ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΔΠΜΣ "Τεχνοοικονομικά Συστήματα Διοίκησης"

Μεταπτυχιακή Εργασία

‘’Ανάπτυξη συστήματος διαχείρισης παραγωγής για βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου που χρησιμοποιούνται στην αυτοκινητοβιομηχανία’’

Μεταπτυχιακός φοιτητής

Λεκάι Ευθύμιος

Αρ. Μητρώου: 7344012200009

Επιβλέπουσα : Καθηγήτρια Αικατερίνη Μαρινάγη

Αθήνα, Ιούνιος 2024

Η παρούσα Μεταπτυχιακή Διπλωματική Εργασία και τα συμπεράσματά της, σε οποιαδήποτε μορφή, αποτελούν συνιδιοκτησία των Τμημάτων ΔΕΟ του ΕΚΠΑ και ΔΙΓΕΣΕ του ΓΠΑ και του φοιτητή. Οι προαναφερόμενοι διατηρούν το δικαίωμα ανεξάρτητης χρήσης και αναπαραγωγής (τμηματικά ή συνολικά) για διδακτικούς και ερευνητικούς σκοπούς. Σε κάθε περίπτωση πρέπει να αναφέρεται ο τίτλος, ο συγγραφέας, ο Επιβλέπων και τα συνεργαζόμενα Τμήματα.

Η έγκριση της παρούσας Μεταπτυχιακής Διπλωματικής Εργασίας από τα συνεργαζόμενα Τμήματα δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του/ης συγγραφέα/ως εκ μέρους των Τμημάτων.

Ο υπογεγραμμένος δηλώνω υπεύθυνα ότι η παρούσα Μεταπτυχιακή Διπλωματική Εργασία είναι εξ' ολοκλήρου δικό μου έργο και συγγράφηκε ειδικά για τις απαιτήσεις του ΔΠΜΣ των Τμημάτων ΔΕΟ του ΕΚΠΑ και ΔΙΓΕΣΕ του ΓΠΑ με τίτλο "Τεχνοοικονομικά Συστήματα Διοίκησης".

Δηλώνω υπεύθυνα ότι κατά τη συγγραφή ακολούθησα την πρέπουσα ακαδημαϊκή δεοντολογία αποφυγής λογοκλοπής και έχω αποφύγει οποιαδήποτε ενέργεια που συνιστά παράπτωμα λογοκλοπής.

Όνοματεπώνυμο: Ευθύμιος Λεκάι

Υπογραφή: 

Ημερομηνία: 21/06/2024

« Ου γαρ δοκείν άριστος, αλλ' είναι θέλει »

Αισχύλος

Πρόλογος

Η παρούσα μεταπτυχιακή εργασία εκπονήθηκε στα πλαίσια του Διιδρυματικού Προγράμματος Μεταπτυχιακών Σπουδών (ΔΠΜΣ) «Τεχνοοικονομικά Συστήματα Διοίκησης» του Τμήματος Διοίκησης Επιχειρήσεων και Οργανισμών του Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών και του Τμήματος Διοίκησης Γεωργικών Επιχειρήσεων και Συστημάτων Εφοδιασμού του Γεωπονικού Πανεπιστημίου Αθηνών.

Η διαδικασία ολοκλήρωσης της μεταπτυχιακής μου εργασίας αποτέλεσε μια περίοδο προσωπικής ανάπτυξης. Μέσα σε αυτό το διάστημα, ανακάλυψα νέες πτυχές του εαυτού μου και ανέπτυξα ικανότητες τόσο στην διαχείριση του χρόνου όσο και στην αναζήτηση και εντοπισμό χρήσιμων πληροφοριών. Η αφοσίωση στο αντικείμενο της μελέτης, μου επέτρεψε να εμβαθύνω τις γνώσεις μου και να εξερευνήσω νέους ορίζοντες.

Κατά τη διάρκεια της εργασίας μου, αντιμετώπισα πολλές προκλήσεις, από την εύρεση σχετικής βιβλιογραφίας μέχρι την επίλυση σύνθετων προβλημάτων. Αυτές οι προκλήσεις ωστόσο, συνέβαλαν στην ανάπτυξη της ικανότητάς μου να αντιμετωπίζω εμπόδια και να βρίσκω δημιουργικές λύσεις. Μέσα από αυτή την εμπειρία, κέρδισα μια πιο σφαιρική άποψη για την επιστήμη μου και για τον τρόπο που η έρευνα μπορεί να εφαρμοστεί για την επίλυση πρακτικών προβλημάτων.

Στο πλαίσιο της ολοκλήρωσης της μεταπτυχιακής μου εργασίας, θα ήθελα να ευχαριστήσω, πρωτίστως την επιβλέπουσα καθηγήτρια μου, κυρία Αικατερίνη Μαρινάγη, η οποία μου παρείχε καθοδήγηση και στήριξη καθ' όλη τη διάρκεια εκπόνησης της εργασίας και αποτέλεσε καθοριστικό παράγοντα για την ολοκλήρωσή της. Επιπρόσθετα, θα ήθελα να ευχαριστήσω την οικογένεια μου για την αμέριστη συμπαράσταση και κατανόηση που έδειξαν όλο αυτό το διάστημα.

Περίληψη

Η εκπόνηση της μεταπτυχιακής εργασίας έχει ως στόχο, τον σχεδιασμό, την ανάπτυξη και την υλοποίηση λογισμικού διαχείρισης παραγωγής σε βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου υπό πίεση για χρήση στην αυτοκινητοβιομηχανία.

Η διαδικασία της ανάπτυξης λογισμικού ξεκίνησε με την ανάλυση των απαιτήσεων. Κατά την διάρκεια αυτής της φάσης πραγματοποιήθηκαν συνεντεύξεις και υποβολές ερωτηματολογίων από τους δυνητικούς χρήστες του λογισμικού. Επίσης συλλέχθηκαν πληροφορίες και δεδομένα από τον τρόπο λειτουργίας της βιομηχανίας. Αυτή η διαδικασία βοήθησε στην δημιουργία ενός πλήρους και ακριβούς σχεδίου για την ανάπτυξη του λογισμικού, εξασφαλίζοντας ότι οι ανάγκες της επιχείρησης, θα ικανοποιηθούν πλήρως. Επιπλέον, η ανάλυση αυτή ήταν σημαντική για την αποφυγή πιθανών προβλημάτων ή παρερμηνειών κατά την υλοποίηση του λογισμικού.

Έπειτα, ακολούθησε ο σχεδιασμός του λογισμικού, με έμφαση στην δημιουργία ενός συστήματος που θα προσφέρει αποτελεσματικές λύσεις στα προβλήματα που εντοπίστηκαν κατά την ανάλυση. Ορίστηκαν οι βασικές οντότητες που θα εμπλέκονται στο σύστημα και επιλέχθηκε το μοντέλο Καταρράκτη για τον κύκλο ζωής του Λογισμικού. Επίσης ορίστηκε το Διάγραμμα Οντοτήτων – Συσχετίσεων και το Σχήμα της Βάσης Δεδομένων.

Επόμενο στάδιο είναι η υλοποίηση του λογισμικού η οποία πραγματοποιήθηκε με χρήση της γλώσσας προγραμματισμού Python. Η επιλογή της παραπάνω γλώσσας έγινε με κριτήρια την ευρεία γκάμα βιβλιοθηκών, την υποστήριξη πολλών στιλ προγραμματισμού και το γεγονός ότι είναι ανοιχτού κώδικα. Επίσης αποτελεί γλώσσα υψηλού επιπέδου, το οποίο σημαίνει ότι είναι ευανάγνωστη και κατανοητή, καθιστώντας την ιδανική για την ανάπτυξη λογισμικού που θα διαχειρίζεται πολύπλοκες λειτουργίες και δεδομένα.

Τέλος, στο τελευταίο στάδιο, μετά την υλοποίηση του πηγαίου κώδικα ακολουθεί η δοκιμή του λογισμικού, όπου χρησιμοποιήθηκαν πραγματικά δεδομένα από το τμήμα παραγωγής βιομηχανίας παραγωγής χυτών εξαρτημάτων αλουμινίου.

Abstract

The aim of this postgraduate thesis is to design, develop, and implement a software for product management in a high pressure aluminum die casting industry, which produces parts to be used in automotive industry.

The software development process started with requirements analysis. During this phase, interviews and questionnaire submissions were conducted with potential users of the software. Additionally, information and data were collected regarding the operation of the industry. This process aided in creating a comprehensive and accurate plan for the software development, ensuring that the needs of the business would be fully met. Furthermore, this analysis was crucial to avoid potential problems or misunderstandings during the software implementation.

Next, software design followed, focusing on creating a system that would provide effective solutions to the problems identified during analysis. The key entities to be involved in the system were defined, and the Waterfall model was chosen for the software lifecycle. Additionally, the Entity-Relationship Diagram was defined as well as the Database Schema.

The next stage was the implementation of the software, which was carried out using the Python programming language. The selection of Python was based on criteria such as its wide range of libraries, support for multiple programming styles, and the fact that it is open source. Additionally, it is a high-level language, meaning it is readable and understandable, making it ideal for developing software that manages complex operations and data.

Finally, in the last stage, after the implementation of the source code, we proceeded with testing the software, using real data from the production department of an aluminum die casting industry.

Περιεχόμενα

Κατάλογος Εικόνων

Πίνακας συντομεύσεων – ακρωνύμια

AD	Active Database
AI	Artificial Intelligent
API	Application Programming Interface
CRM	Customer Relationship Management
DB	Database
DBMS	Database Management System
DW	Database Warehouse
ERP	Enterprise Resource Planning
ENIAC	Electronic Numerical Integrator and Computer
FDBMS	Federal Database Management System
HCM	Human Capital Management
IBM	International Business Machines
IoT	Internet of Things
ML	Machine Learning
MMDB	Main Memory Database System
MRP	Material Requirements Planning
SAP	System Applications and Products
SCM	Supply Chain Management
SE	Software Engineering
ΠΣ	Πληροφοριακά Συστήματα
ΤΛ	Τεχνολογία Λογισμικού

ΚΕΦΑΛΑΙΟ 1: Εισαγωγή

1.1 Στόχος εργασίας

Η μεταπτυχιακή εργασία έχει ως στόχο την ανάπτυξη ενός λογισμικού διαχείρισης παραγωγής στον κλάδο της αυτοκινητοβιομηχανίας, και πιο συγκεκριμένα στις βιομηχανίες χύτευσης αλουμινίου υπό πίεση. Με βάση τη συνεχή εξέλιξη και την τεχνολογική πρόοδο στον τομέα, γίνεται μια προσπάθεια βελτίωσης της απόδοσης, της παραγωγικότητας και της αποτελεσματικότητας στις εγκαταστάσεις παραγωγής.

Ο κύριος στόχος της εργασίας είναι η σχεδίαση, ανάπτυξη και υλοποίηση ενός συστήματος που θα επιτρέπει την αποτελεσματική διαχείριση των διαδικασιών παραγωγής. Το Λογισμικό θα προσφέρει λύσεις για την παρακολούθηση της παραγωγής, τον έλεγχο των αποθεμάτων και τον συντονισμό των διαφόρων διαδικασιών εντός του εργοστασίου.

Τέλος, η παρούσα μεταπτυχιακή εργασία έχει ως στόχο την προώθηση της έρευνας και της ανάπτυξης στον τομέα της διαχείρισης παραγωγής, συμβάλλοντας έτσι στη συνεχή βελτίωση και καινοτομία του κλάδου. Μέσω της εφαρμογής και της αξιολόγησης αυτού του λογισμικού, επιδιώκεται η ενίσχυση της ανταγωνιστικότητας και αποδοτικότητας των επιχειρήσεων.

1.2 Μεθοδολογία εργασίας

Η μεθοδολογία που χρησιμοποιείται στο πλαίσιο της μεταπτυχιακής εργασίας είναι η ανάπτυξη λογισμικού.

Αρχικά, πραγματοποιείται λεπτομερής ανάλυση των απαιτήσεων και των αναγκών της βιομηχανίας σε σχέση με τη διαχείριση παραγωγής. Αυτό συμπεριλαμβάνει τη συλλογή δεδομένων, τη διεξαγωγή συνεντεύξεων, την υποβολή ερωτηματολογίων και την ανάλυση των τρεχουσών πρακτικών και διαδικασιών.

Έπειτα, σχεδιάζεται η αρχιτεκτονική του συστήματος με τακτική αναθεώρηση και αναβάθμιση των απαιτήσεων, με την στενή συνεργασία της βιομηχανίας.

Στην συνέχεια, υλοποιείται η ανάπτυξη του πηγαίου κώδικα με την γλώσσα προγραμματισμού Python, και γίνονται προσομοιώσεις και δοκιμές, προκειμένου να

διασφαλίσουμε την αξιοπιστία, την αποδοτικότητα και την ασφάλεια του λογισμικού. Τέλος όπου είναι αυτό δυνατόν, πραγματοποιούνται βελτιώσεις με στόχο την εξάλειψη ενδεχόμενων αδυναμιών και σφαλμάτων.

1.3 Δομή εργασίας

Το πρώτο κεφάλαιο παρέχει μια επισκόπηση του περιεχομένου της εργασίας. Παρουσιάζεται ο στόχος της εργασίας καθώς και η μεθοδολογία και η δομή που ακολουθείται.

Το δεύτερο κεφάλαιο αναλύει βασικά ζητήματα των πληροφοριακών συστημάτων, όπως την τεχνολογία λογισμικού, την τεχνολογία των πληροφοριακών συστημάτων, καθώς και την εξέλιξη αυτών μέσα στον χρόνο. Επίσης, αναλύεται η σημασία των βάσεων δεδομένων και η διεπαφή ανθρώπου-μηχανής στο πλαίσιο της ανάπτυξης λογισμικού.

Στο τρίτο κεφάλαιο αναλύονται τα μοντέλα κύκλου ζωής λογισμικού και τα κριτήρια επιλογής τους.

Το τέταρτο κεφάλαιο επικεντρώνεται στις βιομηχανίες αλουμινίου που παράγουν εξαρτήματα για χρήση στην αυτοκινητοβιομηχανία, εξετάζοντας πώς η ενσωμάτωση Π.Σ μπορεί να επιφέρει νέες δυνατότητες. Επίσης αναδεικνύει τον τρόπο με τον οποίο τα Π.Σ μπορούν να ενισχύσουν την παραγωγικότητα, την ποιότητα και τη βιωσιμότητα των βιομηχανιών αυτών.

Το πέμπτο κεφάλαιο εστιάζει στη ανάπτυξη λογισμικού διαχείρισης παραγωγής σε βιομηχανία χύτευσης εξαρτημάτων αλουμινίου. Συγκεκριμένα αναλύει τις απαιτήσεις του συστήματος, σχεδιάζει την αρχιτεκτονική και τη λειτουργικότητα του λογισμικού, και υλοποιεί την δημιουργία του πηγαίου κώδικα. Επίσης, περιλαμβάνει την εκτέλεση δοκιμών και τη διενέργεια ελέγχων για τη διασφάλιση της καλής λειτουργίας του.

Το έκτο και τελευταίο κεφάλαιο αποτυπώνει τα κύρια συμπεράσματα που προέκυψαν από την εργασία και παρέχει κατευθύνσεις για μελλοντική έρευνα, επισημαίνοντας πιθανές επεκτάσεις, βελτιώσεις ή νέες προοπτικές για την εξέλιξη του Λογισμικού.

