανάμεσα στις εγγραφές σας με βάση την περιγραφή που τους έχετε δώσει. Μην ξεχνάτε ότι στην αναζήτηση μπορείτε να χρησιμοποιήσετε και τον χαρακτήρα «*». Ένα παράδειγμα αναζήτησης φαίνεται στην εικόνα που ακολουθεί.



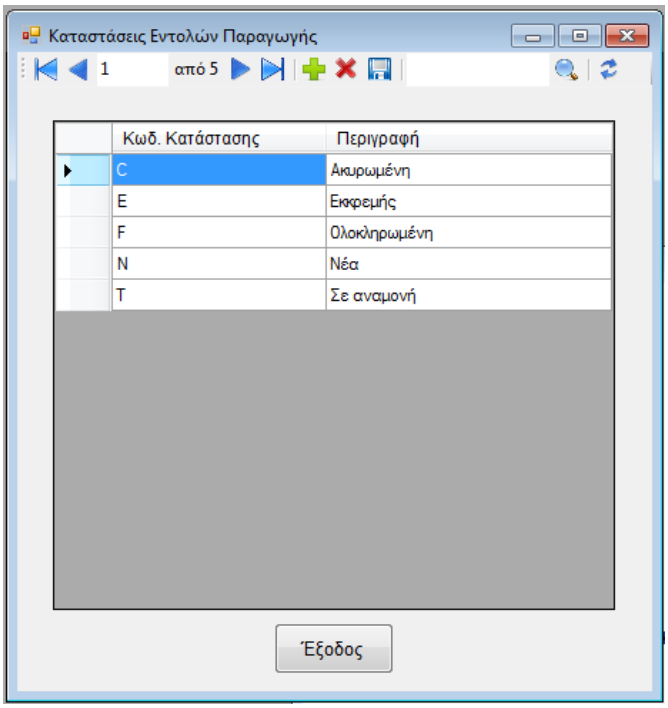
Κατά την δημιουργία ή τροποποίηση κάποιας εγγραφής θα πρέπει να προσέξετε ώστε ο κωδικός ή η περιγραφή που θα δώσετε να μην ανήκει σε κάποια άλλη εγγραφή. Αν από λάθος προσπαθήσετε να αποθηκεύσετε μια εγγραφή που έχει κωδικό ή/και περιγραφή από κάποια άλλη εγγραφή θα σας εμφανιστεί ένα μήνυμα που θα σας ενημερώνει ότι δεν μπορεί να γίνει η αποθήκευσης της εγγραφής που ζητάτε.



Διαχείριση Καταστάσεων Εντολής Παραγωγής

Ανάλογα με τον τρόπο που λειτουργεί το κάθε εργοστάσιο υπάρχουν διάφορες φάσεις και καταστάσεις στις οποίες μπορεί να βρίσκεται μια εντολή παραγωγής. Για να υπάρχει η ελευθερία στο κάθε εργοστάσιο να έχει τις δικές του καταστάσεις εντολών παραγωγής η εφαρμογή διαθέτει ένα παράθυρο στο οποίο διαχειρίζονται οι καταστάσεις αυτές.

Δεν υπάρχει συντόμευση προς το παράθυρο αυτό και για να το ανοίξετε θα πρέπει αν πάτε στο μενού «Γενικά» και να επιλέξετε «Διαχείριση Καταστάσεων Εντολών Παραγωγής». Το παράθυρο που θα ανοίξει έχει την παρακάτω μορφή.

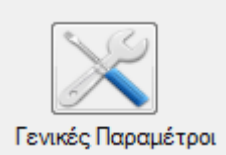


Όπως μπορείτε να παρατηρήσετε το παράθυρο αυτό διαθέτει τη μπάρα εργαλείων που διαθέτουν και τα περισσότερα παράθυρα και ένα πίνακα στον οποίο εμφανίζονται οι εγγραφές που έχετε δημιουργήσει. Στο παράθυρο αυτό μπορείτε να δημιουργήσετε δικές σας καταστάσεις, να τροποποιήσετε παλαιές εγγραφές ή να διαγράψετε κάποιες εγγραφές που δεν χρησιμοποιούνται σε κάποια εντολή παραγωγής. Επιπλέον υπάρχει η δυνατότητα για αναζήτηση με βάση την περιγραφή που έχετε δώσει στις καταστάσεις.

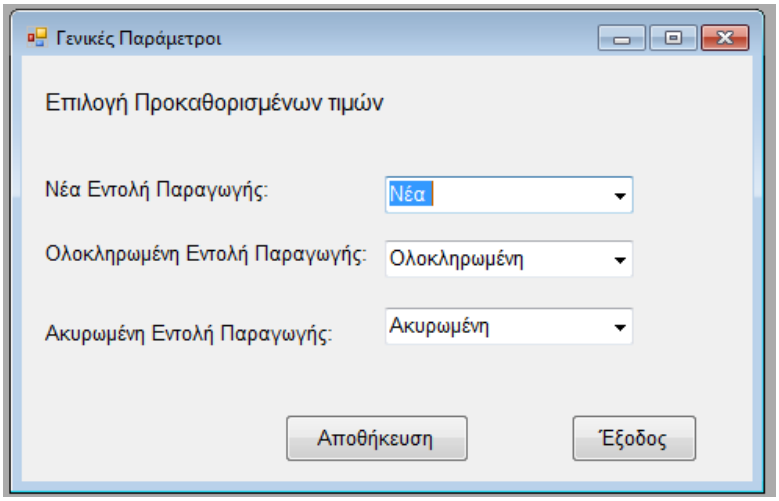
Ορισμός Γενικών Παραμέτρων

Η εφαρμογή δίνει στους χρήστες της την δυνατότητα να δημιουργήσουν τις δικές τους καταστάσεις στις οποίες θα μπορούν να βρεθούν οι εντολές παραγωγής. Για να μπορέσει όμως να λειτουργήσει η εφαρμογή καλύτερα θα πρέπει να ορίσετε ποια από τις καταστάσεις που έχετε δημιουργήσει θα έχει μια εντολή που μόλις έχει δημιουργηθεί, μια εντολή που έχει ολοκληρωθεί και μια ακυρωμένη εντολή.

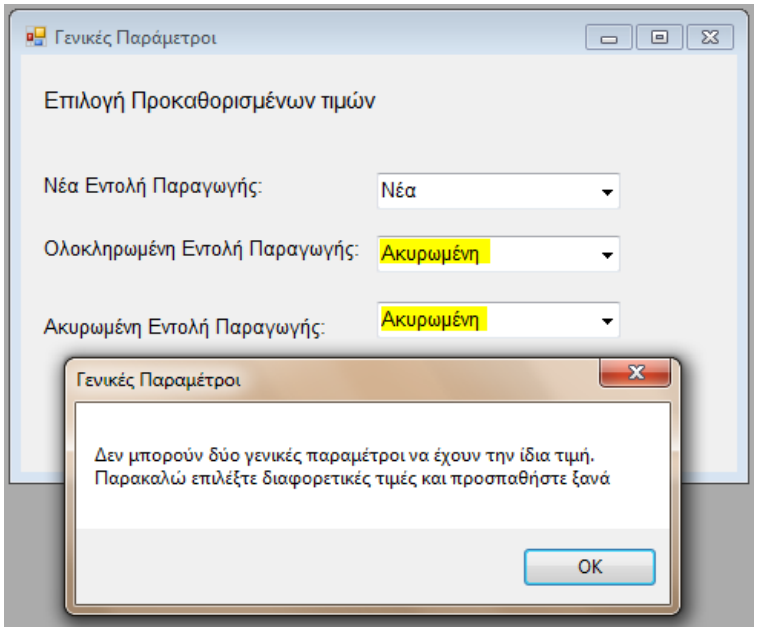
Για να ορίσετε τις καταστάσεις αυτές θα πρέπει να ανοίξετε το παράθυρο των «Γενικών Παραμέτρων». Το παράθυρο αυτό μπορείτε να το ανοίξετε από το μενού «Γενικά» και την επιλογή «Διαχείριση Γενικών Παραμέτρων» καθώς και από το πλαίσιο των συντομεύσεων.



Η φόρμα αυτή έχει τρείς λίστες στις οποίες περιέχει όλες τις καταστάσεις που έχετε δημιουργήσει (αυτές που εμφανίζονται στο παράθυρο «Διαχείριση Καταστάσεων Εντολών Παραγωγής»). Από την κάθε λίστα θα πρέπει να επιλέξετε την κατάσταση που θα δώσετε στην νέα, στην ολοκληρωμένη και την ακυρωμένη εντολή παραγωγής. Το παράθυρο έχει την μορφή που βλέπετε στην ακόλουθη εικόνα.



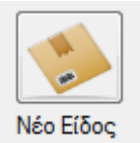
Όταν θα επιλέξετε τη κατάσταση που θα δώσετε σε κάθε εντολή παραγωγής θα πρέπει να πατήσετε το πλήκτρο «Αποθήκευση» για να αποθηκεύσετε τις επιλ φ ές σας στη βάση δεδομένων. Δεν είναι υποχρεωτικ ό να επιλ ξετε κατάσταση σε κάθε εντολή παραγωγής. Αν δεν θέλετε μπορείτε να μην επιλέξετε κατάσταση για καμία εντολή παραγωγής. Καλό θα ήταν όμως να το κάνετε για να μπορέσει η εφαρμογή να διαχειριστεί καλύτερα τις εντολές παραγωγής σας. Όταν πατήσετε το πλήκτρο αποθήκευσης θα πρέπει να ξέρετε ότι δεν μπορείτε να δώσετε ίδια τιμή σε δύο παραμέτρους. Αν για κάποιο λόγο το κάνετε αυτό θα εμφανιστεί στην οθόνη ένα μήνυμα που θα σας ενημερών α ότι η αποθήκ ευση που θέλετε δεν μπορεί να γίνει. Ένα τέτοιο στιγμιότυπο φαίνεται στην παρακάτω εικόνα.



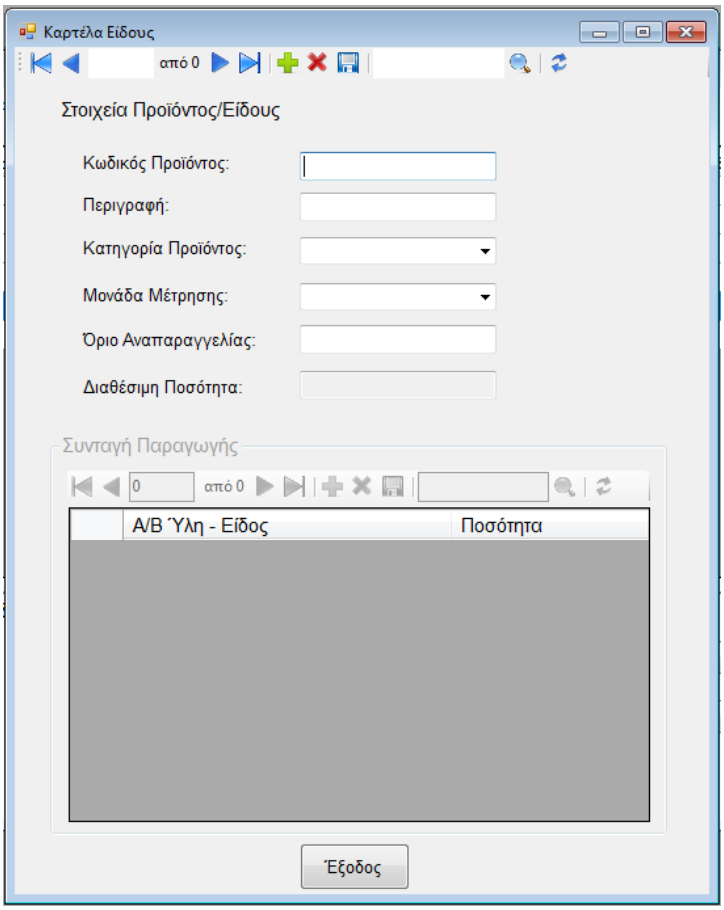
Δημιουργία Νέου Είδους/Προϊόντος

Σκοπός της εφαρμογής αυτής είναι να μπορέσει να διαχειρίζεται τα είδη που υπάρχουν στην αποθήκη ενός εργοστασίου και να δημιουργεί με βάση τις συνταγές παραγωγής διάφορα προϊόντα. Τα είδη που χρησιμοποιεί το εργοστάσιο και τα προϊόντα που παράγει δεν τα γνωρίζει η εφαρμογή για αυτό τον λόγο θα πρέπει να τα δημιουργήσετε και να τα καταχωρήσετε σε αυτήν.

Για να ανοίξετε την φόρμα δημιουργίας νέου είδους θα πρέπει να πατήσετε από το μενού «Προϊόντα - Ύλες» και στη συνέχεια «Δημιουργία Νέου Είδους» ή να πατήσετε από τις συντομεύσεις το κουμπί «Νέο Είδος».



Το παράθυρο που θα εμφανιστεί στην οθόνη σας αποτελεί μια καρτέλα είδους. Μέσα στην καρτέλα αυτή μπορείτε επίσης να ανοίξετε είδη και προϊόντα που ήδη είναι καταχωρημένα στη βάση δεδομένων της εφαρμογής.



Η «Καρτέλα Είδους» αποτελείται από δύο μέρη. Το πρώτο μέρος είναι για τα στοιχεία του είδους/προϊόντος και το δεύτερο μέρος είναι για την συνταγή παραγωγής του.

Στην καρτέλα αυτή θα πρέπει να καταχωρήσετε τα στοιχεία που σας ζητούνται για το νέο είδος που θέλετε να δημιουργήσετε. Για το νέο σας είδος θα πρέπει να επιλέξετε την κατηγορία στην οποία θα ανήκει και την μονάδα μέτρησης η οποία θα το χαρακτηρίζει. Θα παρατηρήσετε ότι όταν επιλέγετε μια κατηγορία, το πλαίσιο για την συνταγή παραγωγής του προϊόντος, κρύβεται όταν η κατηγορία αυτή δεν δέχεται συνταγή παραγωγής, ενώ εμφανίζεται όταν η κατηγορία που επιλέξετε δέχεται συνταγή παραγωγής. Επιπλέον θα παρατηρήσετε ότι το κουτάκι με το όριο αναπαραγγελίας δεν σας επιτρέπει να του δώσετε τιμή όταν η κατηγορία προϊόντος που επιλέξατε δέχεται συνταγή παραγωγής. Ο λόγος που γίνεται αυτό είναι γιατί τα προϊόντα που ανήκουν σε αυτή την κατηγορία δεν παραγγέλνονται αλλά παράγονται από την παραγωγή του εργοστασίου.

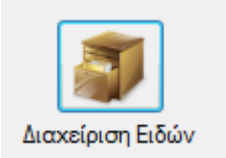
Αφού καταχωρήσετε τα στοιχεία του είδους, θα πρέπει να τα αποθηκεύσετε στη βάση δεδομένων. Για να τα αποθηκεύσετε θα πρέπει να πατήσετε το κουμπί «Αποθήκευση» από την γραμμή εργαλείων που βρίσκεται στο πάνω μέρος της οθόνης. Αν η εγγραφή σας καταχωρηθεί με επιτυχία τότε, αν το πλαίσιο της συνταγής παραγωγής δεν είναι κρυμμένο, θα το δείτε να ενεργοποιείται. Αυτό σημαίνει ότι μπορείτε να επιλέξετε τα είδη και την ποσότητα τους που χρειάζονται για να δημιουργηθεί το προϊόν σας. Την συνταγή παραγωγής του προϊόντος μπορείτε να την διαχειριστείτε όποτε θέλετε με την βοήθεια της μπάρας εργαλείων που βρίσκεται μέσα στο πλαίσιο της συνταγής παραγωγής.