ΚΕΦΑΛΑΙΟ 2: Πληροφοριακά Συστήματα

2.1 Τεχνολογία Λογισμικού

Ένας ηλεκτρονικός υπολογιστής είναι σε θέση να πραγματοποιεί εκατομμύρια υπολογισμούς το δευτερόλεπτο. Η εισαγωγή του ηλεκτρονικού υπολογιστή όμως σε πρακτικές εφαρμογές, δε θα ήταν δυνατή χωρίς την χρήση Λογισμικού, καθώς ο άνθρωπος δεν μπορεί να αλληλεπιδράσει άμεσα με έναν ηλεκτρονικό υπολογιστή (Φιτσιλής, 2015).

Το λογισμικό δεν αποτελεί αυτοτελή υπόσταση αλλά είμαστε σε θέση να δούμε τα αποτελέσματά του. Η διαδικασία κατασκευής του μπορεί να κωδικοποιηθεί δύσκολα διότι δεν είναι ξεκάθαρο πια είναι τα βήματα που πρέπει να ακολουθήσουμε και με πια σειρά. Όμως όπως και με τις τεχνικές κατασκευές έτσι και η κατασκευή Λογισμικού δεν είναι δυνατόν να οδηγήσει κατευθείαν σε έναν εκτελέσιμο κώδικα χωρίς να έχει προηγηθεί μελέτη. Αυτό που το διαφοροποιεί ωστόσο σημαντικά σε σχέση με τις τεχνικές κατασκευές είναι το γεγονός ότι το παραδοτέο προϊόν, αποτελεί μία πρώτη έκδοσή του η οποία υπόκεινται στην συνέχεια σε τροποποιήσεις και βελτιώσεις (Βεσκούκης, 2015).

Η επιστήμη της πληροφορικής, ανέπτυξε έναν ειδικό κλάδο για να αποτυπώσει την οντότητα του Λογισμικού η οποία ονομάστηκε στην συνέχεια «Τεχνολογία Λογισμικού» (Software Engineering). Έτσι η Τεχνολογία Λογισμικού είναι η επιστήμη η οποία ασχολείται με την εύρεση και την θεμελίωση μεθόδων, για την περιγραφή, κατασκευή και συντήρηση ενός Λογισμικού. (Schmidt, 2013).

Υπάρχουν δύο κατηγορίες λογισμικών. Η πρώτη κατηγορία αφορά τα Λογισμικά συστήματος τα οποία είναι υπεύθυνα για την λειτουργία ενός ηλεκτρονικού υπολογιστή. Τέτοια συστήματα αποτελούν τα Windows, τα Linux κλπ. Η δεύτερη κατηγορία αφορά τα Λογισμικά εφαρμογών. Στην κατηγορία αυτή έχουμε τα προγράμματα τα οποία αποδέκτης είναι ο τελικός χρήστης ο οποίος μπορεί να είναι είτε άνθρωπος ή ένα άλλο πρόγραμμα εφαρμογών. Τέτοια λογισμικά μπορεί να είναι οι επεξεργαστές κειμένου και τα λογιστικά φύλλα (Μητάκος, 2015).

2.2 Τεχνολογία Πληροφοριακών Συστημάτων

Ένα Π.Σ είναι ένα σύστημα το οποίο δέχεται δεδομένα ως είσοδο, στην συνέχεια επεξεργάζεται τα δεδομένα και τέλος παράγει πληροφορίες στην έξοδο. Τα δεδομένα εισάγονται είτε από άνθρωπο είτε από κάποιο μηχάνημα είτε και από κάποιο άλλο Π.Σ. Η επεξεργασία αφορά την μετατροπή των δεδομένων σε πληροφορίες κατά την έξοδο μέσω υπολογισμών, λογικών πράξεων και συγκρίσεων (Μητάκος, 2015).

Το Π.Σ περιλαμβάνει τον υλικό εξοπλισμό (hardware), το λογισμικό (software), τους ανθρώπους, τα δεδομένα και τις διαδικασίες. Ο υλικός εξοπλισμός αφορά τις υποδομές οι οποίες έχουν φυσική υπόσταση. Το λογισμικό αφορά τις εντολές που εκτελεί ο υπολογιστής και μπορούμε να δούμε μόνο τα αποτελέσματά του. Οι άνθρωποι περιλαμβάνουν τους χρήστες του συστήματος και τους εξειδικευμένους εργαζόμενους που συντηρούν και ενημερώνουν το σύστημα (Μητάκος, 2015). Τα δεδομένα είναι τα στοιχεία τα οποία παράγονται από το σύστημα έπειτα από την επεξεργασία τους. Οι διαδικασίες περιλαμβάνουν τους τρόπους με τους οποίους τα δεδομένα συλλέγονται, επεξεργάζονται, αποθηκεύονται και διατίθενται (Hustad & Stensholt, 2023).

Τα πληροφοριακά συστήματα διαδραματίζουν καίριο ρόλο στις σύγχρονες επιχειρήσεις ανεξαρτήτως του αντικειμένου που απασχολούνται. Επιτρέπουν στις επιχειρήσεις να αυξήσουν την αποδοτικότητα των εσωτερικών διαδικασιών, να βελτιώσουν την ποιότητα των προϊόντων και των υπηρεσιών, να ενισχύσουν την ικανότητα επικοινωνίας τόσο με τους πελάτες όσο και με τους προμηθευτές και να βοηθήσουν στην ανάπτυξη νέων ανταγωνιστικών στρατηγικών (Μιαούλης, Μπουσδέκης, & Θεοδωροπούλου, 2021).

2.3 Εξέλιξη των Πληροφοριακών Συστημάτων

Η εξέλιξη των ΠΣ είναι στενά συνδεδεμένη με την άνθηση των ηλεκτρονικών υπολογιστών. Έως και την δεκαετία του 1950 οι επιχειρήσεις και οι οργανισμοί διαχειρίζονταν τις διάφορες πληροφορίες χειρωνακτικά με την χρήση χαρτιού και φυσικών αρχείων (Βεσκούκης, 2015).

Την δεκαετία του 1960 είχαμε την εμφάνιση των πρώτων υπολογιστών όπως ο ENIAC, που χρησιμοποιούνταν κυρίως για στρατιωτικούς και ερευνητικούς σκοπούς (Βεσκούκης, 2015). Την ίδια δεκαετία και με την ανάπτυξη των πρώτων υπολογιστών η ανάγκη για την διαχείριση της παραγωγής και της ζήτησης, οδήγησε στην δημιουργία του Material Requirements Planning - MRP (Προγραμματισμός Απαιτήσεων Υλικών). Η J.I. Case, παραγωγός μηχανημάτων κατασκευής και γεωργικών μηχανημάτων, συνεργάστηκε με την IBM για να καταφέρει να το αναπτύξει (Sommerville, 2011). Η διαδικασία της ανάπτυξης ήταν ιδιαίτερα κοστοβόρα και χρειάστηκε να δουλέψει μία ειδική ομάδα ώστε να το ολοκληρώσει. Οι επιχειρήσεις ήταν σε θέση να παρακολουθούν τα αποθέματα και την παραγωγή ώστε να μπορούν να σχεδιάζουν καλύτερα τις παραγωγικές διαδικασίες.

Κατά τη δεκαετία του 1970, εκατοντάδες εταιρείες χρησιμοποιούσαν το λογισμικό MRP, παρόλο που ήταν πολύ ακριβό και λειτουργούσε σε τεράστιους υπολογιστές. Το 1972, η SAP ιδρύθηκε στη Γερμανία με στόχο τη δημιουργία λογισμικού που λειτουργούσε σε πραγματικό χρόνο, κάτι που δεν είχε γίνει ποτέ ξανά.

Την δεκαετία του 1980 ενισχύθηκαν οι δυνατότητες του MRP, το οποίο άλλαξε ονομασία σε Manufacturing Resource Planning - MRP II (Προγραμματισμός Πόρων Παραγωγής). Το λογισμικό έγινε πιο πολύπλοκο και προστέθηκαν περισσότερες δυνατότητες στην διαχείριση διαδικασιών προγραμματισμού και παραγωγής (Βεσκούκης, 2015).

Την δεκαετία του 1990 εμφανίζεται ο όρος Enterprise Resource Planning – ERP (Διαχείριση Επιχειρησιακών Πόρων) για να περιγράψει πληρέστερα τις δυνατότητες του λογισμικού, ενώ αρχίζουν και άλλοι προμηθευτές να παράγουν αντίστοιχα λογισμικά όπως η Oracle, JD Edwards και η Visibility. Τα συστήματα επεκτάθηκαν πέρα από τις βασικές διαδικασίες ελέγχου αποθεμάτων και κατασκευής και περιλάμβαναν πλέον και άλλα τμήματα όπως τα οικονομικά και οι πωλήσεις.

Το 2000, η Gartner επινόησε άλλον έναν όρο - ERP II, ο οποίος αναφερόταν σε λογισμικό ERP το οποίο λειτουργούσε σε πραγματικό χρόνο μέσω διαδικτύου. Η δεκαετία του 2000 περιλάμβανε επίσης την ένταξη της διαχείρισης πελατειακών σχέσεων (Customer Relationship Management - CRM), αλυσίδας εφοδιασμού (Supply Chain Management - SCM) και ανθρωπίνων πόρων (Human Resource Management -

HRM). Έπειτα με την εξάπλωση του διαδικτύου ξεκίνησαν να εμφανίζονται τεχνολογίες όπως το Υπολογιστικό Νέφος (cloud computing), το Διαδίκτυο των Πραγμάτων (Internet of Things – IoT) και η τεχνητή νοημοσύνη (artificial intelligence) μετατρέποντας τα Π.Σ σε αυτά που γνωρίζουμε σήμερα (Sommerville, 2011).

Σήμερα, τα κορυφαία συστήματα ERP έχουν τεράστια αποθέματα πληροφοριών και μπορούν να παράγουν αναφορές οι οποίες μπορούν να αναδείξουν την απόδοση κάθε πτυχής της επιχείρησης, από τις πωλήσεις και το μάρκετινγκ μέχρι την ανάπτυξη προϊόντων. Υπάρχουν αμέτρητες διαθέσιμες εφαρμογές, σχεδιασμένες για διάφορους κλάδους, ώστε να καλύψουν τις μοναδικές και ιδιαίτερες ανάγκες και προκλήσεις τους (Chopra, Sawant, Kodi, & Terkar, 2022).

Ο ταχεία μεταβαλλόμενος ρυθμός των τεχνολογικών εξελίξεων και οι αλλαγές στις επιχειρηματικές ανάγκες αναμένεται να έχουν σημαντικό αντίκτυπο στο μέλλον των ERP. Αυτά τα συστήματα θα συνεχίσουν να αναπτύσσονται καθώς οι επιχειρήσεις αναζητούν πιο οικονομικές, αποδοτικές και εξατομικευμένες λύσεις (Schmidt, 2013). Η υιοθέτηση του cloud computing, η ενσωμάτωση με αλγορίθμους τεχνητής νοημοσύνης και μηχανικής μάθησης και η αυξημένη επικέντρωση στην ασφάλεια δεδομένων αναμένεται να καθορίσουν το μέλλον των συστημάτων ERP (Gessa, Jiménez, & Sancha, 2023). Οι επιχειρήσεις που υιοθετούν αυτές τις τεχνολογίες θα αποκτήσουν ανταγωνιστικό πλεονέκτημα και θα είναι καλύτερα προετοιμασμένες να αντιμετωπίσουν τις δυσκολίες που παρουσιάζει η ταχεία αλλαγή του επιχειρηματικού περιβάλλοντος (Hustad & Stensholt, 2023).

2.4 Βάσεις δεδομένων

Σε ένα πληροφοριακό σύστημα, οι βάσεις δεδομένων είναι ουσιαστικά η αποθηκευτική δομή που χρησιμοποιείται για την αποθήκευση και τη διαχείριση των δεδομένων. Αναλαμβάνουν τον ρόλο ενός αποθηκευτικού χώρου όπου τα δεδομένα μπορούν να οργανωθούν, να αναζητηθούν, και να ανακτηθούν με αποτελεσματικό τρόπο. Μπορεί να περιλαμβάνουν πληροφορίες για πελάτες, προϊόντα, παραγγελίες, οικονομικά στοιχεία και πολλά άλλα, ανάλογα με τη φύση του συστήματος και τις ανάγκες της επιχείρησης ή του οργανισμού (Φιτσιλής, 2015). Η γνώση των διαφορετικών επιλογών βάσεων δεδομένων, επηρεάζει τον σχεδιασμό, τόσο σε φυσικό

επίπεδο όσο και σε λογικό επίπεδο κατά την δημιουργία ενός Π.Σ. Στις παρακάτω παραγράφους γίνεται μια αναφορά στις σημαντικότερες κατηγορίες βάσεων δεδομένων (Μητάκος, 2015).

2.4.1 Βάση δεδομένων στην μνήμη

Οι βάσεις δεδομένων στην μνήμη (main memory database system MMDB) βρίσκονται πρωτίστως στην κύρια μνήμη, αλλά συνήθως δημιουργούνται αντίγραφα ασφαλείας σε μία αποθήκη δεδομένων. Αποτελούν καλή επιλογή όταν κύρια απαίτηση είναι η ταχύτητα απόκρισης του συστήματος (Βεσκούκης, 2015).

2.4.2 Ενεργή βάση δεδομένων

Μια ενεργή βάση δεδομένων (active database) βασίζεται σε γεγονότα και μπορεί να ανταποκρίνεται σε συνθήκες τόσο εντός όσο και εκτός της βάσης. Βρίσκει εφαρμογή σε περιπτώσεις όπου θέλουμε να έχουμε παρακολούθηση της ασφάλειας, ειδοποιήσεις και συλλογή στατιστικών στοιχείων.

2.4.3 Βάση δεδομένων στο νέφος

Μία βάση δεδομένων στο νέφος (cloud) βασίζεται στην τεχνολογία cloud. Η βάση δεδομένων αλλά και το μεγαλύτερο μέρος του DBMS της, βρίσκονται απομακρυσμένα στο νέφος, ενώ οι εφαρμογές της, αναπτύσσονται, συντηρούνται και χρησιμοποιούνται μέσω ενός προγράμματος περιήγησης ιστού και των Open API (Κοκκίνου, 2023).

2.4.4 Αποθήκη δεδομένων

Η αποθήκη δεδομένων (Data Warehouse-DW) χρησιμοποιείται για ανάλυση δεδομένων. Τα DW είναι κεντρικές αποθήκες δεδομένων οι οποίες προέρχονται από μία η περισσότερες διαφορετικές πηγές. Αποθηκεύουν τρέχοντα και ιστορικά δεδομένα και χρησιμοποιούνται για την εξαγωγή, ανάλυση και διαχείριση αναφορών.

2.4.5 Ενοποιημένο σύστημα βάσης δεδομένων

Σε ένα ενοποιημένο σύστημα βάσης δεδομένων περιλαμβάνονται πολλές ξεχωριστές βάσεις δεδομένων, ενώ κάθε μία έχει το δικό της DBMS. Η διαχείρισή της γίνεται από ένα ομοσπονδιακό σύστημα διαχείρισης βάσεων δεδομένων (FDBMS), το οποίο ενσωματώνει τα πολλαπλά αυτόνομα DBMS (Βεσκούκης, 2015).

2.5 Διεπαφή Ανθρώπου - Μηχανής

Με τον όρο διεπαφή, εννοούμε το πρόγραμμα εκείνο το οποίο παρεμβάλλεται μεταξύ χρήστη και υπολογιστή ώστε να καταστήσει δυνατή την αμφίδρομη επικοινωνία μεταξύ τους. Συνεπώς είναι το κομμάτι του Λογισμικού το οποίο ο χρήστης αντιλαμβάνεται με μία ή περισσότερες αισθήσεις και μπορεί να αλληλεπιδράσει μαζί του (Κοκκίνου, 2023).