Με την αποθήκευση των στοιχείων του είδους σας θα παρατηρήσετε ότι πλέον έχετε την δυνατότητα να χρησιμοποιήσετε την μπάρα εργαλείων στο πάνω μέρος της οθόνης για να δείτε και διαχειριστείτε και τα είδη που είναι καταχωρημένα στη βάση δεδομένων. Επιπλέον έχετε την δυνατότητα να δημιουργήσετε επιπλέον είδη αν θέλετε μέσα από την καρτέλα αυτή χωρίς να χρειάζεται να την κλείσετε και να την ανοίξετε πάλι από το μενού.

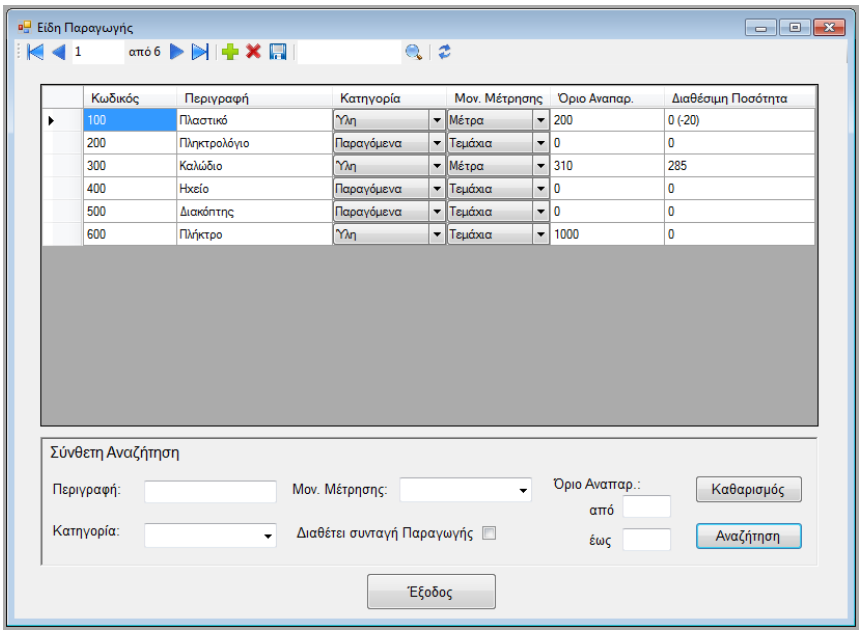
Διαχείριση Ειδών

Προηγουμένως είχε αναφερθεί το πώς μπορείτε να δημιουργήσετε ένα καινούργιο είδος στην εφαρμογή. Στο μέρος αυτό θα αναφερθεί το πώς μπορείτε

να διαχειριστείτε όλα τα είδη που υπάρχουν στην εφαρμογή σας. Για να ανοίξετε το παράθυρο διαχείρισης των ειδών σας θα πρέπει να επιλέξετε από το μενού «Προϊόντα - Ύλες» την επιλογή «Διαχείριση Ειδών» ή να πατήσετε από τις συντομεύσεις την αντίστοιχή επιλογή.



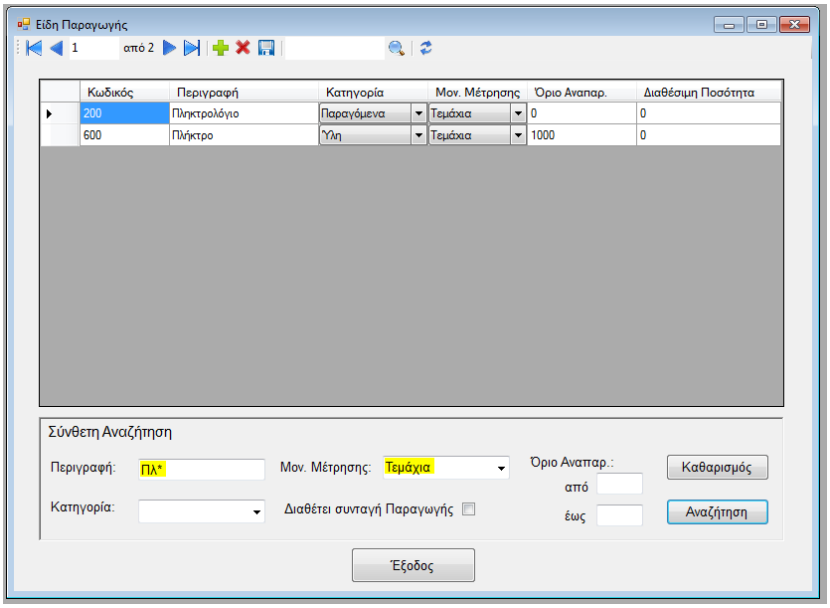
Μέσα στο παράθυρο αυτό μπορείτε να δείτε όλα τα είδη που υπάρχουν καταχωρημένα στη βάση δεδομένων της εφαρμογής.



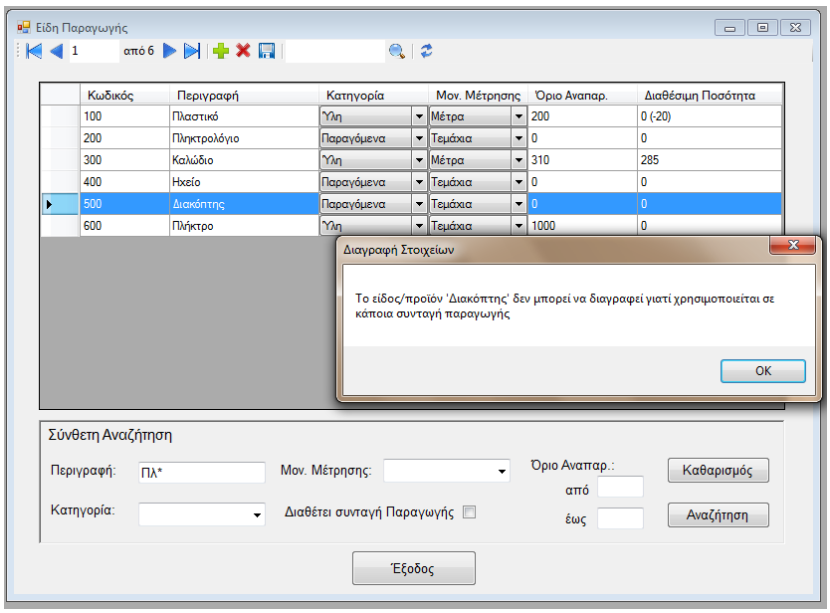
Στο πάνω μέρος του παραθύρου υπάρχει η μπάρα εργαλείων με την οποία έχετε την δυνατότητα να διαχειρίζεστε τα είδη που υπάρχουν στην βάση δεδομένων.

Επειδή τα ειδή που υπάρχουν σε έν α εργ στάσιο είναι συνήθως πολλά, η εφαρμογή διαθέτει και μια σύνθετη αναζήτηση ώστε να σας βοηθήσει να διαχωρίζετε τα είδη που θέλετε να εμφανίσετε. Το μόνο που έχετε να κάνετε για να την χρησιμοποιήσετε είναι να συμπληρώσετε τα πεδία για αναζήτηση που θέλετε και να πατήσετε το κουμπί της «Αναζήτησης». Αμέσως θα δείτε στον πίνακα να εμφανίζονται τα είδη που αντιστοιχούν στα πεδία της αναζήτησης που

έχετε θέσει. Αν θέλετε να εμφανίσετε πάλι όλα τα δεδομένα απλά θα πρέπει να πατήσετε το πλήκτρο «Ανανέωση» από την γραμμή εργαλείων.