Ο σωστός σχεδιασμός συστημάτων διεπαφής, αποτελεί σημαντικό κριτήριο για την επιτυχή ανάπτυξη ενός Π.Σ, καθώς η συνολική εμπειρία των χρηστών διαδραματίζει καθοριστικό παράγοντα αποδοχής ενός συστήματος. Ένα ορθά σχεδιασμένο σύστημα διεπαφής, θα πρέπει να βοηθάει τους χρήστες κατά την εκτέλεση των εργασιών τους και να είναι αποδοτικό (Μιαούλης, Μπουσδέκης, & Θεοδωροπούλου, 2021).

Τα σημαντικότερα χαρακτηριστικά που θα πρέπει να διαθέτει ένα τέτοιο σύστημα, είναι τα παρακάτω:

<u>Απλότητα</u>

Με τον όρο απλότητα, αναφερόμαστε στην ευκολία χρήσης και κατανόησης του συστήματος. Το σύστημα θα πρέπει να είναι με τέτοιο τρόπο σχεδιασμένο ώστε να αποκρύπτει την όποια πολυπλοκότητα και να αφαιρεί στοιχεία τα οποία δεν είναι αναγκαία να εμφανίζονται στον χρήστη (Βεσκούκης, 2015). Το παραπάνω μπορεί να επιτευχθεί μέσω οπτικής ιεραρχίας των στοιχείων του συστήματος και την κατάλληλη διάταξή τους, καθώς και τη δυνατότητα χρήσης προεπιλεγμένων τιμών (Φιτσιλής, 2015).

Σαφήνεια

Η σαφήνεια αποτελεί το οπτικό, εννοιολογικό και γλωσσολογικό κομμάτι του συστήματος. Ένα αποτελεσματικό σύστημα θα πρέπει να είναι ξεκάθαρο και εύκολα κατανοητό από τον χρήστη. Τα κείμενα διεπαφής θα πρέπει να είναι απλά, και να μην περιέχουν δυσνόητη τεχνική ορολογία η οποία ενδεχομένως να δυσκολέψει τον χρήστη (Μιαούλης, Μπουσδέκης, & Θεοδωροπούλου, 2021).

Ευελιξία

Η ευελιξία αφορά την ικανότητα ενός συστήματος διεπαφής, να ανταποκρίνεται στις διαφορετικές ανάγκες των χρηστών της. Κάτι τέτοιο μπορεί να επιτευχθεί προσφέροντας πολλαπλούς τρόπους χρήσης διεπαφής, μέσω των οποίων ο χρήστης μπορεί να φτάσει στο ίδιο αποτέλεσμα. Επίσης καλή τακτική αποτελεί και η δυνατότητα προσαρμογής του συστήματος διεπαφής από τον ίδιο τον χρήστη.

Αποκριτικότητα

Η αποκριτικότητα σχετίζεται με την ανταπόκριση του συστήματος στις ενέργειες του χρήστη. Αυτό σημαίνει ότι ο χρήστης θα πρέπει να λαμβάνει ανατροφοδότηση από το σύστημα σχετικά με τις ενέργειες που έχει πραγματοποιήσει. Τέτοια παραδείγματα αποτελούν οι αλλαγές στον κέρσορα βάσει ενεργειών που μπορεί να έχει εκτελέσει, η εμφάνιση κάποιου προειδοποιητικού μηνύματος, η αλλαγή χρωμάτων κάποιων στοιχείων της διεπαφής καθώς και οι ενδείξεις προόδου (Μιαούλης, Μπουσδέκης, & Θεοδωροπούλου, 2021).

Προσβασιμότητα

Η προσβασιμότητα, επιτρέπει στους χρήστες να μπορούν να χρησιμοποιούν το σύστημα όπου και αν βρίσκονται (κινητές συσκευές, υπολογιστές, tablets) και καλύπτει τις διαφορετικές ανάγκες που μπορεί να απαιτούνται (άτομα με ειδικές ανάγκες) (Φιτσιλής, 2015).

Φιλικότητα προς τον χρήστη

Ένα αποδοτικό σύστημα διεπαφής θα πρέπει να είναι φιλικό προς το χρήστη. Ο χρήστης θα πρέπει να είναι σε θέσει να μπορεί να διακρίνει με ποια στοιχεία της διεπαφής μπορεί να αλληλεπιδράσει, πως να εκτελέσει μία ενέργεια και το αναμενόμενο αποτέλεσμά της . Επιπρόσθετα, θα πρέπει να αποφεύγονται πολύπλοκα

στοιχεία διεπαφής, δυσνόητη ορολογία, δυσδιάκριτα κείμενα και κακή επιλογή χρωμάτων, τα οποία μπορεί να κουράζουν και να συγχέουν τον χρήστη (Μιαούλης, Μπουσδέκης, & Θεοδωροπούλου, 2021).

Συνέπεια

Η συνέπεια στον σχεδιασμό του συστήματος, αφορά τα στοιχεία του συστήματος, τα οποία θα πρέπει να είναι ομοιόμορφα στην εμφάνιση, την τοποθέτηση και την συμπεριφορά. Με αυτόν τον τρόπο μειώνεται ο απαιτούμενος χρόνος εκμάθησης και βελτιώνεται η συνολική εμπειρία του χρήστη (Μιαούλης, Μπουσδέκης, & Θεοδωροπούλου, 2021).

Αποδοτικότητα

Ένα σύστημα διεπαφής, οφείλει να είναι αποδοτικό. Για να εξασφαλιστεί το παραπάνω, θα πρέπει ο χρήστης να μπορεί εύκολα και γρήγορα να εκτελέσει τις εργασίες του. Τα στοιχεία πλοήγησης και διεπαφής, θα πρέπει να είναι εύχρηστα, και τα μονοπάτια πλοήγησης όσο το δυνατόν συντομότερα (Φιτσιλής, 2015). Επιπρόσθετα, θα πρέπει η προσοχή του χρήστη να εστιάζεται στα στοιχεία εκείνα τα οποία είναι σημαντικά για την ολοκλήρωση των ενεργειών που θέλει να ολοκληρώσει (Μητάκος, 2015).

ΚΕΦΑΛΑΙΟ 3: Μοντέλα Κύκλου Ζωής Λογισμικού

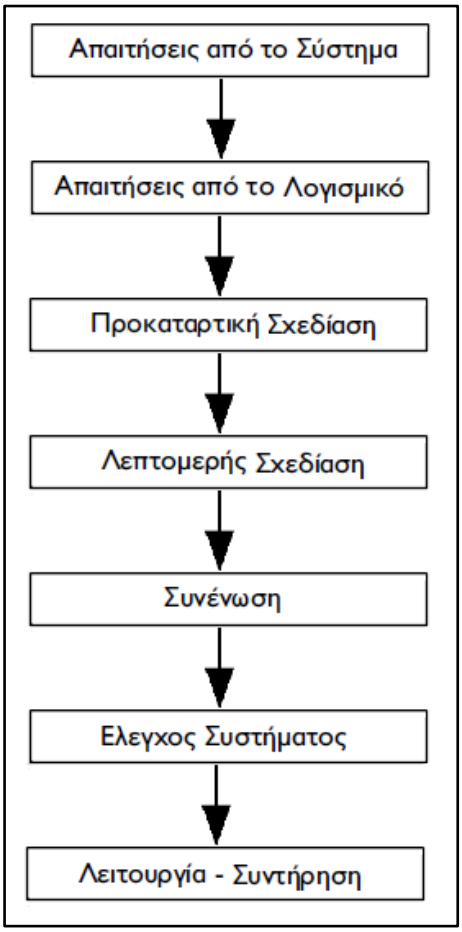
3.1 Εισαγωγικά ζητήματα Μοντέλων Κύκλων Ζωής Λογισμικού

Τα μοντέλα κύκλου ζωής λογισμικού περιγράφουν τις φάσεις από τις οποίες αποτελείται μία εφαρμογή λογισμικού από το αρχικό μέχρι το τελικό του στάδιο καθώς και τις ενέργειες που πραγματοποιούνται σε κάθε ένα από αυτά τα στάδια. Η διαδικασία ανάπτυξης (software process) καθορίζει ποιες ενέργειες πρέπει να εκτελεστούν σε κάθε στάδιο του κύκλου ζωής, ενώ η μεθοδολογία (software development methodology) καθορίζει το πως πρέπει να εκτελεστούν (Βεσκούκης, 2015).

Τα μοντέλα κύκλου ζωής λογισμικού διακρίνονται σε ακολουθιακά και επαναληπτικά. Στα ακολουθιακά μοντέλα η ανάπτυξη του λογισμικού πραγματοποιείται σε διαδοχικές διακριτές φάσεις ενώ στα επαναληπτικά μοντέλα η ανάπτυξη του λογισμικού γίνεται σε τμήματα (Μιαούλης, Μπουσδέκης, & Θεοδωροπούλου, 2021). Στις επόμενες παραγράφους γίνεται μία μικρή αναφορά στα πιο σημαντικά μοντέλα και στον τρόπο με τον οποίο λειτουργούν.

3.1.1 Μοντέλο Καταρράκτη

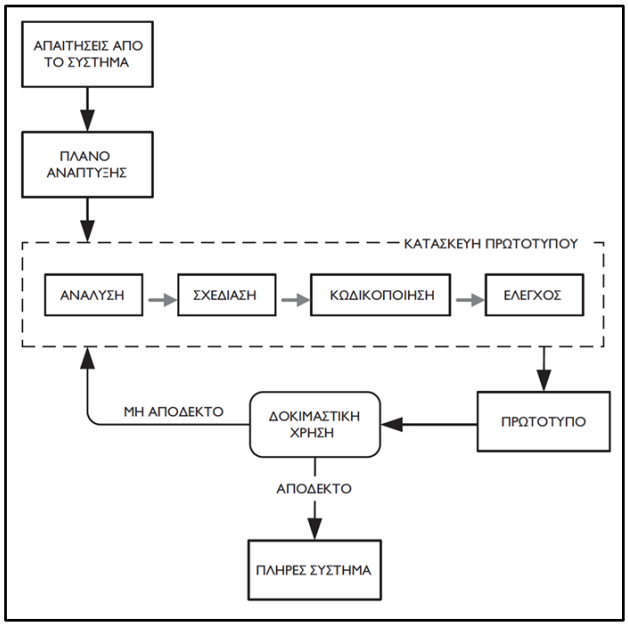
Το Μοντέλο Καταρράκτη (Waterfall Model) αποτελεί ένα ακολουθιακό μοντέλο όπου κάθε επιμέρους διακριτή φάση του λογισμικού διαδέχεται η μια την άλλη με την ολοκλήρωση της προηγούμενης. Κάθε επιμέρους φάση ολοκληρώνεται με μία εργασία επαλήθευσης κατά την οποία αποφασίζεται αν θα γίνει η μετάβαση ή όχι στην επόμενη φάση. Στο πρώτο στάδιο καθορίζονται οι απαιτήσεις του συστήματος και του λογισμικού αντίστοιχα. Στην συνέχεια γίνεται η προκαταρκτική και η λεπτομερής σχεδίαση του λογισμικού. Στην προκαταρτική σχεδίαση καθορίζονται οι διαφορετικές οντότητες του συστήματος καθώς και οι συσχετίσεις μεταξύ τους. Στην λεπτομερή σχεδίαση καθορίζεται η εσωτερική δομή κάθε οντότητας του λογισμικού. Επόμενο στάδιο είναι η συνένωση των οντοτήτων στο σύστημα και τέλος ακολουθεί το βήμα ελέγχου του συστήματος. Στην συνέχεια το λογισμικού ολοκληρώνεται και περνάμε στην φάση της λειτουργίας και της συντήρησης (Βεσκούκης, 2015).



***Εικόνα 1**: Το μοντέλο Καταρράκτη*

3.1.2 Μοντέλο Πρωτοτυποποίησης

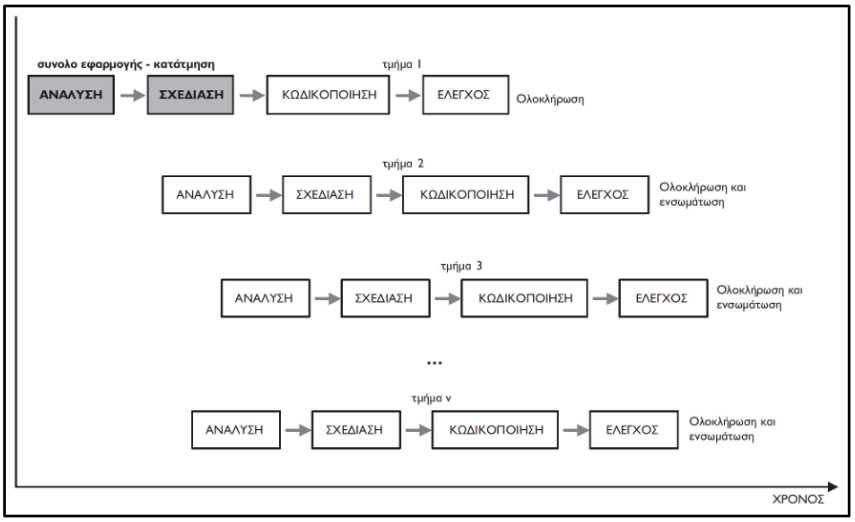
Στο Μοντέλο Πρωτοτυποποίησης (Prototyping Model) έχουμε την ανάπτυξη του λογισμικού σε τμήματα, τα οποία καλούνται πρωτότυπα. Ανήκει στην κατηγορία επαναληπτικών μοντέλων καθώς οι διαδικασίες ανάπτυξης για ένα τμήμα, επαναλαμβάνονται κάθε φορά έως ότου αυτό ολοκληρωθεί. Κάθε πρωτότυπο περιλαμβάνει τις βασικές λειτουργίες οι οποίες δίνονται στον πελάτη για δοκιμασία και στην συνέχεια συλλέγονται πληροφορίες και παρατηρήσεις για την διαδικασία κατασκευής νέου πρωτότυπου. Η διαδικασία συνεχίζεται επαναληπτικά έως ότου το πρωτότυπο ικανοποιεί τις απαιτήσεις και θεωρείται αποδεκτό από τον πελάτη. Με την ίδια λογική γίνεται ανάπτυξη και στα υπόλοιπα τμήματα και το λογισμικό ολοκληρώνεται αφού γίνουν αποδεκτά όλα τα πρωτότυπα (Βεσκούκης, 2015).



Εικόνα 2: *Το μοντέλο Πρωτοτυποποίησης (Βεσκούκης, 2015)*

3.1.3 Μοντέλο Λειτουργικής Επαύξησης

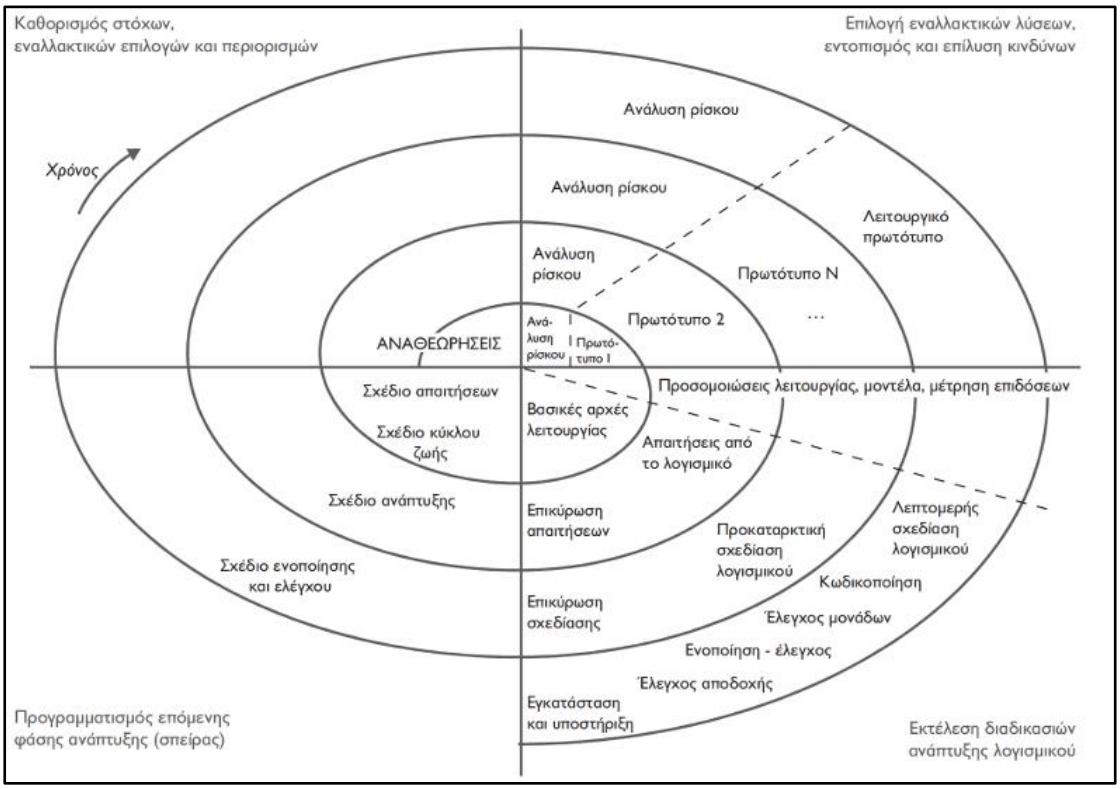
Το Μοντέλο Λειτουργικής Επαύξησης (Incremental Model) είναι ένας συνδυασμός των μοντέλων Καταρράκτη και Πρωτοτυποποίησης. Το λογισμικό χωρίζεται σε τμήματα με βάση το μοντέλο Πρωτοτυποποίησης, τα οποία αναπτύσσονται ανεξάρτητα μεταξύ τους, ενώ η ανάπτυξή τους γίνεται ακολουθιακά με βάση το μοντέλο του Καταρράκτη. Συνεπώς κατά την αρχική φάση έχουμε ανάλυση και σχεδίαση των τμημάτων από των οποίων θα αποτελείται το λογισμικό. Στην συνέχεια η ανάπτυξή τους πραγματοποιείται ανεξάρτητα και παράλληλα και όταν ολοκληρωθεί η ανάπτυξη κάθε τμήματος, ενσωματώνεται στο σύνολο της εφαρμογής.



Εικόνα 3: *Το μοντέλο Λειτουργικής Επαύξησης (Βεσκούκης, 2015)*

3.1.4 Σπειροειδές Μοντέλο

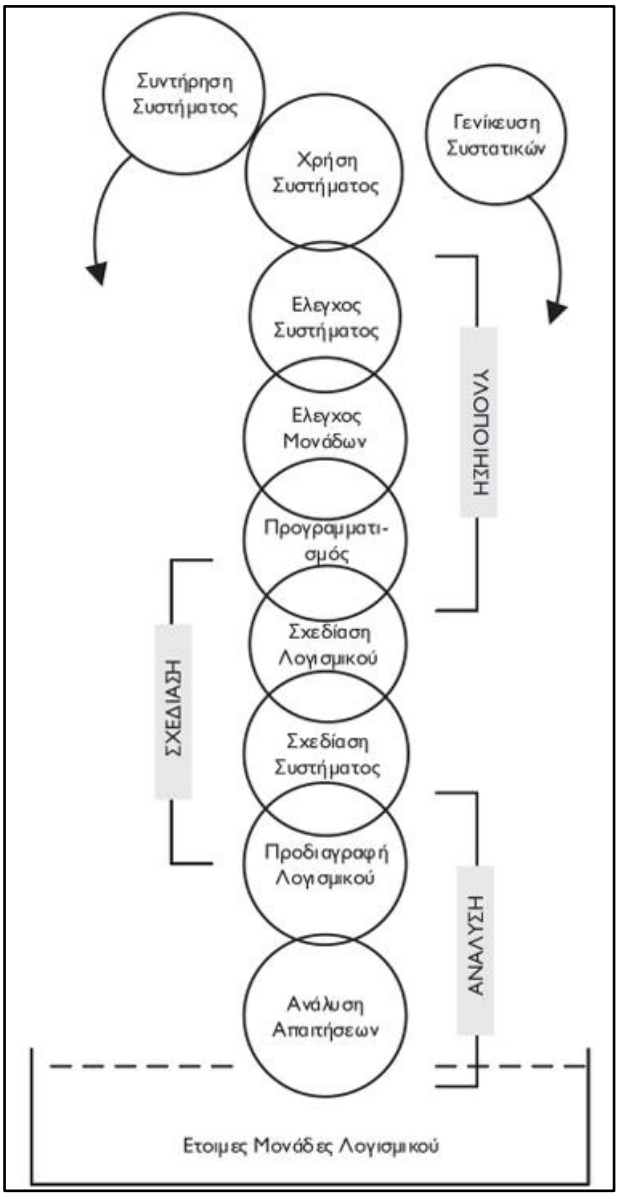
Το Σπειροειδές Μοντέλο (Spiral Model) αποτελεί μία γενίκευση των μοντέλων Λειτουργικής Επαύξησης και Πρωτοτυποποίησης. Οι φάσεις και οι διαδικασίες της ανάπτυξης του λογισμικού δεν είναι προκαθορισμένες από το μοντέλο, αλλά εξειδικεύονται στον εκάστοτε χώρο της εφαρμογής τους. Η ανάπτυξη πραγματοποιείται μέσω πολλών κύκλων όπου κάθε φορά προστίθενται καινούργια χαρακτηριστικά. Κατά την έναρξη κάθε κύκλου, γίνεται ανάλυση κινδύνων από την οποία προκύπτουν οι εργασίες που θα εκτελεστούν μέσα στο κύκλο και ελέγχεται η εφικτότητα εκτέλεσή του (Κοκκίνου, 2023). Μπορούμε να πούμε ότι στο σπειροειδές μοντέλο διακρίνουμε τέσσερεις κύκλους εργασιών. Ο πρώτος κύκλος αφορά των προσδιορισμό των στόχων όπου καθορίζονται οι εργασίες κάθε επανάληψης και οι πιθανοί περιορισμοί του. Ο δεύτερος κύκλος αφορά τον εντοπισμό και την ανάλυση κινδύνου και την αποτίμηση εναλλακτικών λύσεων. Στον τρίτο κύκλο έχουμε την εκτέλεση των διαδικασιών ανάπτυξης του λογισμικού ενώ στον τέταρτο και τελευταίο κύκλο γίνεται προγραμματισμός για την συνέχιση της ανάπτυξης του λογισμικού.



Εικόνα 4: *Το Σπειροειδές μοντέλο (Βεσκούκης, 2015)*

3.1.5 Μοντέλο Πίδακα

Το Μοντέλο Πίδακα (Fountain Model) στηρίζεται στο γεγονός ότι οι έννοιες ανάλυση, σχεδίαση και κωδικοποίηση έρχονται πιο κοντά στην αντικειμενοστρεφή τεχνολογία, και το αποτέλεσμα κάθε διαδικασίας κατασκευής λογισμικού, είναι επαναχρησιμοποιούμενες μονάδες και όχι απλώς ένα σύστημα. Κατά την ανάπτυξη έχουμε επικαλύψεις των φάσεων ανάλυση, σχεδίαση και κωδικοποίηση ενώ στο τέλος της ανάπτυξης ορισμένα συστατικά του λογισμικού που έχουν δημιουργηθεί, ενσωματώνονται και χρησιμοποιούνται για την ανάπτυξη νέων συστημάτων.



Εικόνα 5: *Το μοντέλο Πίδακα (Βεσκούκης, 2015)*

3.2 Κριτήρια επιλογής μοντέλου

Η επιλογή του κατάλληλου μοντέλου κύκλου ζωής ενός Λογισμικού αποτελεί ένα κρίσιμο βήμα κατά την ανάπτυξή του. Τα διάφορα μοντέλα που έχουν αναπτυχθεί, εμφανίζουν διαφορετικά πλεονεκτήματα και μειονεκτήματα όπως αναφέρθηκαν και στις προηγούμενες παραγράφους. Γίνεται αντιληπτό ότι δεν υπάρχει κάποιο συγκεκριμένο μοντέλο το οποίο να είναι καλύτερο από τα υπόλοιπα, αλλά υπάρχουν ορισμένα κριτήρια τα οποία με βάση την εφαρμογή θα μπορούσαν να φανούν χρήσιμα κατά την διαδικασία επιλογής.

Τα βασικότερα κριτήρια επιλογής είναι τα παρακάτω:

- Απαιτήσεις χρηστών
- Εξοικείωση με την τεχνολογία
- Πολυπλοκότητα συστήματος
- Αξιοπιστία συστήματος
- Βραχυπρόθεσμο χρονοδιάγραμμα
- Ορατότητα βαθμού προόδου

Σε περιπτώσεις όπου οι απαιτήσεις έχουν πλήρως καθοριστεί και είναι αμετάβλητες, μπορεί να επιλεγεί το μοντέλο Καταρράκτη. Αν από την άλλη κάτι τέτοιο δεν ισχύει, και οι απαιτήσεις είναι δυναμικές, τότε κατάλληλη επιλογή μπορεί να αποτελέσει ένα μοντέλο επανάληψης ή ένα μοντέλο επαύξησης (Μητάκος, 2015). Σε περιπτώσεις όπου το χρονοδιάγραμμα είναι πιεστικό, τότε το μοντέλο Καταρράκτη δεν ενδείκνυται. Ακόμα, σε περιπτώσεις όπου εμπλέκονται οικονομικά, πολιτικά ή συνδικαλιστικά συμφέροντα, και τα οποία απαιτούν ιδιαίτερες δεξιότητες και τεχνολογίες, είναι πιο εύκολο να χρησιμοποιηθούν μοντέλα αυστηρού προγραμματισμού, όπως είναι και το μοντέλο Καταρράκτη.

ΚΕΦΑΛΑΙΟ 4: Πληροφοριακά Συστήματα στη Βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου

4.1 Ο κλάδος της Αυτοκινητοβιομηχανίας

Η αυτοκινητοβιομηχανία αντιμετωπίζει μια σειρά απαιτήσεων και προκλήσεων λόγω της σύγχρονης και ανταγωνιστικής φύσης του κλάδου και χαρακτηρίζεται από την συνεχή ανάγκη για καινοτομία, ασφάλεια, απόδοση και βελτίωση της ποιότητας.

Οι καινοτομίες σε τεχνολογικό επίπεδο, όπως η ανάπτυξη νέων κινητήρων, συστημάτων ασφαλείας καθώς και η εφαρμογή αυτόνομων τεχνολογιών οδήγησης, απαιτεί από τις εταιρίες να δαπανούν τεράστια ποσά και ανθρώπινο κεφάλαιο ώστε να παραμένουν ανταγωνιστικοί. Επίσης ο κλάδος της αυτοκινητοβιομηχανίας έχει πολύ υψηλές απαιτήσεις σε ό,τι αφορά την ασφάλεια των οχημάτων, όχι μόνο για τους επιβάτες αλλά και για τους πεζούς και τα υπόλοιπα οχήματα στο δρόμο. Μία ακόμα συνιστώσα που διέπει τον κλάδο της αυτοκινητοβιομηχανίας είναι οι ολοένα και μεγαλύτερες απαιτήσεις για την μείωση των εκπομπών και τη χρήση βιώσιμων υλικών και διαδικασιών παραγωγής. Τέλος, οι εταιρίες που απασχολούνται σε αυτόν τον κλάδο, θα πρέπει να διασφαλίσουν υψηλή ποιότητα στα οχήματά τους ώστε να διατηρήσουν την εμπιστοσύνη των καταναλωτών και να ανταποκρίνονται στις αυξημένες απαιτήσεις.

Η αυτοκινητοβιομηχανία έχει υποστεί σημαντικές μεταβολές τα τελευταία χρόνια λόγω της εξέλιξης των πληροφοριακών συστημάτων σε όλες τις φάσεις της αλυσίδας παραγωγής και διανομής. Αυτή η εξέλιξη έχει οδηγήσει σε μια νέα εποχή για τον κλάδο, όπου η τεχνολογία παίζει καθοριστικό ρόλο σε κάθε διαδικασία, από τη σχεδίαση και την παραγωγή έως τη διανομή και την εξυπηρέτηση μετά την πώληση.

4.2 Βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου

Η βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου αποτελεί έναν σημαντικό κομμάτι του βιομηχανικού τομέα, με εφαρμογές που καλύπτουν ένα ευρύ φάσμα βιομηχανικών και καταναλωτικών αναγκών. Τα χυτήρια αλουμινίου, αναλαμβάνουν τη μετατροπή του αλουμινίου σε διάφορα εξαρτήματα, από μικρά μέχρι

μεγάλα, πολύπλοκα ή απλά, που χρησιμοποιούνται σε πολλούς τομείς, ένας εξ αυτών, και ο κλάδος της αυτοκινητοβιομηχανίας.

Η διαδικασία της χύτευσης ξεκινά με το λιώσιμο του αλουμινίου σε μεγάλους ειδικούς φούρνους, και στην συνέχεια τροφοδοτείται στην χυτευτική μηχανή. Εκεί, μέσω ενός καλουπιού, το χύδην αλουμίνιο παίρνει την μορφή του καλουπιού και απομακρύνεται από την μηχανή με την χρήση ρομπότ. Στην συνέχεια, περνάει στην κοπτική μηχανή, όπου υπάρχει ένα ειδικά προσαρμοσμένο καλούπι, το οποίο έχει μαχαίρια, και αφαιρεί τα κομμάτια που δεν χρειάζονται, ώστε να πάρουμε το τελικό εξάρτημα. Μετά από αυτήν την φάση το εξάρτημα είτε είναι έτοιμο προς τελική διαλογή και αποστολή, είτε υπόκεινται σε άλλες κατεργασίες όπως ατσαλοβολή, μηχανοποίηση, έλεγχος διαρροής κ.λπ., ανάλογα με τις απαιτήσεις του πελάτη.

4.3 Διαχείριση Εφοδιαστικής Αλυσίδας στη Βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου

Η Διαχείριση Εφοδιαστικής Αλυσίδας (Supply Chain Management - SCM) περιλαμβάνει τον σχεδιασμό και τη διαχείριση όλων των δραστηριοτήτων που αφορούν την προμήθεια υλικών και πρώτων υλών, καθώς και την παραγωγή και τη διανομή προϊόντων. Επίσης, περιλαμβάνει τη συνεργασία με προμηθευτές και πελάτες. Η διαχείριση της εφοδιαστικής αλυσίδας αφορά τη διαχείριση της ροής των αγαθών προς και από την επιχείρηση και περιλαμβάνει τη ροή και την αποθήκευση των πρώτων υλών, των προϊόντων που βρίσκονται μέσα στην παραγωγική διαδικασία, καθώς και των έτοιμων προϊόντων (Chopra, Sawant, Kodi, & Terkar, 2022).