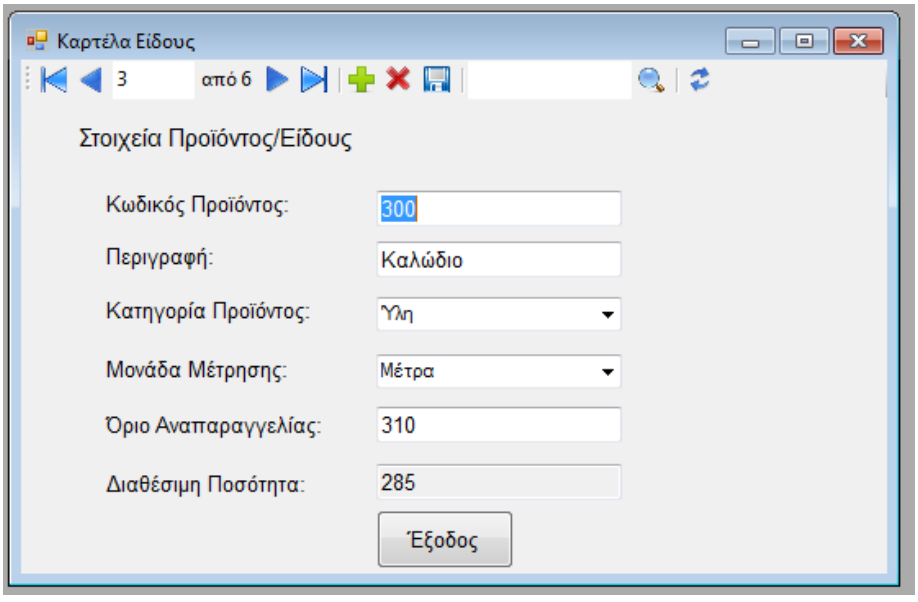
Αν κάποιο είδος από αυτά που εμφανίζονται δεν το χρησιμοποιείτε πλέον και θέλετε να το διαγράψετε μπορείτε να το κάνετε. Φτάνει μόνο να το επιλέξετε και να πατήσετε το κουμπί της διαγραφής. Για να διαγραφεί όμως ένα είδος δε θα πρέπει να χρησιμοποιείτε σε καμία εντολή ή συνταγή παραγωγής. Αν δοκιμάσετε να διαγράψετε ένα είδος που χρησιμοποιείται θα δείτε να σας εμφανίζεται ένα μήνυμα το οποία θα σας ενημερώνει ότι η διαγραφή δεν μπορεί να γίνει.



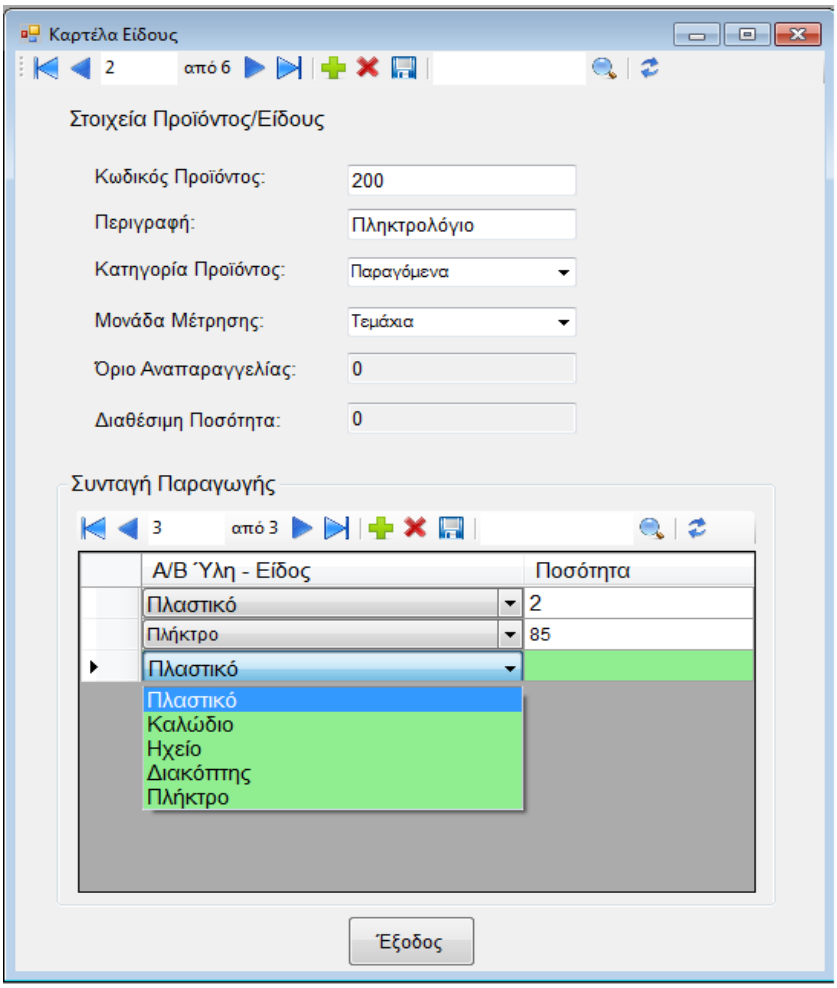
Μέσω της γραμμής εργαλείων που υπάρχει μπορείτε να δημιουργήσετε ένα νέο είδος. Η δημιουργία του νέου είδους μπορεί να γίνει αν πατήσετε το κουμπί «+». Πατώντας το πλήκτρο αυτό θα εμφανιστεί μια κενή καρτέλα είδους σαν αυτή που περιγράφηκε στη «Δημιουργία Νέου Είδους». Στη συνέχεια για την δημιουργία του είδους θα κάνετε τις ίδιες ενέργειες που περιγράφηκαν στην «Δημιουργία Νέου Είδους».

Για να τροποποιήσετε κάποια από τα χαρακτηριστικά των ειδών μπορείτε να το κάνετε απευθείας από την σελίδα αυτή ή να πατήσετε διπλό κλικ πάνω στη γραμμή του είδους που θέλετε να τροποποιήσετε ώστε να ανοίξει η καρτέλα του. Μέσω της καρτέλας του είδους μπορείτε να διαχειριστείτε και την συνταγή παραγωγής τους, αν έχει, πράγμα που δεν μπορείτε να κάνετε από το παράθυρο αυτό.

Η καρτέλα του κάθε είδους είναι διαφορετική ανάλογα με το αν η κατηγορία στην οποία ανήκουν μπορεί να δεχτεί συνταγή παραγωγής ή όχι. Αν η κατηγορία στην οποία ανήκει δεν δέχεται συνταγή παραγωγής τότε η καρτέλα είδους έχει αυτή την μορφή.