Η βιομηχανία που εξετάζουμε προμηθεύεται ως πρώτη ύλη χελώνες αλουμινίου οι οποίες χρησιμοποιούνται για την παραγωγή εξαρτημάτων. Η καλή συνεργασία και επικοινωνία με τους προμηθευτές είναι καίριας σημασίας για την ομαλή λειτουργία της εταιρίας και την παραγωγή εξαρτημάτων που να πληρούν τα κριτήρια ποιότητας και ασφάλειας που ορίζονται από τους πελάτες. Επίσης ο σωστός σχεδιασμός και η διαχείριση της αποθήκης αποτελεί σημείο κλειδί για την ελαχιστοποίηση του κόστους και την ικανοποίηση των πελατών, συμβάλλοντας στην συνολική αύξηση της αποδοτικότητας και αποτελεσματικότητας της εταιρίας.

4.4 Διαχείριση Παραγωγής στη Βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου

Κατά την φάση της παραγωγής, τα Π.Σ μπορούν να χρησιμοποιηθούν για τον έλεγχο και την βελτιστοποίηση της γραμμής παραγωγής. Μπορούν να παρέχουν έλεγχο και ανάλυση δεδομένων στη λειτουργία των μηχανημάτων με σκοπό την αποτελεσματική και απρόσκοπτη λειτουργία τους. Με την προγραμματισμένη λειτουργία τους, τα Π.Σ μπορούν να εξασφαλίσουν ότι οι εργασίες εκτελούνται με συνέπεια και σταθερότητα, μειώνοντας έτσι τον κίνδυνο ανθρώπινων λαθών και αυξάνοντας την παραγωγικότητα. Επιπλέον, τα Π.Σ ενσωματώνουν δυνατότητες παρακολούθησης και ανάλυσης δεδομένων, που επιτρέπουν στους διαχειριστές να παρακολουθούν την απόδοση της γραμμής παραγωγής σε πραγματικό χρόνο. Αυτή η δυνατότητα επιτρέπει την άμεση αντίδραση σε ενδεχόμενα προβλήματα ή δυνητικές βλάβες, μειώνοντας τον χρόνο και το κόστος συντήρησης (Sommerville, 2011).

Αρχίζοντας από τον προγραμματισμό της παραγωγής, το λογισμικό επιτρέπει την εύκολη δημιουργία προγραμμάτων παραγωγής με βάση τις ανάγκες και τους πόρους της επιχείρησης. Κατά τη διάρκεια της παραγωγής, το λογισμικό παρέχει μηχανισμούς για την αυτοματοποίηση διαδικασιών, τον έλεγχο της ποιότητας και την αντιμετώπιση ενδεχόμενων προβλημάτων. Επιπλέον, το λογισμικό παρέχει εργαλεία για την παρακολούθηση και τον έλεγχο της παραγωγικότητας με τη βοήθεια αναλυτικών αναφορών και γραφημάτων.

Η αυτοματοποίηση των διαδικασιών αποτελεί κρίσιμο στοιχείο για την επίτευξη αυξημένης αποτελεσματικότητας και ακρίβειας στη διαχείριση παραγωγής. Οι επαναλαμβανόμενες και χρονοβόρες διαδικασίες εκτελούνται με μεγαλύτερη ακρίβεια και ταχύτητα, μειώνοντας τον κίνδυνο λαθών και αποφέροντας ουσιαστικές εξοικονομήσεις σε χρόνο και πόρους. Από την άλλη πλευρά, η επιτυχής υλοποίηση της αυτοματοποίησης στη διαχείριση παραγωγής απαιτεί την ανάλυση και την καταγραφή των υφιστάμενων διαδικασιών και την επιλογή των κατάλληλων τεχνολογιών και εργαλείων λογισμικού. Με τη σωστή στρατηγική και τη σωστή υλοποίηση όμως, η αυτοματοποίηση μπορεί να οδηγήσει σε σημαντικές βελτιώσεις στην απόδοση, την ακρίβεια και την αποτελεσματικότητα της διαχείρισης παραγωγής.

Οι ανάγκες στη διαχείριση παραγωγής διαφέρουν ανάλογα με τον τύπο της παραγωγής και την κλίμακα των εργασιών. Κάθε επιχείρηση λειτουργεί σε ένα μοναδικό περιβάλλον και αντιμετωπίζει διαφορετικές προκλήσεις. Ο τύπος της παραγωγής επηρεάζει τις ανάγκες σε όρους προγραμματισμού, εξοπλισμού, προμηθειών και αποθήκευσης. Η κλίμακα των εργασιών από την άλλη, επηρεάζει την απαίτηση για ακρίβεια, την ανάγκη για αυτοματοποίηση και την αποτελεσματικότητα των διαδικασιών παραγωγής. Επομένως, η κατανόηση του τύπου της παραγωγής και της κλίμακας των εργασιών είναι κρίσιμη για την ανάπτυξη και την εφαρμογή ενός συστήματος που θα προσφέρει τις κατάλληλες λύσεις για τις συγκεκριμένες ανάγκες και απαιτήσεις της επιχείρησης.

Η διαχείριση παραγωγής απαιτεί επίσης στενή συνεργασία και αποτελεσματική επικοινωνία μεταξύ των διαφόρων τμημάτων μιας επιχείρησης. Κάθε τμήμα έχει τον δικό του ρόλο και συμβάλλει στην ολοκληρωμένη διαχείριση της παραγωγής, αλλά η αποτελεσματική λειτουργία απαιτεί συνεργασία και αλληλεπίδραση μεταξύ τους. Για παράδειγμα, το τμήμα αποθήκης πρέπει να επικοινωνεί με το τμήμα παραγωγής για τη διαχείριση των αποθεμάτων. Επίσης, η διαχείριση ποιότητας πρέπει να συνεργάζεται με το τμήμα παραγωγής για την ανίχνευση και την αντιμετώπιση ενδεχόμενων προβλημάτων ποιότητας. Η επιτυχής συνεργασία μεταξύ των διαφόρων τμημάτων απαιτεί ανοικτή και αποτελεσματική επικοινωνία, κοινό προσανατολισμό προς τους κοινούς στόχους της επιχείρησης, και συνεχή συνεννόηση για την αντιμετώπιση των προκλήσεων. Αυτό εξασφαλίζει όχι μόνο την αποτελεσματική λειτουργία της παραγωγής, αλλά και τη συνολική επιτυχία και ανταγωνιστικότητα της επιχείρησης στην αγορά.

Στην βιομηχανία που εξετάζουμε η διαχείριση της παραγωγής αφορά την κατασκευή εξαρτημάτων τα οποία να πληρούν τις απαιτήσεις του πελάτη. Η εταιρία θα πρέπει να είναι σε θέση να έχει έλεγχο της παραγωγής της με το ελάχιστο δυνατό κόστος. Επίσης θα πρέπει να υπάρχει αμφίδρομη επικοινωνία με όλα τα εμπλεκόμενα μέρη ούτως ώστε να αντιμετωπίζονται έγκαιρα τα όποια προβλήματα μπορεί να προκύψουν. Ακόμα η σωστή διαχείριση της συντήρησης των μηχανών αποτελεί σημαντικό παράγοντα για την ομαλή και απρόσκοπτη λειτουργία της επιχείρησης. Τέλος η κατάλληλη εκπαίδευση των εργαζομένων συμβάλλει στην μείωση ενδεχόμενων σφαλμάτων και στην αύξηση της αποδοτικότητας.

ΚΕΦΑΛΑΙΟ 5: Ανάπτυξη λογισμικού διαχείρισης παραγωγής σε βιομηχανία παραγωγής χυτών εξαρτημάτων αλουμινίου

Για την ανάπτυξη του λογισμικού χρησιμοποιήθηκε το μοντέλο κύκλου ζωής του Καταρράκτη, που παρουσιάστηκε στην ενότητα 3.3.1. Συγκεκριμένα ακολουθήσαμε την ανάλυση απαιτήσεων, τη σχεδίαση, την υλοποίηση, και τη δοκιμή (ή έλεγχο). Οι φάσεις της εγκατάστασης και συντήρησης δεν είναι υλοποιήσιμες στο πλαίσιο της διπλωματικής εργασίας. Στη συνέχεια αναλύεται κάθε μία από τις φάσεις που υλοποιήθηκαν.

5.1 Ανάλυση απαιτήσεων

Κατά την φάση της ανάλυσης των απαιτήσεων για την εκτεταμένη συλλογή πληροφοριών, έγινε μελέτη εγγράφων, αναφορών, και σημειώσεων εργασίας που παράγονται από τα υπάρχοντα συστήματα, και παρατήρηση του τρόπου λειτουργίας αυτών των συστημάτων. Επίσης πραγματοποιήθηκε υποβολή ερωτηματολογίων στους χρήστες και διενέργεια συνεντεύξεων. Κατά τη διάρκεια αυτού του σταδίου, όλες οι πληροφορίες που συγκεντρώθηκαν, αξιοποιήθηκαν για τον καθορισμό των αναγκών.

Η εταιρία είναι προμηθευτής χυτών εξαρτημάτων αλουμινίου για την αυτοκινητοβιομηχανία. Ο τρόπος που λειτουργεί είναι ο εξής:

- Ζήτηση εξαρτημάτων από πελάτες σε ετήσια βάση με αποστολές κάθε μήνα, τρίμηνο ή εξάμηνο.
- Παραγωγή εξαρτημάτων με χυτόπρεσσες
- Μηχανοποίηση εξαρτημάτων με CNC αν απαιτείται
- Διάφορες άλλες κατεργασίες όπως ατσαλοβολή, έλεγχος διαρροής κλπ.
- Ποιοτικός έλεγχος εξαρτημάτων
- Τελική διαλογή και αποθήκευση
- Αποστολή προς τον πελάτη

Η διαδικασία ξεκινάει με την παραγγελιοληψία όπου καταγράφεται η απαιτούμενη ποσότητα και ο κωδικός παραγγελίας. Στην συνέχεια με βάση την ποσότητα της παραγγελίας δεσμεύεται από την αποθήκη η αντίστοιχη ποσότητα που απαιτείται και καταχωρείται ένας κωδικός παραγωγής. Έπειτα κατά την διάρκεια της παραγωγής, ο χειριστής τραβάει τον αντίστοιχο κωδικό παραγγελίας και κωδικό παραγωγής ώστε να αποδεσμευτούν οι ποσότητες από τις παραγγελίες και από την αποθήκη αντίστοιχα. Μετά την διαδικασία της χύτευσης, το εξάρτημα είτε αποστέλλεται στον πελάτη μετά από τελική διαλογή και συσκευασία, είτε υπόκεινται σε περαιτέρω κατεργασίες όπως μηχανοποίηση, ατσαλοβολή κλπ και μετά αποστέλλεται.

Η επιχείρηση λειτουργεί με 3 βάρδιες (06-14, 14-22, 22-06). Ο κάθε χειριστής βρίσκεται σε μία συγκεκριμένη μηχανή χύτευσης και παρακολουθεί την ροή της μηχανής. Στο τέλος της βάρδιας καταγράφει τα πατήματα (πάτημα είναι ένας κύκλος της μηχανής όπου τροφοδοτείται σε ένα καλούπι υγρό μέταλλο και παίρνει την μορφή του καλουπιού, δημιουργώντας το εξάρτημα) και τον αριθμό των σκάρτων.

Η αποθήκη προμηθεύεται πρώτες ύλες (χελώνες αλουμινίου) και δευτερεύοντες ύλες (oring, ένθετα, αισθητήρες). Επίσης χρησιμοποιείται και για την αποθήκευση ημιέτοιμων και έτοιμων προϊόντων.

Στην εταιρία πραγματοποιούνται εσωτερικές εκπαιδεύσεις τόσο από εξωτερικούς συνεργάτες όσο και από εσωτερικά στελέχη και καταχωρούνται στο σύστημα.

Τα διάφορα είδη μηχανών καταχωρούνται στο σύστημα και υπάρχει ιστορικό συντηρήσεων και επισκευών κάθε μηχανής.

Οι προμηθευτές αξιολογούνται από την επιχείρηση και υπάρχει επίσης αρχείο με τις πιστοποιήσεις τους (ISO 9001, ISO 14001).

Οι απαιτήσεις κατηγοριοποιήθηκαν σε απαιτήσεις δεδομένων, απαιτήσεις διασύνδεσης (ή διεπαφής) με τον χρήστη, λειτουργικές απαιτήσεις και μη λειτουργικές απαιτήσεις. Σε κάθε κατηγορία καταγράφονται οι ερωτήσεις που χρησιμοποιήθηκαν για τη συλλογή απαιτήσεων και ακολουθεί η περιγραφή των απαιτήσεων.

5.1.1 Καταγραφή Απαιτήσεων δεδομένων

Η επιχείρηση αλληλεπιδρά με οντότητες οι οποίες βρίσκονται στο εσωτερικό της, αλλά και στο εξωτερικό της περιβάλλον. Οι διαφορετικές αυτές οντότητες, πρέπει να συνεργάζονται αρμονικά, ώστε η επιχείρηση να μπορεί να πετυχαίνει τους στόχους της.

Η σχέση μεταξύ επιχείρησης και προμηθευτών είναι σημαντική, καθώς οι προμηθευτές μπορούν να επηρεάσουν τις τιμές των τελικών προϊόντων, την ποιότητά τους αλλά και τον χρόνο παράδοσης. Οι πελάτες έχουν και αυτοί με την σειρά τους διαπραγματευτική δύναμη, και μπορούν να επηρεάσουν την ζήτηση των προϊόντων. Οι εργαζόμενοι παρέχουν υπηρεσίες στην επιχείρηση και θα πρέπει να είναι εκπαιδευμένοι και κατάλληλα καταρτισμένοι ώστε να είναι αποδοτικοί. Τέλος τα υλικά που βρίσκονται στην Αποθήκη πρέπει να καλύπτουν τις προδιαγραφές ασφάλειας και λειτουργικότητας με βάση τις εκάστοτε απαιτήσεις.