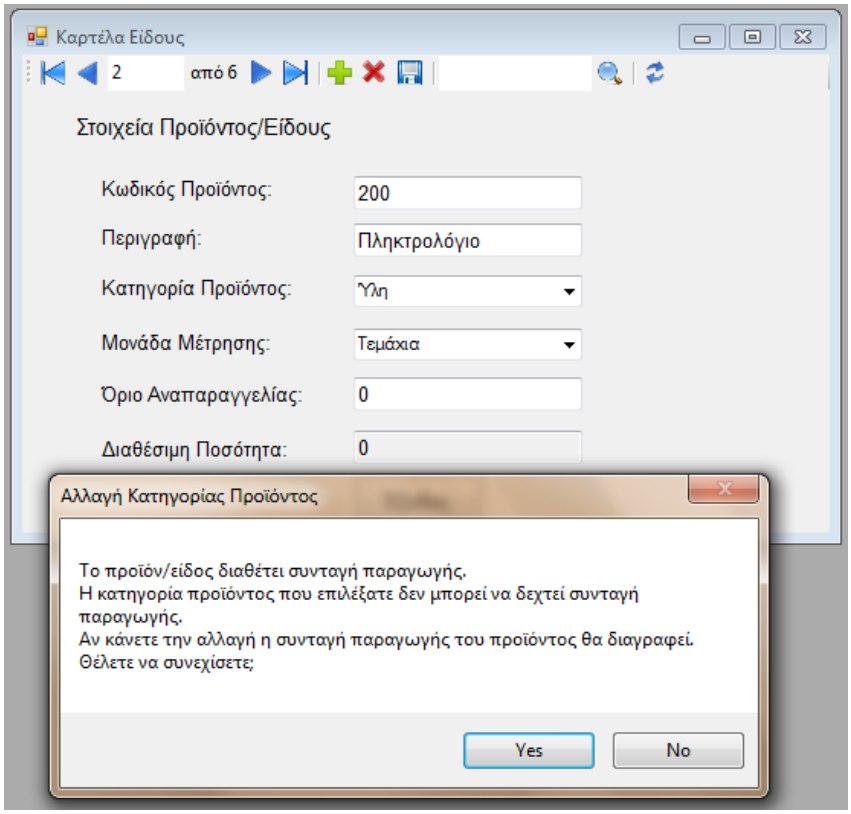
Αν η κατηγορία του προϊόντος δέχεται συνταγή παραγωγής τότε η καρτέλα είδους έχει άλλη μορφή και διαθέτει και το πλαίσιο για την συνταγή παραγωγής του είδους αυτού. Μια τέτοια καρτέλα είναι η ακόλουθη:



Θα παρατηρήσετε ότι στην συνταγή του είδους μπορείτε να επιλέξετε για συστατικά του όλα τα είδη που διαθέτει η αποθήκη σας εκτός από το τρέχον είδος, το είδος δηλαδή στο οποίο αντιστοιχεί η συνταγή παραγωγής. Κάτι άλλο που χρειάζεται να γνωρίζετε είναι ότι η εφαρμογή δεν σας επιτρέπει να καταχωρήσετε σε μια συνταγή παραγωγής το ίδιο είδος σαν συστατικό. Αν το κάνετε αυτό θα λάβετε ένα μήνυμα από την εφαρμογή που θα σας ενημερώνει ότι το συστατικό αυτό είναι ήδη καταχωρημένο και δεν μπορεί να καταχωρηθεί ξανά.

Εκτός από την συνταγή παραγωγής στην καρτέλα αυτή μπορείτε να διαχειριστείτε όλα τα είδη σας με την βοήθεια της γραμμής εργαλείων που υπάρχει στο πάνω μέρος της καρτέλας. Μπορείτε να τροποποιήσετε τον κωδικό και την περιγραφή του είδους αρκεί τα νέα δεδομένα που θα δώσετε να μην ανήκουν σε άλλο είδος. Επιπλέον μπορείτε να αλλάξετε την μονάδα μέτρησης του είδους καθώς και την κατηγορία του.

Όταν αλλάξετε την κατηγορία του είδους θα πρέπει να είσαστε προσεκτικοί. Αν το είδος διαθέτει συνταγή παραγωγής και εσείς επιλέξετε μια κατηγορία η οποία δεν μπορεί να δεχτεί συνταγή παραγωγής, θα εμφανιστεί στην οθόνη σας ένα μήνυμα όπως αυτό που εμφανίζεται στην εικόνα παρακάτω.



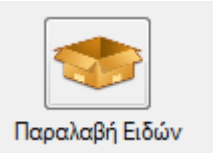
Το μήνυμα αυτό σας ενημερώνει ότι το είδος διαθέτει κάποια συνταγή παραγωγής και αν θέλετε να κάνετε την αλλαγή θα διαγραφεί η συνταγή παραγωγής του. Η απόφαση θα είναι πλέον δική σας.

Παραλαβή Ειδών

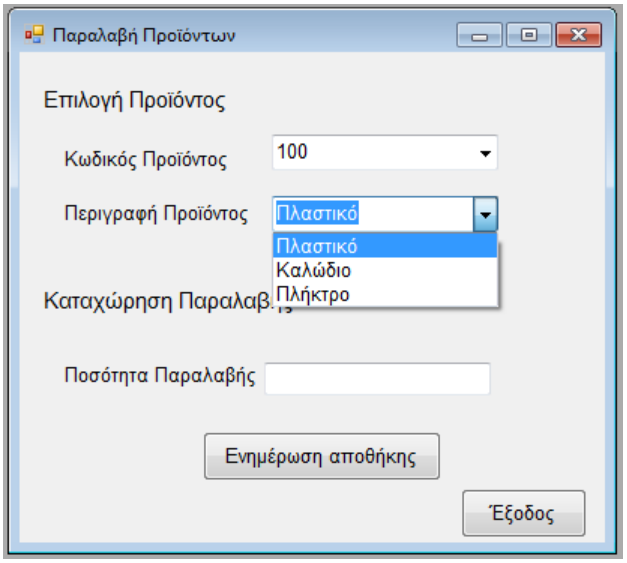
Όταν δημιουργήσετε ένα νέο είδος θα παρατηρήσετε ότι η διαθέσιμη ποσότητα του είναι μηδέν. Αν αυτό το είδος είναι παραγόμενο δε υπάρχει πρόβλημα αφού τα παραγόμενα είδη δημιουργούνται μετά από μια εντολή παραγωγής. Τα είδη όμως που δεν είναι παραγόμενα δεν μπορούν να δημιουργηθούν και το εργοστάσιο για να έχει διαθέσιμη ποσότητα θα πρέπει να τα αγοράσει. Όταν τα

είδη έρθουν στο εργοστάσιο θα πρέπει να τα καταχωρήσετε και στην αποθήκη της εφαρμογής μέσω της λειτουργίας της παραλαβής ειδών.

Την λειτουργία αυτή μπορείτε να την βρείτε αν πατήσετε στο μενού «Προϊόντα - Ύλες» και στην συνέχεια «Παραλαβή Ειδών» ή από αν πατήσετε την αντίστοιχη επιλογή από το πλαίσιο των συντομεύσεων.



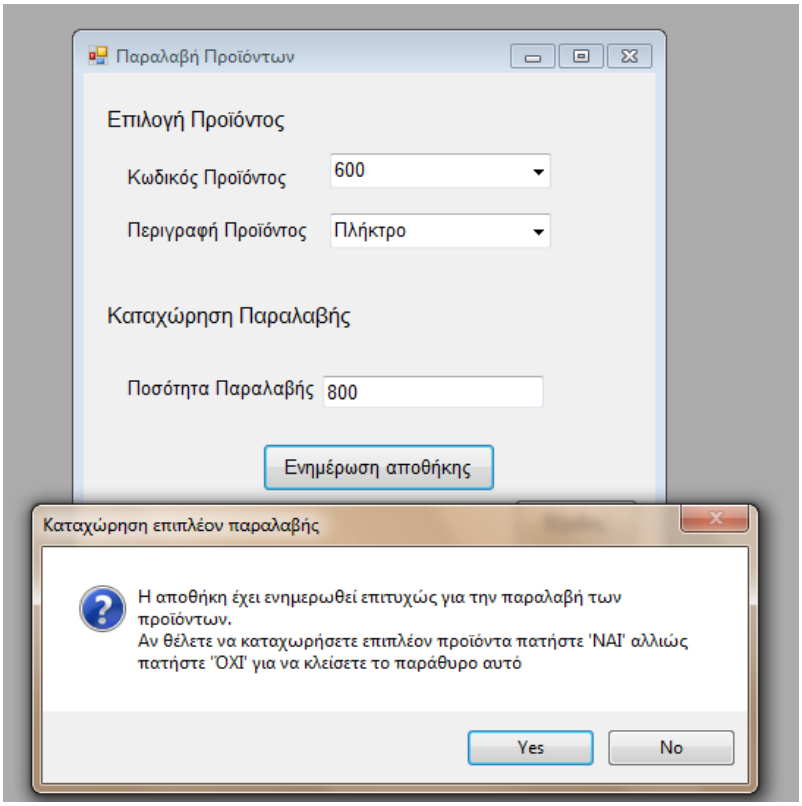
Στη φόρμα αυτή θα πρέπει να επιλέξετε το είδος που έχετε παραλάβει με βάση τον κωδικό του ή την περιγραφή του από μια λίστα με τα είδη που δεν είναι παραγόμενα. Αφού επιλέξετε το είδος που θέλετε θα πρέπει να καταχωρήσετε και την ποσότητα που έχετε παραλάβει από το είδος αυτό.



Αφού καταχωρήσετε και την ποσότητα του είδους, τότε για να εισαχθούν τα είδη και στη αποθήκη της εφαρμογής θα πρέπει να πατήσετε το κουμπί «Ενημέρωση αποθήκης». Αυτόματα η αποθήκη ενημερώνεται.

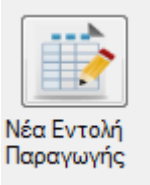
Αν θέλετε να παραλάβετε περισσότερα από ένα είδη δεν χρειάζεται να κλείσετε και να ανοίξετε ξανά το παράθυρο αυτό. Μετά το τέλος της ενημέρωσης της

αποθήκης εμφανίζεται ένα μήνυμα το οποίο σας ρωτάει αν θέλετε να καταχωρήσετε επιπλέον προϊόντα.

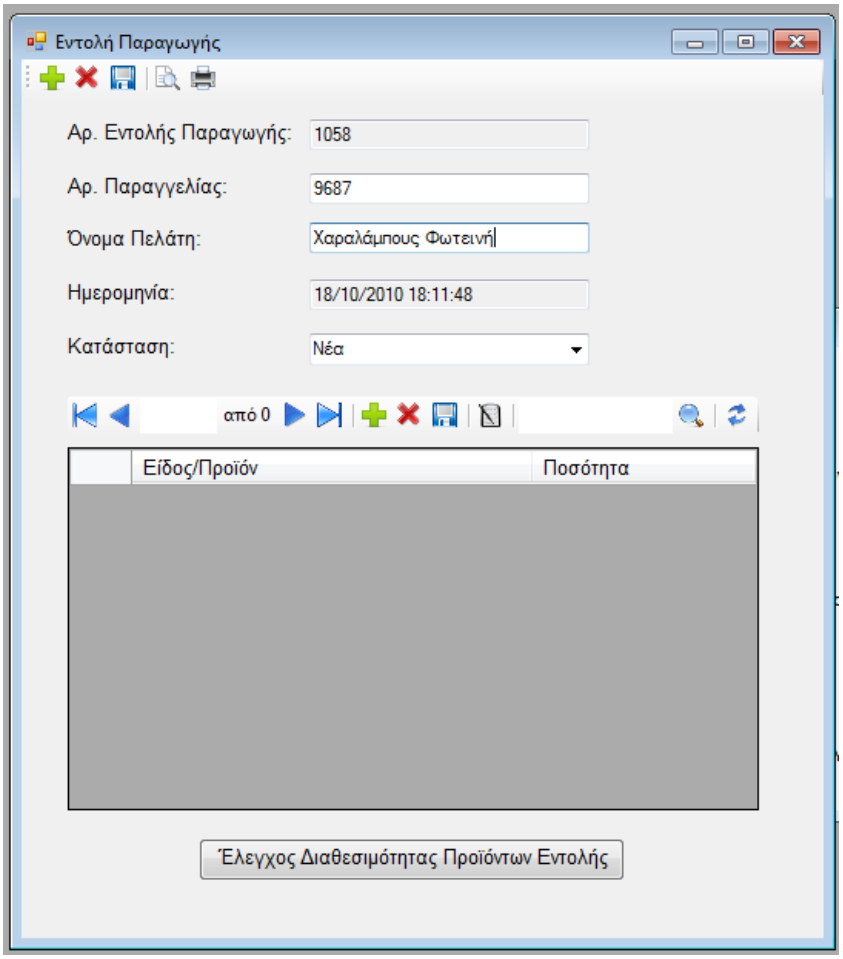


Δημιουργία Νέας Εντολής Παραγωγής

Για να μπορέσει ένα προϊόν να παραχθεί θα πρέπει να σταλεί στην παραγωγή του εργοστασίου μια εντολή παραγωγής για το συγκεκριμένο προϊόν. Η δημιουργία μιας εντολής παραγωγής μπορεί να γίνει από το πλαίσιο συντομεύσεων πατώντας το κουμπί «Νέα Εντολή Παραγωγής» ή από το μενού «Εντολές Παραγωγής» επιλέγοντας το «Νέα Εντολή Παραγωγής».

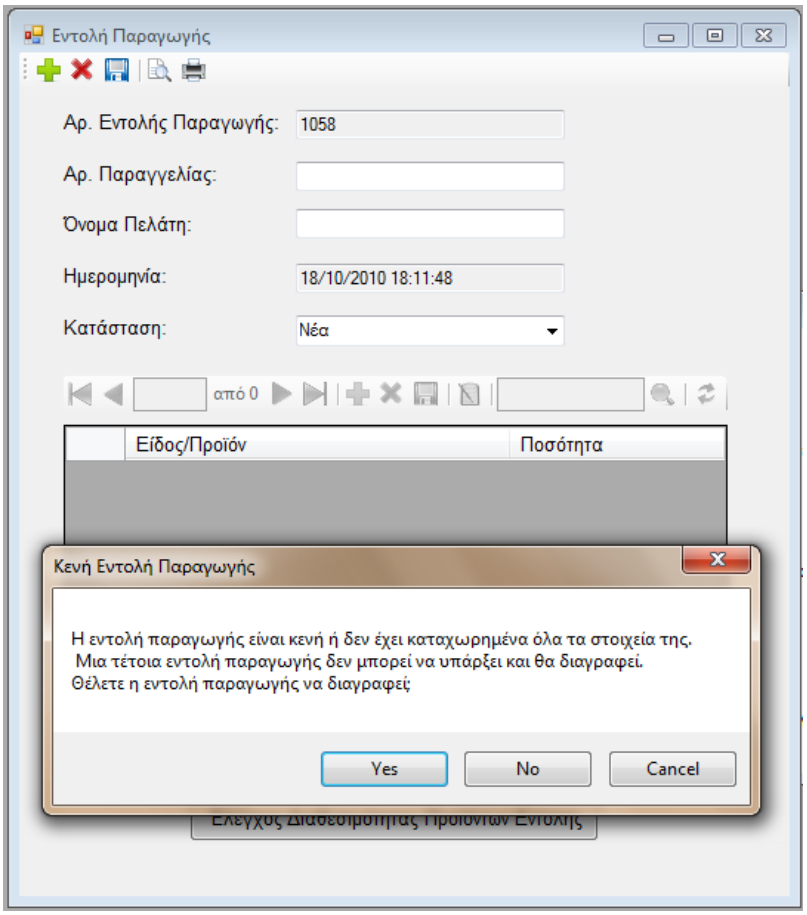


Όταν ανοίξει το παράθυρο για δημιουργία μιας νέας εντολής παραγωγής θα έχει ένα αριθμό εντολής παραγωγής που δίνεται αυτόματα από την εφαρμογή και την ημερομηνία που παίρνει από το σύστημα. Αν έχετε ορίσει στις γενικές παραμέτρους την κατάσταση που θα έχουν οι νέες εντολές παραγωγής τότε στη νέα εντολή παραγωγής που θα δημιουργήσετε θα δείτε να εμφανίζεται η κατάσταση αυτή.