Ερ.1: *Ποια στοιχεία χρειάζεται να καταγράψουμε για την Αποθήκη;*

Αποθήκη

Υλικό, Κωδικός υλικού, Μονάδα μέτρησης, Διαθέσιμη ποσότητα, Ελάχιστη ποσότητα

Ερ.2: *Ποια στοιχεία χρειάζεται να καταγράψουμε για τις Εκπαιδεύσεις;*

Εκπαιδεύσεις

Τίτλος, Κωδικός, Εκπαιδευτής, Ημερομηνία

Ερ.3: *Ποια στοιχεία χρειάζεται να καταγράψουμε για τις Εξαρτήματα;*

Εξαρτήματα

Κωδικός εξαρτήματος, Κράμα, Κοιλότητες, Βάρος χυτού, Βάρος πατήματος, Καλούπι χύτευσης, Κοπτικό καλούπι

Ερ.4: *Ποια στοιχεία χρειάζεται να καταγράψουμε για τους εργαζόμενους;*

Εργαζόμενοι

Κωδικός, Ονοματεπώνυμο, Θέση, Ημερομηνία πρόσληψης

Ερ.5: *Ποια στοιχεία χρειάζεται να καταγράψουμε για τις Μηχανές;*

Μηχανές

Κωδικός, Ονομασία, Μηχανή

Ερ.6: *Ποια στοιχεία χρειάζεται να καταγράψουμε για τις Παραγγελίες;*

Παραγγελίες

Πελάτης, Εξάρτημα, Τεμάχια, Κωδικός παραγγελίας, Παρτίδα, Ημερομηνία

Ερ.7: *Ποια στοιχεία χρειάζεται να καταγράψουμε για την Παραγωγή;*

Παραγωγή

Κωδικός παραγωγής, Κωδικός παραγγελίας, Ώρα έναρξης, Ώρα λήξης, Τεμάχια, Σκάρτα, Ημερομηνία

Ερ.8: *Ποια στοιχεία χρειάζεται να καταγράψουμε για τους Πελάτες;*

Πελάτες

Πελάτης, Κωδικός πελάτη, Έδρα

Ερ.9: *Ποια στοιχεία χρειάζεται να καταγράψουμε για τους Προμηθευτές;*

Προμηθευτές

Επωνυμία, Κωδικός, Προϊόντα, Email

Ερ.10: *Ποια στοιχεία χρειάζεται να καταγράψουμε για την Συντήρηση καλουπιών;*

Συντήρηση καλουπιών

Κωδικός καλουπιού, Είδος επισκευής, Ημερομηνία

Ερ.11: *Ποια στοιχεία χρειάζεται να καταγράψουμε για την Συντήρηση μηχανών;*

Συντήρηση μηχανών

Μηχανή, Είδος επισκευής, Ημερομηνία

5.1.2 Καταγραφή Απαιτήσεων διασύνδεσης με το χρήστη

Ερ.12: *Ποιες απαιτήσεις υπάρχουν για τη διασύνδεση με το χρήστη;*

- Οθόνη εισόδου χρήστη
- Οθόνη κυρίως μενού
- Οθόνη για την διαχείριση της αποθήκης
- Οθόνη για την διαχείριση της παραγωγής
- Οθόνη για την διαχείριση των παραγγελιών

- Οθόνη για την διαχείριση των εργαζομένων
- Οθόνη για την διαχείριση των προμηθευτών
- Οθόνη για την διαχείριση των συντηρήσεων των καλουπιών
- Οθόνη για την διαχείριση συντηρήσεων των μηχανών
- Οθόνη για την διαχείριση της χρήσης των υλικών
- Οθόνη για την διαχείριση των εκπαιδεύσεων
- Οθόνη για την παροχή πληροφοριών των εξαρτημάτων
- Οθόνη για την παροχή πληροφοριών για τους πελάτες
- Οθόνη για την παροχή πληροφοριών για τις μηχανές

5.1.3 Καταγραφή Λειτουργικών Απαιτήσεων

Ερ.13 Ποιες είναι οι λειτουργικές απαιτήσεις του λογισμικού;

Αποθήκη

- Ο χρήστης θα μπορεί να επιλέξει υλικό, κωδικό υλικού, μονάδα μέτρησης, διαθέσιμη ποσότητα, ελάχιστη ποσότητα και κωδικό προμηθευτή
- Ο χρήστης θα μπορεί να δημιουργήσει μία νέα εγγραφή προσθέτοντας ένα νέο υλικό
- Ο χρήστης θα μπορεί να διαγράψει μία εγγραφή αφαιρώντας ένα υλικό
- Ο χρήστης θα μπορεί να τροποποιήσει μία ήδη υπάρχουσα εγγραφή ενός υλικού

Εκπαιδεύσεις

- Ο χρήστης θα μπορεί να επιλέξει τίτλο και κωδικό εκπαίδευσης, εκπαιδευτή και ημερομηνία
- Ο χρήστης θα μπορεί να δημιουργήσει μία νέα εγγραφή για νέα εκπαίδευση
- Ο χρήστης θα μπορεί να διαγράψει μία εγγραφή αφαιρώντας μία εκπαίδευση
- Ο χρήστης θα μπορεί να τροποποιήσει μία ήδη υπάρχουσα εγγραφή μιας εκπαίδευσης

- Ο χρήστης θα μπορεί να φιλτράρει τις εγγραφές του με βάση τον κωδικό εκπαίδευσης ή την ημερομηνία
- Ο χρήστης θα μπορεί να δει για μία εκπαίδευση ποιοι ήταν οι συμμετέχοντες

Εξαρτήματα

- Ο χρήστης θα μπορεί να δει πληροφορίες για ένα εξάρτημα οι οποίες θα είναι ο κωδικός του εξαρτήματος, το κράμα, τον αριθμό των κοιλοτήτων, το βάρος του χυτού αλλά και του πατήματος, και τον κωδικό του καλουπιού χύτευσης και του κοπτικού καλουπιού
- Ο χρήστης θα μπορεί να βρίσκει πληκτρολογώντας το εξάρτημα, σε ποια μηχανή παράγεται
- Ο χρήστης θα μπορεί να δει μια προεπισκόπηση του εξαρτήματος

Εργαζόμενοι

- Ο χρήστης θα μπορεί να δει πληροφορίες για τους εργαζομένους οι οποίες περιλαμβάνουν το ονοματεπώνυμο, την θέση στην εταιρία και την ημερομηνία πρόσληψης
- Ο χρήστης θα μπορεί να βρίσκει ποιες εκπαιδεύσεις έχει παρακολουθήσει ένας εργαζόμενος

Μηχανές

- Ο χρήστης θα μπορεί να δει πληροφορίες για τις μηχανές οι οποίες θα είναι ο κωδικός της, η ονομασία της και το είδος της μηχανής

Παραγγελίες

- Ο χρήστης θα μπορεί να επιλέξει πελάτη, εξάρτημα, τεμάχια, κωδικό παραγγελίας, παρτίδα και ημερομηνία
- Ο χρήστης θα μπορεί να δημιουργήσει μία νέα εγγραφή παραγγελίας
- Ο χρήστης θα μπορεί να διαγράψει μία εγγραφή αφαιρώντας μία παραγγελία
- Ο χρήστης θα μπορεί να τροποποιήσει μία ήδη υπάρχουσα εγγραφή μιας παραγγελίας

- Ο χρήστης θα μπορεί να φιλτράρει τις εγγραφές του με βάση το όνομα του πελάτη ή του εξαρτήματος, της παρτίδας αλλά και της ημερομηνίας
- Ο χρήστης θα μπορεί να δει διάγραμμα πίτας για τις παραγγελίες ανά πελάτη

Παραγωγή

- Ο χρήστης θα μπορεί να επιλέξει κωδικό παραγωγής, κωδικό παραγγελίας, μηχανή, χειριστή, ώρα έναρξης και ώρα λήξης, τεμάχια, σκάρτα και ημερομηνία
- Ο χρήστης θα μπορεί να δημιουργήσει μία νέα εγγραφή παραγωγής
- Ο χρήστης θα μπορεί να διαγράψει μία εγγραφή αφαιρώντας μία παραγωγή
- Ο χρήστης θα μπορεί να τροποποιήσει μία ήδη υπάρχουσα εγγραφή μιας παραγωγής
- Ο χρήστης θα μπορεί να φιλτράρει τις εγγραφές του με βάση τον κωδικό παραγωγής ή το κωδικό παραγγελίας, την μηχανή και την ημερομηνία
- Ο χρήστης θα μπορεί να δει διάγραμμα πίτας για την παραγωγή ανά μηχανή
- Ο χρήστης θα μπορεί να δει διάγραμμα πίτας για το συνολικό ποσοστό των σκάρτων
- Ο χρήστης θα μπορεί να δει ιστόγραμμα της ετήσιας παραγωγής

Πελάτες

- Ο χρήστης θα μπορεί να δει πληροφορίες για τους πελάτες οι οποίες θα είναι το όνομα του πελάτη, ο κωδικός του καθώς και η έδρα του

Προμηθευτές

- Ο χρήστης θα μπορεί να δει πληροφορίες για τους προμηθευτές οι οποίες θα είναι η επωνυμία και ο κωδικός του, τα προϊόντα του και η ηλεκτρονική του διεύθυνση
- Ο χρήστης θα μπορεί να δίνει παραγγελία ενός υλικού στον εκάστοτε προμηθευτή του

Συντήρηση καλουπιών

- Ο χρήστης θα μπορεί να επιλέξει κωδικό καλουπιού, είδος επισκευής, όνομα συντηρητή και ημερομηνία
- Ο χρήστης θα μπορεί να δημιουργήσει μία νέα εγγραφή
- Ο χρήστης θα μπορεί να διαγράψει μία εγγραφή συντήρησης καλουπιού
- Ο χρήστης θα μπορεί να τροποποιήσει μία ήδη υπάρχουσα εγγραφή μιας συντήρησης καλουπιού
- Ο χρήστης θα μπορεί να φιλτράρει τις εγγραφές του κατά κωδικό καλουπιού, όνομα συντηρητή και ημερομηνία
- Ο χρήστης θα μπορεί να δει ραβδόγραμμα του αριθμό επισκευών που έχει πραγματοποιηθεί ανά συντηρητή

Συντήρηση μηχανών

- Ο χρήστης θα μπορεί να επιλέξει μηχανή, είδος επισκευής, συντηρητή και ημερομηνία
- Ο χρήστης θα μπορεί να δημιουργήσει μία νέα εγγραφή για νέα συντήρηση μηχανής
- Ο χρήστης θα μπορεί να διαγράψει μία εγγραφή αφαιρώντας μία συντήρηση μηχανής
- Ο χρήστης θα μπορεί να τροποποιήσει μία ήδη υπάρχουσα εγγραφή μιας συντήρησης μηχανής
- Ο χρήστης θα μπορεί να φιλτράρει τις εγγραφές του κατά μηχανή, όνομα συντηρητή και ημερομηνία
- Ο χρήστης θα μπορεί να δει ραβδόγραμμα του αριθμό επισκευών που έχει πραγματοποιηθεί ανά συντηρητή
- Ο χρήστης θα μπορεί να δει διάγραμμα πίτας του ποσοστού επισκευών ανά μηχανή που έχουν πραγματοποιηθεί

Χρήση υλικών

- Ο χρήστης θα μπορεί να επιλέξει κωδικό παραγωγής, κωδικό υλικού και ποσότητα

- Ο χρήστης θα μπορεί να δημιουργήσει μία νέα εγγραφή για νέα χρήση υλικού
- Ο χρήστης θα μπορεί να διαγράψει μία εγγραφή αφαιρώντας μία χρήση ενός υλικού
- Ο χρήστης θα μπορεί να τροποποιήσει μία ήδη υπάρχουσα εγγραφή μιας χρήσης υλικού
- Ο χρήστης θα μπορεί να φιλτράρει τις εγγραφές του με βάση τον κωδικό παραγωγής ή το κωδικό υλικού

5.1.4 Καταγραφή Μη Λειτουργικών Απαιτήσεων

Ερ.14: *Ποια ποιοτικά χαρακτηριστικά απαιτούνται για το σύστημα ;*

- Απλότητα
- Σαφήνεια
- Ευελιξία
- Αποκριτικότητα
- Προσβασιμότητα
- Φιλικότητα προς τον χρήστη
- Συνέπεια
- Αποδοτικότητα

Ερ.15: *Ποια άλλα χαρακτηριστικά απαιτούνται για το σύστημα*

Απαιτήσεις Λογισμικού

- Λειτουργικό σύστημα Windows 10 ή μεταγενέστερες εκδόσεις
- Η εφαρμογή πρέπει να είναι αναπτυγμένη σε Python 3.8 ή νεότερη έκδοση

Απαιτήσεις υλικού

- Τετραπύρηνος επεξεργαστής ή καλύτερος (π.χ., Intel i5, AMD Ryzen 5)
- Τουλάχιστον 8GB RAM μνήμη
- Χώρος αποθήκευσης τουλάχιστον 256GB SSD

- Δίκτυο Ethernet 1Gbps για ενσύρματη σύνδεση και Wi-Fi 802.11ac για ασύρματη σύνδεση

Απαιτήσεις ασφαλείας

- Χρήση firewall για προστασία από εξωτερικές απειλές

5.2 Σχεδίαση Λογισμικού

5.2.1. Εννοιολογική Σχεδίαση

Στο στάδιο της εννοιολογικής σχεδίασης, δημιουργείται το εννοιολογικό σχήμα της βάσης δεδομένων, το οποίο εκφράζεται μέσω του Διαγράμματος Οντοτήτων – Συσχετίσεων που αποτελεί μια διαγραμματική αναπαράσταση της δομής της βάσης δεδομένων. Το Διάγραμμα Οντοτήτων - Συσχετίσεων περιέχει τις έννοιες οντότητα, γνώρισμα και συσχέτιση οι οποίες συμβολίζονται με γεωμετρικά σχήματα που συνδέονται μεταξύ τους με γραμμές (Κεχρής, 2015). Η συσχέτιση αποτελεί μία σύνδεση μεταξύ δύο ή περισσότερων οντοτήτων. Στην σύνδεση ορίζεται ο αριθμός των οντοτήτων (βαθμός συσχέτισης) αλλά και των στιγμιοτύπων (πληθικότητα) από κάθε οντότητα που συμμετέχει.

Οι οντότητες που καθορίστηκαν από την ανάλυση απαιτήσεων είναι οι εξής:

Αποθήκη, Εκπαιδεύσεις, Εξαρτήματα, Εργαζόμενοι, Μηχανές, Παραγγελίες, Παραγωγή, Πελάτες, Προμηθευτές, Συντήρηση Καλουπιών, Συντήρηση Μηχανών.

Επίσης καθορίστηκαν τα γνωρίσματα κάθε οντότητας ως εξής:

Αποθήκη

Υλικό, Κωδικός υλικού, Μονάδα μέτρησης, Διαθέσιμη ποσότητα, Ελάχιστη ποσότητα

Εκπαιδεύσεις

Τίτλος, Κωδικός, Εκπαιδευτής, Ημερομηνία

Εξαρτήματα

Κωδικός εξαρτήματος, Κράμα, Κοιλότητες, Βάρος χυτού, Βάρος πατήματος, Καλούπι χύτευσης, Κοπτικό καλούπι

Εργαζόμενοι

Κωδικός, Ονοματεπώνυμο, Θέση, Ημερομηνία πρόσληψης

Μηχανές

Κωδικός, Ονομασία, Μηχανή

Παραγγελίες

Πελάτης, Εξάρτημα, Τεμάχια, Κωδικός παραγγελίας, Παρτίδα, Ημερομηνία

Παραγωγή

Κωδικός παραγωγής, Κωδικός παραγγελίας, Ώρα έναρξης, Ώρα λήξης, Τεμάχια, Σκάρτα, Ημερομηνία

Πελάτες

Πελάτης, Κωδικός πελάτη, Έδρα

Προμηθευτές

Επωνυμία, Κωδικός, Προϊόντα, Email

Συντήρηση καλουπιών

Κωδικός καλουπιού, Είδος επισκευής, Ημερομηνία

Συντήρηση μηχανών

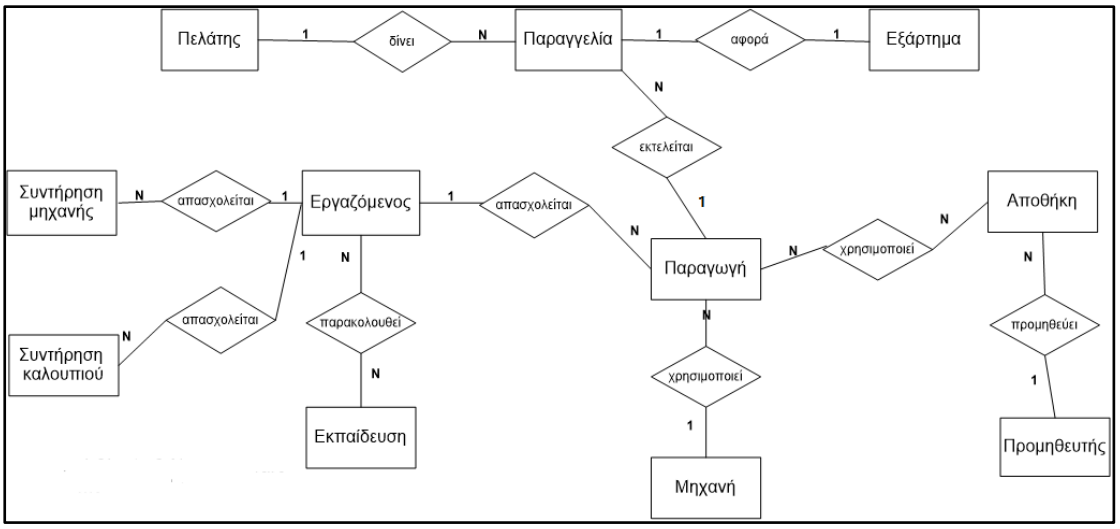
Μηχανή, Είδος επισκευής, Ημερομηνία

Στη συνέχεια καθορίστηκαν οι περιορισμοί των συσχετίσεων μεταξύ των οντοτήτων:

- Κάθε Πελάτης δίνει πολλές Παραγγελίες.
- Κάθε Παραγγελία δίνεται από έναν Πελάτη.
- Κάθε Παραγγελία αφορά ένα Εξάρτημα.
- Κάθε Εξάρτημα αφορά μια Παραγγελία.
- Κάθε Παραγγελία εκτελείται μια φορά στην Παραγωγή
- Σε κάθε Παραγωγή εκτελούνται πολλές παραγγελίες

- Κάθε εργαζόμενος απασχολείται σε πολλές Συντηρήσεις Μηχανής
- Σε κάθε Συντήρηση μηχανής απασχολείται ένας εργαζόμενος
- Κάθε εργαζόμενος απασχολείται σε πολλές Συντηρήσεις Καλουπιού
- Σε κάθε Συντήρηση Καλουπιού απασχολείται ένας εργαζόμενος
- Κάθε εργαζόμενος απασχολείται σε πολλές Παραγωγές
- Σε κάθε Παραγωγή απασχολείται ένας εργαζόμενος
- Κάθε εργαζόμενος εκπαιδεύεται σε πολλές Εκπαιδεύσεις
- Σε κάθε Εκπαίδευση εκπαιδεύονται πολλοί εργαζόμενοι
- Κάθε υλικό της Αποθήκης χρησιμοποιείται σε πολλές Παραγωγές
- Κάθε Παραγωγή χρησιμοποιεί πολλά υλικά της Αποθήκης
- Κάθε Προμηθευτής προμηθεύει πολλά υλικά Αποθήκης
- Κάθε υλικό της Αποθήκης προμηθεύεται από έναν Προμηθευτής

Με βάση τα παραπάνω σχεδιάστηκε το Διάγραμμα Οντοτήτων-Συσχετίσεων που απεικονίζεται στην Εικόνα 6.

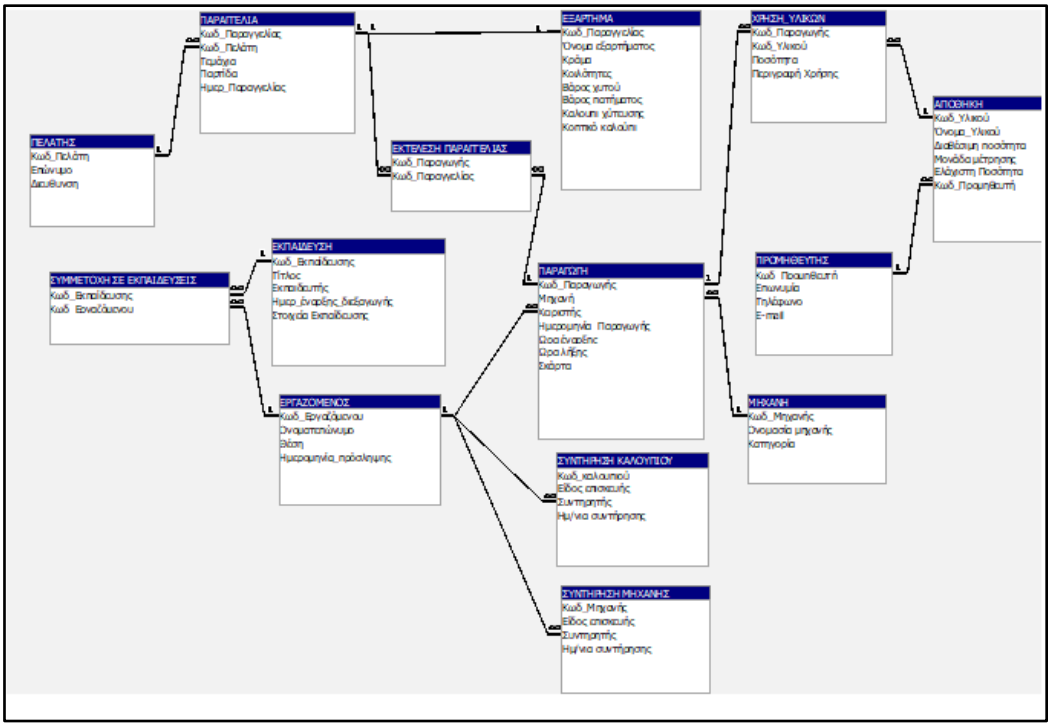


Εικόνα 6: *Διάγραμμα Οντοτήτων – Συσχετίσεων*

5.2.2. Λογική Σχεδίαση Βάσης Δεδομένων

Το Διάγραμμα Οντοτήτων-Συσχετίσεων στη συνέχεια μετατράπηκε σε Σχήμα Βάσης Δεδομένων, σύμφωνα με τη διαδικασία των έξι βημάτων που περιγράφεται από τον Κεχρή (2015).

Στην Εικόνα 7, βλέπουμε το Σχήμα Βάσης Δεδομένων που προέκυψε και στη συνέχεια χρησιμοποιήθηκε ως βάση για το στάδιο της φυσικής σχεδίασης της Βάσης Δεδομένων.



Εικόνα 7: *Σχήμα Βάσης Δεδομένων*

5.3 Υλοποίηση Λογισμικού

Η υλοποίηση του Λογισμικού έγινε με την γλώσσα προγραμματισμού Python. Η Python είναι μια γλώσσα υψηλού επιπέδου, το οποίο σημαίνει ότι είναι ευανάγνωστη και κατανοητή, καθιστώντας την ιδανική για την ανάπτυξη λογισμικού που θα διαχειρίζεται πολύπλοκες λειτουργίες και δεδομένα.

Επιπλέον, η Python προσφέρει ευρεία γκάμα βιβλιοθηκών που μπορούν να χρησιμοποιηθούν για την επεξεργασία δεδομένων, την ανάλυση, την απεικόνιση και την αποθήκευση, καθιστώντας την ιδανική επιλογή για τη διαχείριση πληροφοριών.

Επιπρόσθετα, υποστηρίζει πολλά στιλ προγραμματισμού, συμπεριλαμβανομένου του αντικειμενοστραφούς προγραμματισμού, του λειτουργικού προγραμματισμού και του δομημένου προγραμματισμού. Αυτό την καθιστά κατάλληλη για την ανάπτυξη λογισμικού που πρέπει να προσαρμοστεί σε διάφορες απαιτήσεις και περιβάλλοντα.

Ένα ακόμα σημαντικό πλεονέκτημα αποτελεί το γεγονός ότι είναι ανοικτού κώδικα, συνεπώς δεν απαιτείται η αγορά κάποιας άδειας χρήσης και δικαιωμάτων.

Στη φάση της φυσικής σχεδίασης της βάσης δεδομένων πραγματοποιήθηκαν τροποποιήσεις του σχήματος της βάσης δεδομένων που προέκυψε στη φάση της λογικής σχεδίασης (εικόνα 7), ώστε να απλουστευτεί και τελικά να κωδικοποιηθεί με βάση την αποδοτικότητα.

Η υλοποίηση του πηγαίου κώδικα χρειάστηκε πολλές ώρες ανάπτυξης και αρκετές τροποποιήσεις μέχρι να φτάσει στην τελική του μορφή που περιλαμβάνεται στο Παράρτημα 2.

5.4 Δοκιμή λογισμικού

Μετά την υλοποίηση του πηγαίου κώδικα προχωρήσαμε στην δοκιμή του Λογισμικού. Ο σκοπός των παραπάνω ήταν να διαπιστωθεί σε ποιο βαθμό το νέο σύστημα ικανοποιεί τους αρχικούς στόχους του και κατά πόσο χρειάζονται αναθεωρήσεις ή μετατροπές. Κατά την δοκιμή, προέκυψαν ορισμένα ζητήματα τα οποία και αναλύονται στις επόμενες παραγράφους.

Αρχικά στο παράθυρο της Αποθήκης ανακαλύψαμε ότι όταν επεξεργαζόμασταν μία ήδη υπάρχουσα γραμμή και αποθηκεύαμε τα νέα δεδομένα μας, τότε δημιουργούνταν μία νέα γραμμή με τα δεδομένα που επεξεργαστήκαμε ενώ εξακολουθούσε να υπάρχει και η προηγούμενη. Το πρόβλημα εντοπίστηκε στο σημείο του κώδικα όπου ανανεώνονταν τα δεδομένα στην βάση δεδομένων και επιδιορθώθηκε.

Ένα ακόμα πρόβλημα εντοπίστηκε στο παράθυρο της Παραγωγής. Όταν πιέζαμε το κουμπί της αποθήκευσης δύο φορές, τότε είχαμε διπλό περιεχόμενο. Αυτό συνέβαινε διότι όταν εκτελούνταν ο κώδικας που αφορούσε την αποθήκευση δεδομένων, δημιουργούσε από την αρχή μία βάση δεδομένων για το συγκεκριμένο παράθυρο αντί να την τροποποιεί. Τροποποιώντας τις γραμμές που αφορούσαν την αποθήκευση των δεδομένων, το πρόβλημα εξαλείφθηκε.

Μία ακόμα περίπτωση που εντοπίστηκε να μην λειτουργεί σωστά, ήταν η σύνδεση της Αποθήκης με την Χρήση υλικών. Ο στόχος ήταν, η ποσότητα για ένα υλικό το οποίο βρίσκονταν στην Αποθήκη, να αντανακλούσε στο παράθυρο της

Χρήσης υλικών, συνεπώς αυτή η πληροφορία θα έπρεπε να είναι αμφίδρομη. Στην πράξη όμως, η σύνδεση αυτών δεν λειτουργούσε με ικανοποιητικό τρόπο. Η ποσότητα ενός υλικού που βρίσκονταν στην Αποθήκη εμφανιζόταν στην Χρήση υλικών, αλλά όταν τραβούσαμε μία ποσότητα, αυτή η ποσότητα δεν αφαιρούνταν από την Αποθήκη. Το πρόβλημα επιδιορθώθηκε συνδέοντας και τα δύο παράθυρα με την βάση δεδομένων με συγκεκριμένη αρχιτεκτονική.

Ένα επιπρόσθετο πρόβλημα που προέκυψε κατά την δομική του συστήματος ήταν η εμφάνιση του προειδοποιητικού μηνύματος «έχετε έλλειψη υλικού» στο παράθυρο της Αποθήκης. Η λογική πίσω από αυτό το μήνυμα, ήταν ότι στην περίπτωση όπου η διαθέσιμη ποσότητα ενός υλικού ήταν μικρότερη από την ελάχιστη ποσότητα την οποία είχαμε ορίσει, τότε θα έπρεπε να εμφανίζεται το εν λόγω μήνυμα. Ενώ αρχικά θέτοντας μικρότερη τιμή στην διαθέσιμη ποσότητα από ότι στην ελάχιστη, το μήνυμα εμφανιζόταν, αν άλλαζε αυτή η συνθήκη, δηλαδή ανανεώναμε την διαθέσιμη ποσότητα με τιμή μεγαλύτερη από αυτήν της ελάχιστης, τότε το μήνυμα συνέχιζε να εμφανίζεται. Το πρόβλημα επιδιορθώθηκε αλλάζοντας το indentation στις γραμμές που ήταν υπεύθυνες για αυτή την συνθήκη.

Διαπιστώθηκε επίσης, στο παράθυρο των εκπαιδεύσεων, ότι στην περίπτωση που θέλαμε να δούμε ποιοι εργαζόμενοι έχουν παρακολουθήσει έναν κωδικό εκπαίδευσης, λειτουργούσε σωστά, το αντίθετο δεν εμφάνιζε αποτελέσματα. Δηλαδή στην περίπτωση που θέλαμε να δούμε τις εκπαιδεύσεις που έχει παρακολουθήσει ένας συγκεκριμένος εργαζόμενος, δεν μπορούσαμε να τις δούμε. Το πρόβλημα επιδιορθώθηκε δημιουργώντας ένα json αρχείο, ανεξάρτητο από την βάση δεδομένων.

ΚΕΦΑΛΑΙΟ 6: Συμπεράσματα και μελλοντική έρευνα

6.1 Συμπεράσματα

Η διαχείριση παραγωγής αποτελεί ένα κρίσιμο κομμάτι για κάθε επιχείρηση που επιθυμεί να λειτουργεί με αποτελεσματικότητα και αποδοτικότητα. Για να επιτευχθεί το παραπάνω, είναι ουσιώδες να υπάρχει ένα ολοκληρωμένο λογισμικό που να καλύπτει όλες τις κρίσιμες διαδικασίες και λειτουργίες.

Στην παρούσα εργασία περιγράψαμε τα στάδια ανάπτυξης λογισμικού διαχείρισης παραγωγής σε βιομηχανία χύτευσης αλουμινίου που παράγει προϊόντα για την αυτοκινητοβιομηχανία. Συγκεκριμένα, στο στάδιο της συλλογής απαιτήσεων πραγματοποιήθηκαν συνεντεύξεις με στελέχη της επιχείρησης. Ακολούθησε η ανάλυση και καταγραφή των απαιτήσεων. Στη συνέχεια ακολούθησε η σχεδίαση της βάσης δεδομένων. Η ανάπτυξη του λογισμικού πραγματοποιήθηκε με χρήση της δημοφιλούς γλώσσας ανοιχτού κώδικα Python. Χρησιμοποιήθηκαν πραγματικά δεδομένα της επιχείρησης στη φάση της δοκιμής ώστε να ελεγχθούν όλες οι λειτουργικές απαιτήσεις. Το περιβάλλον που αναπτύχθηκε είναι φιλικό στον χρήστη καθώς περιλαμβάνει οθόνες που διευκολύνουν την υλοποίηση των λειτουργιών.

Το λογισμικό που αναπτύχθηκε θα μπορούσε να συμβάλλει στη διευκόλυνση της διαχείρισης παραγωγής, προσφέροντας στο χρήστη ένα πρακτικό εργαλείο για τις καθημερινές λειτουργίες που μπορούν να εκτελεστούν με μεγαλύτερη ταχύτητα, ακρίβεια, ασφάλεια, και αποτελεσματικότητα. Αρκετά είναι και τα διοικητικά οφέλη τα οποία συνδέονται με τη χρήση του λογισμικού, όπως η βελτίωση της αποδοτικότητας, η μείωση του ρίσκου, και η υποστήριξη λειτουργιών όπως ο προγραμματισμός της παραγωγής.

6.2 Μελλοντική έρευνα

Με την ταχεία εξέλιξη των τεχνολογιών και των μεθόδων διαχείρισης, υπάρχουν ευκαιρίες για να αναπτυχθούν νέες προσεγγίσεις και εργαλεία που θα βελτιώσουν την απόδοση και την αποτελεσματικότητα στη διαχείριση παραγωγής με την βοήθεια ενός πληροφοριακού συστήματος.