Σε μια νέα εντολή παραγωγής θα πρέπει να συμπληρώσετε τα στοιχεία που υπολείπονται και τα καταχωρήσετε είδη/προϊόντα που θέλετε να παραχθούν με αυτή την εντολή παραγωγής. Δεν μπορείτε να αφήσετε μια εντολή παραγωγής κενή. Θα πρέπει να συμπληρώσετε και τα στοιχεία της και να καταχωρήσετε τουλάχιστον ένα προϊόν που θέλετε να παράγετε. Αν αφήσετε μια εντολή παραγωγής κενή και προσπαθήσετε να κλείσετε το παράθυρο θα δείτε ένα

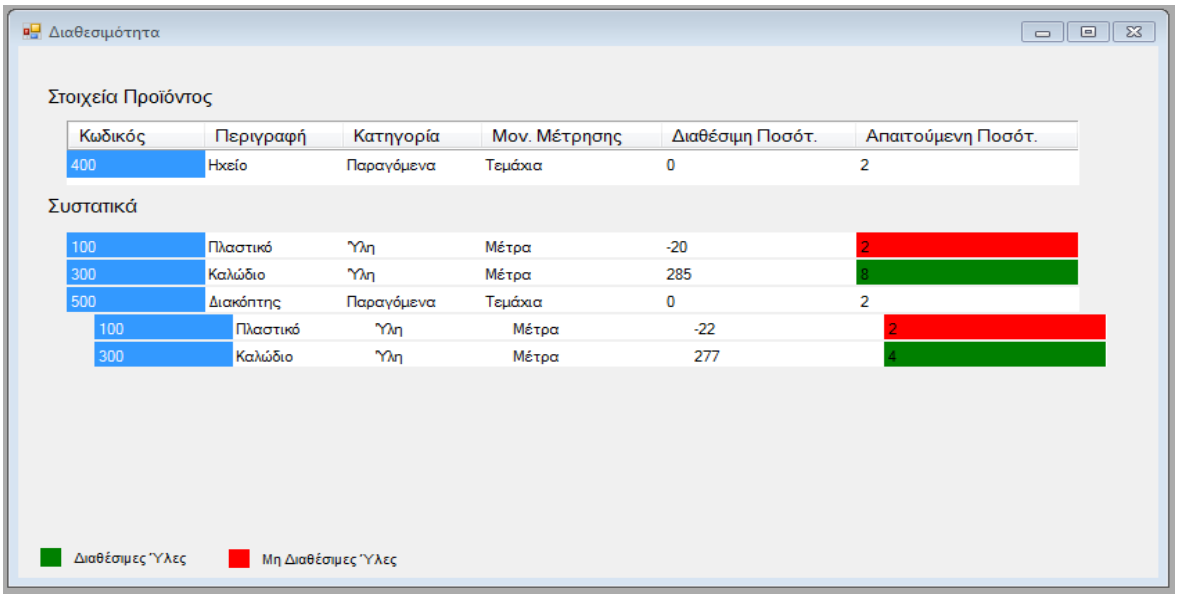
μήνυμα να εμφανίζεται που θα σας λέει ότι δεν μπορεί να υπάρξει μια κενή εντολή παραγωγής και θα διαγραφεί.



Τα προϊόντα που μπορείτε να καταχωρήσετε σε μια εντολή παραγωγής είναι μόνο αυτά που διαθέτουν συνταγή παραγωγής. Δηλαδή δεν μπορείτε να επιλέξετε προϊόντα που μπορεί να ανήκουν σε κάποια κατηγορία που να δέχεται συνταγή παραγωγής αλλά αυτά να μην έχουν συνταγή παραγωγής καταχωρημένη.

Στην γραμμή εργαλείων, που βρίσκεται πάνω από τον πίνακα των προϊόντων που υπάρχουν στην εντολή παραγωγής, εκτός από τα βασικά εργαλεία υπάρχει και ένα επιπλέον εργαλείο. Το εργαλείο έχει το παρακάτω εικονίδιο και σας δίνει την δυνατότητα να ελέγξετε, για ένα συγκεκριμένο προϊόν, αν υπάρχει διαθεσιμότητα των υλών του στην αποθήκη με βάση βέβαια την συνταγή παραγωγής του.

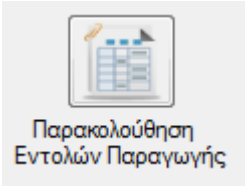
Αν δεν θέλετε να ελέγχετε ένα – ένα τα προϊόντα, που υπάρχουν στην εντολή παραγωγής για την διαθεσιμότητα τους, μπορείτε να πατήσετε το κουμπί «Έλεγχος Διαθεσιμότητας Προϊόντων Εντολής». Το κουμπί αυτό θα ελέγξει την διαθεσιμότητα των πρώτων υλών που χρειάζεται για την παραγωγή των προϊόντων και θα εμφανίσει τα αποτελέσματα αναλυτικά.



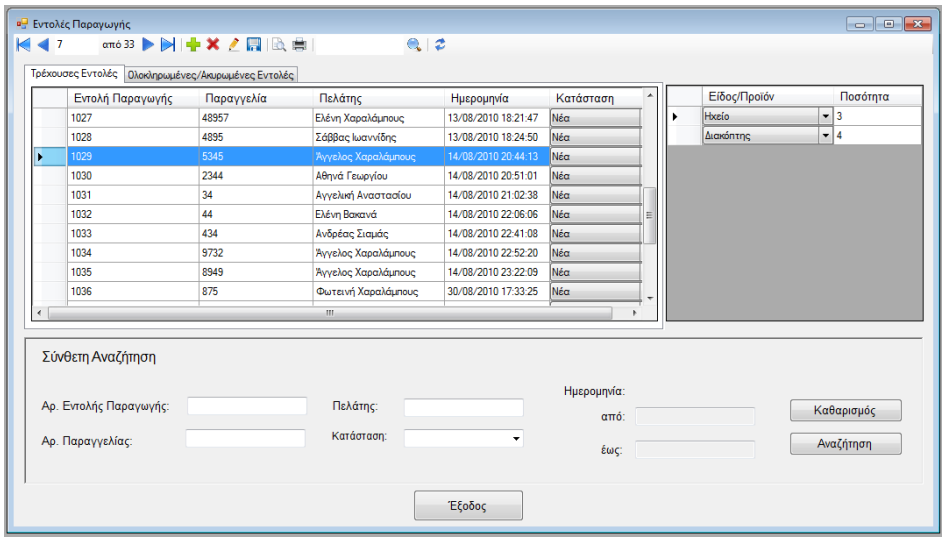
Με κόκκινο χρώμα φαίνονται τα μη διαθέσιμα είδη που χρειάζονται για να δημιουργηθούν τα προϊόντα ενώ με πράσινο χρώμα εμφανίζονται τα διαθέσιμα είδη.

Διαχείριση Εντολών Παραγωγής

Τις εντολές παραγωγής που έχετε δημιουργήσει, μπορείτε να τις παρακολουθήσετε και να τις διαχειριστείτε. Το παράθυρο παρακολούθησης μπορείτε να το ανοίξετε από το μενού «Εντολές Παραγωγής» και την επιλογή «Παρακολούθηση Εντολών Παραγωγής» ή από τις συντομεύσεις πατώντας το αντίστοιχο κουμπί.

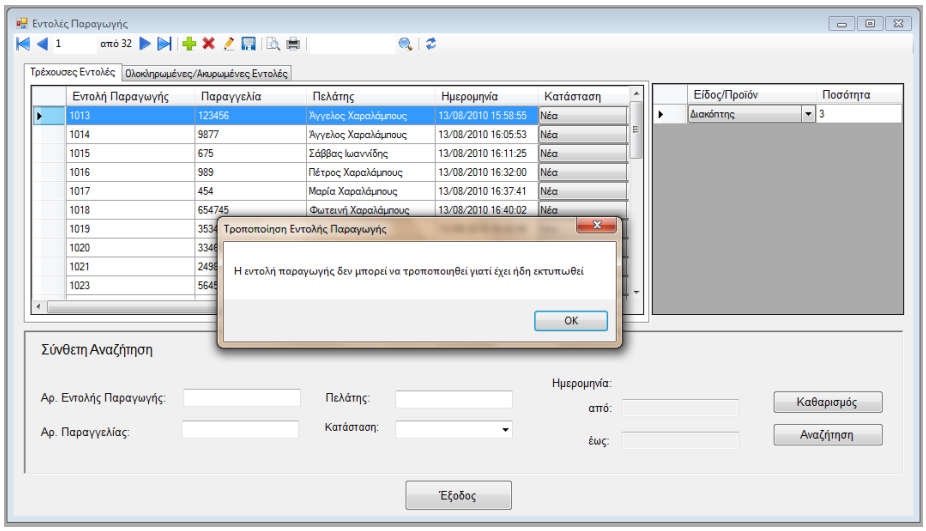


Μέσα στο παράθυρο που θα ανοίξει, αν έχετε ορίσει τις γενικές παραμέτρους της εφαρμογής, θα δείτε τις τρέχουσες εντολές παραγωγής σε μια καρτέλα και τις ολοκληρωμένες ή ακυρωμένες σε μια άλλη καρτέλα.

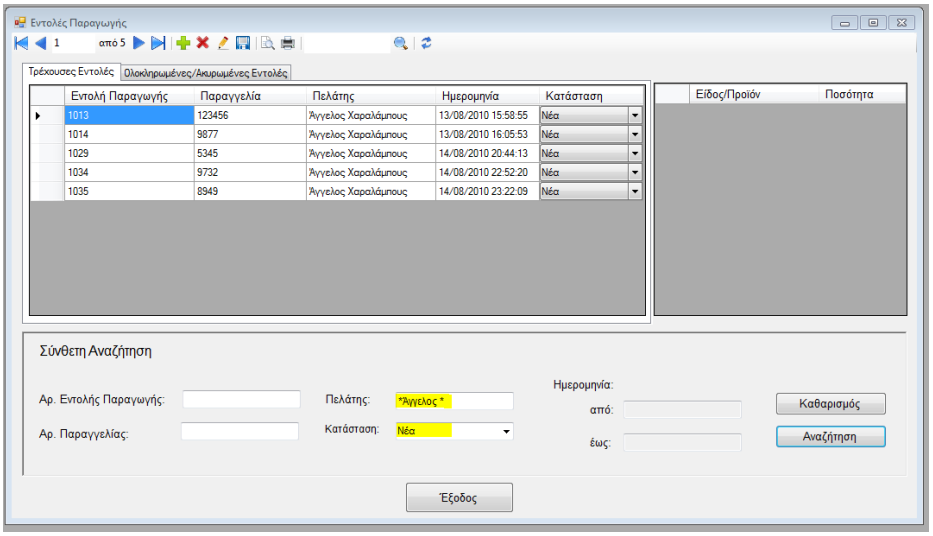


Αν θέλετε να δείτε τα προϊόντα που υπάρχουν σε κάποια εντολή παραγωγής αρκεί να την επιλέξετε και τα προϊόντα της, μαζί με τις ποσότητες τους, θα εμφανιστούν στο πίνακα που βρίσκεται στο γκρί πλαίσιο δεξιά.

Τα δεδομένα των εντολών παραγωγής μπορείτε να τα τροποποιήστε αν θέλετε. Η τροποποίηση μπορεί να γίνει αν επιλέξετε την εντολή που θέλετε και πατήστε από την μπάρα εργαλείων το «μολύβι». Προϋπόθεση για να μπορέσετε να τροποποιήσετε τα δεδομένα μιας εντολής παραγωγής είναι η εντολή αυτή να μην έχει εκτυπωθεί. Η εκτύπωση μιας εντολής παραγωγής σημαίνει ότι η εντολή έχει σταλεί στην παραγωγή.

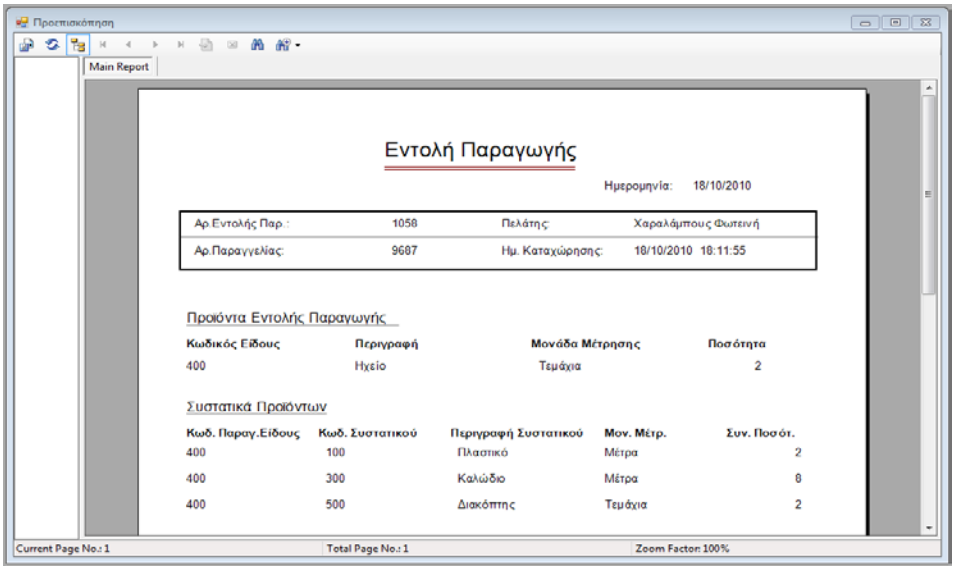


Οι εντολές παραγωγής χωρίζονται σε τρέχουσες και ολοκληρωμένες ή ακυρωμένες. Μπορείτε όμως να εμφανίσετε όποιες εντολές θέλετε με τη βοήθεια της σύνθετης αναζήτησης που υπάρχει. Στη αναζήτηση συμπληρώνετε τα πεδία που θέλετε και πατάτε το κουμπί «Αναζήτηση». Ανάλογα με την καρτέλα στην οποία βρίσκεστε θα εμφανιστούν οι εντολές παραγωγής που «πληρούν τα κριτήρια» της αναζήτησης.



Τις εντολές παραγωγής που είναι τρέχουσες μπορείτε να τις εκτυπώσετε ή να τους τροποποιήσετε τα στοιχεία τους, να τους αλλάξετε την κατάσταση στην οποία βρίσκονται ή να την ακυρώσετε αν θέλετε.

Πριν εκτυπώσετε την εντολή παραγωγής μπορείτε να κάνετε προεπισκόπηση εκτύπωσης και να δείτε πως θα είναι η εντολή παραγωγής όταν την εκτυπώσετε.



Εμφάνιση Κατάστασης Ελλείψεων

Για να μπορείτε να προγραμματίσετε καλύτερα την παραγγελία των πρώτων υλών σας, η εφαρμογή σας, εμφανίζει με βάση το όριο αναπαραγγελίας που έχετε ορίσει, τις ελλείψεις των πρώτων υλών σας.

Η εμφάνιση των ελλείψεων της αποθήκης γίνεται αν πατήσετε το κουμπί «Κατάσταση Ελλείψεων» από το πλαίσιο των συντομεύσεων ή αν πατήσετε την αντίστοιχη επιλογή από το μενού «Έντυπο».