Μια τεχνολογία που θα μπορούσε να ενσωματωθεί σε ένα τέτοιο σύστημα, είναι η χρήση της τεχνητής νοημοσύνης και της μηχανικής μάθησης στη διαχείριση παραγωγής. Οι αλγόριθμοι μηχανικής μάθησης μπορούν να αναλύουν μεγάλα σύνολα δεδομένων για να παράγουν προβλέψεις σχετικά με τη ζήτηση, τον προγραμματισμό της παραγωγής και τη διαχείριση του αποθέματος με μεγαλύτερη ακρίβεια και αποτελεσματικότητα.

Επιπλέον, η χρήση των αισθητήρων βασισμένων σε τεχνολογίες όπως το IoT στο πεδίο της παραγωγής, μπορεί να προσφέρει νέες δυνατότητες για την παρακολούθηση και τον έλεγχο των διαδικασιών παραγωγής σε πραγματικό χρόνο. Αυτό μπορεί να οδηγήσει σε συστήματα που ανταποκρίνονται δυναμικά στις μεταβαλλόμενες συνθήκες παραγωγής.

Ακόμα, πρακτικές που εστιάζουν στο περιβαλλοντικό αποτύπωμα, μπορούν να ενσωματωθούν σε ένα τέτοιο πληροφοριακό σύστημα με σκοπό την συστηματική παρακολούθηση των ρύπων και την μείωση αυτών.

Ξενόγλωσση Βιβλιογραφία

Chopra, R., Sawant, L., Kodi, D., & Terkar, R. (2022). Utilization of ERP systems in manufacturing industry for productivity improvement. *Materials Today: Proceedings, 62*(2).

Downey, A. (2016). *Think Python: How to Think Like a Computer Scientist.* O'Reilly Media.

Gessa, A., Jiménez, A., & Sancha, P. (2023). Exploring ERP systems adoption in challenging times. Insights of SMEs stories. *Technological Forecasting and Social Change, 195*(22).

Hustad, E., & Stensholt, J. (2023). Customizing ERP-systems: A framework to support the decision-making process. *Procedia Computer Science, 219*(35).

Lutz, M. (2013). *Learning Python, 5th Edition.* O'Reilly Media, Inc.

Malhotra, R., Goyal, S., Varshney, S., & Gupta, V. (2018). *Python Programming.* Namya press.

Pinciroli, F., Justo,, J., & Forradellas, R. (2022). Systematic mapping study: On the coverage of aspect-oriented methodologies for the early phases of the software development life cycle. *Journal of King Saud University - Computer and Information Sciences, 34*(6).

Ramalho, L. (2014). *Fluent Python: Clear, Concise, and Effective Programming.* O'Reilly.

Schmidt, R. F. (2013). *Software Engineering, Architecture-driven Software Development.* Morgan Kaufmann.

Sommerville, I. (2011). *Software Engineering* (9 ed.). Pearson.

Ελληνική Βιβλιογραφία

Βεσκούκης, Β. (2015). *Στοιχεία Τεχνολογίας Λογισμικού.* Αθήνα: Κάλλιπος.

Κεχρής, Ε. (2015). *Σχεσιακές βάσεις δεδομένων*. 2η έκδοση, Κριτική.

Κοκκίνου, Α. (2023). *Διαχείριση Επιχειρησιακών Πόρων: Σύγχρονα Πληροφοριακά Συστήματα.* Αθήνα: Κάλλιπος.

Μητάκος, Θ. (2015). *Πληροφοριακά Συστήματα Διοίκησης.* Αθήνα: Κάλλιπος.

Μιαούλης, Γ., Μπουσδέκης, Α., & Θεοδωροπούλου, Γ. (2021). *Πληροφοριακά Συστήματα: Ανάλυση, Σχεδιασμός, Ανάπτυξη.* Αθήνα: Κάλλιπος.

Φιτσιλής, Π. (2015). *Σύγχρονα Πληροφοριακά Συστήματα Επιχειρήσεων.* Αθήνα: Κάλλιπος.

Χατζηγιαννάκης, Ν. Μ. (2023). *Η γλώσσα Python σε βάθος.* Αθήνα: Κλειδάριθμος .

Παράρτημα 1: Εγχειρίδιο χρήσης

Είσοδος

Εικόνα 8: *Είσοδος στο πρόγραμμα*

Όταν ανοίγουμε το πρόγραμμα, βλέπουμε το περιβάλλον της εικόνας 8, όπου πληκτρολογούμε τον προσωπικό μας κωδικό χρήστη για να εισέλθουμε στο κυρίως μενού. Σε περίπτωση που πληκτρολογήσουμε λανθασμένο κωδικό 3 φορές συνεχόμενα, το πρόγραμμα μας κλειδώνει για ένα λεπτό και στην συνέχεια μπορούμε να εισάγουμε και πάλι τον κωδικό.

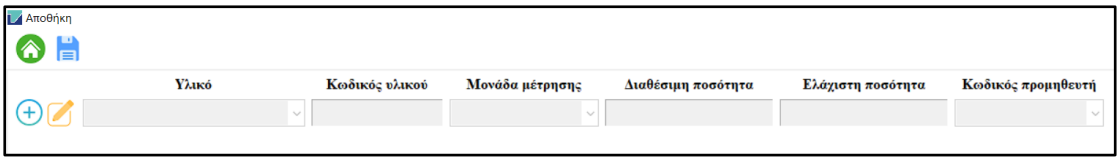
Κεντρικό μενού

Εικόνα 9: *Κεντρικό μενού προγράμματος*

Στο κεντρικό μενού (εικόνα 9) βλέπουμε τα modules που μπορούμε να διαχειριστούμε στην πάνω αριστερή γωνία του παραθύρου. Στην κάτω αριστερή γωνία, μπορούμε πιέζοντας το κόκκινο κουμπί να αποσυνδεθούμε. Τέλος στην πάνω δεξιά γωνία βλέπουμε το όνομα του χρήστη σύμφωνα με τον κωδικό που πληκτρολόγησε κατά την είσοδο.

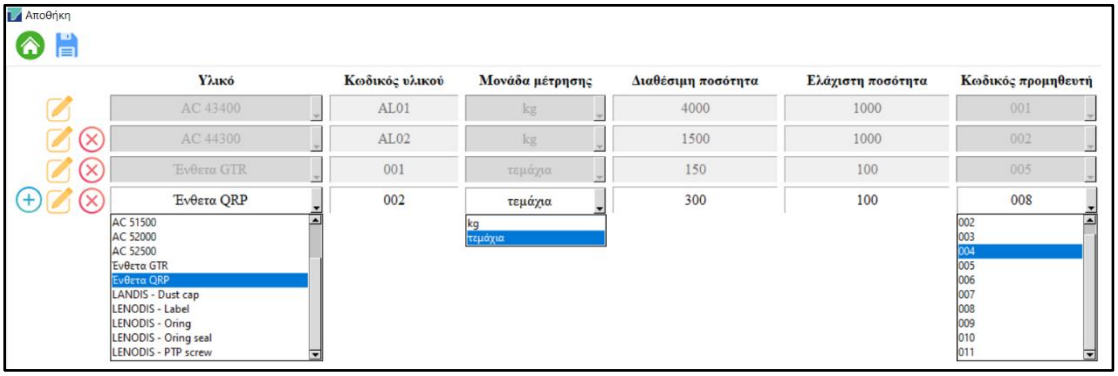
Αποθήκη

Επιλέγοντας το module της αποθήκης, εμφανίζεται στην οθόνη το παρακάτω παράθυρο (εικόνα 10).



***Εικόνα 10**: Παράθυρο Αποθήκη*

Αν πατήσουμε το κουμπί της επεξεργασίας ενεργοποιούνται τα πεδία και μπορούμε να διαλέξουμε μεταξύ των διαφόρων επιλογών που μας δίνονται. Όπου δεν υπάρχουν διαθέσιμες επιλογές, συμπληρώνουμε χειροκίνητα τα πεδία (εικόνα 11).



***Εικόνα 11:** Επεξεργασία πεδίων Αποθήκης*

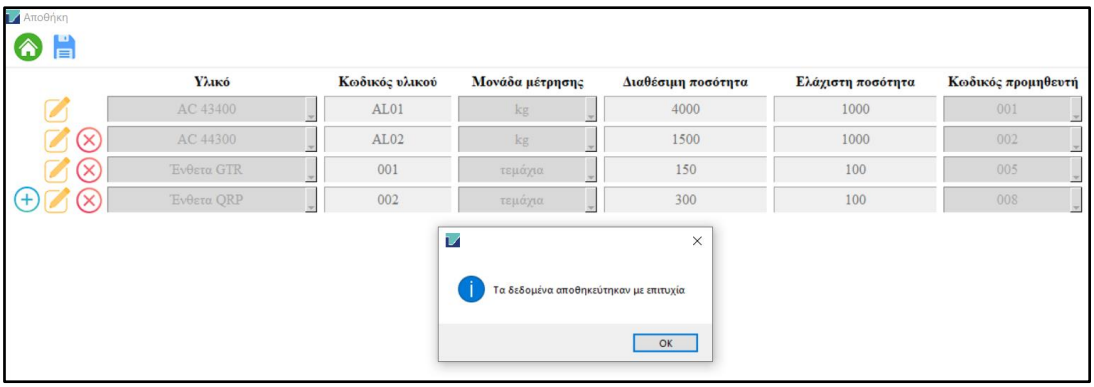
Στην περίπτωση που η διαθέσιμη ποσότητα είναι μικρότερη από την ελάχιστη ποσότητα που έχουμε ορίσει, τότε εμφανίζεται το μήνυμα: » Έχετε έλλειψη υλικού », υποδεικνύοντάς μας, ότι θα πρέπει να ανανεώσουμε την ποσότητα (εικόνα 12).

***Εικόνα 12:** Μήνυμα έλλειψης υλικού στην Αποθήκη*

Αν πιέσουμε το πλήκτρο ⊕ μπορούμε να προσθέσουμε μία νέα γραμμή και αντίστοιχα αν πιέσουμε το πλήκτρο ⊗ μπορούμε να αφαιρέσουμε μία ήδη υπάρχουσα γραμμή (εικόνα 13).

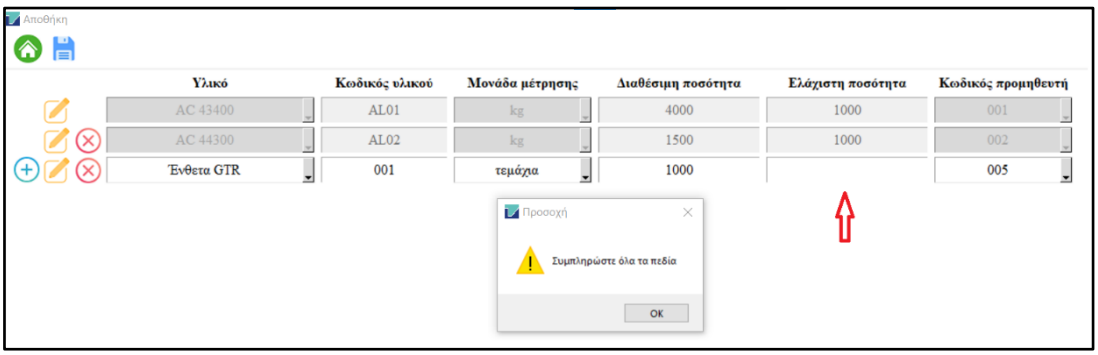
Εικόνα 13: *προσθήκη και αφαίρεση στοιχείων στην Αποθήκη*

Όταν ολοκληρώσουμε τις αλλαγές, πιέζουμε το κουμπί της αποθήκευσης 💾 και εμφανίζεται το μήνυμα που φαίνεται στην εικόνα 14. Πατώντας ξανά το κουμπί της επεξεργασίας, μπορούμε να τροποποιήσουμε, εάν θέλουμε, μία υπάρχουσα γραμμή.



Εικόνα 14: *Διαδικασία αποθήκευσης δεδομένων στην Αποθήκη*

Εάν έχουμε αφήσει κάποιο πεδίο κενό, τότε το πρόγραμμα δεν μας αφήνει να πραγματοποιήσουμε την αποθήκευση και μας εμφανίζει το μήνυμα « Συμπληρώστε όλα τα πεδία « (εικόνα 15). Στην συνέχεια πιέζοντας το πλήκτρο 🏠 μπορούμε να επιστρέψουμε στην αρχική.



Εικόνα 15: *Μήνυμα υποχρεωτικής συμπλήρωσης όλων των πεδίων της Αποθήκης*

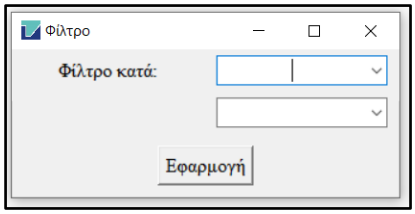
Εκπαιδεύσεις

Επιλέγοντας από το κεντρικό μενού το module εκπαιδεύσεις, θα οδηγηθούμε στο περιβάλλον που βλέπουμε στην εικόνα 16.

Εικόνα 16: *Παράθυρο Εκπαιδεύσεις*

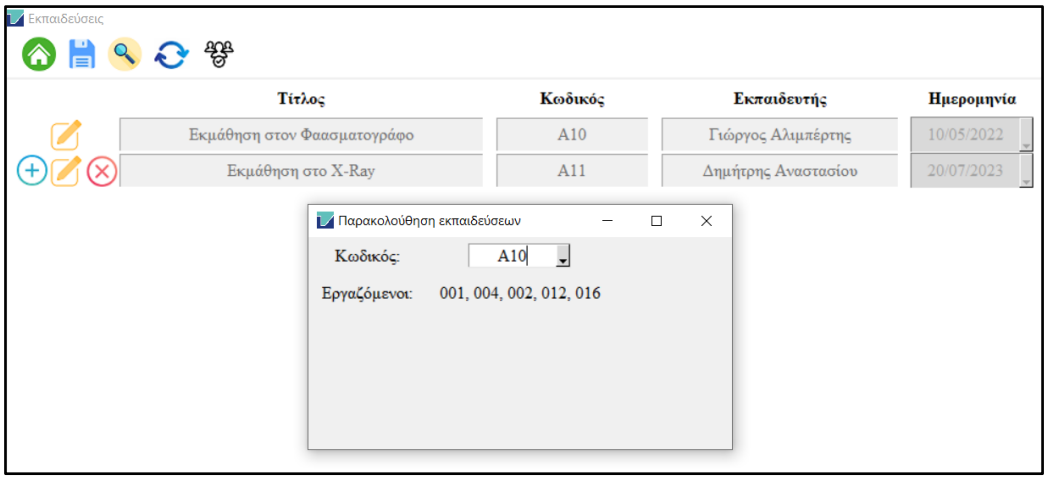
Πατώντας το πλήκτρο της επεξεργασίας, μπορούμε να επεξεργαστούμε τα διάφορα πεδία. Επίσης μπορούμε να προσθέσουμε αλλά και να αφαιρέσουμε ή να τροποποιήσουμε γραμμές (βλέπε Αποθήκη). Αφού πραγματοποιήσουμε τις αλλαγές, πατάμε το πλήκτρο της αποθήκευσης, ώστε να αποθηκεύσουμε τα δεδομένα μας.

Πατώντας το πλήκτρο , εμφανίζεται το παράθυρο που φαίνεται στην εικόνα 17. Μπορούμε να φιλτράρουμε τα δεδομένα μας με βάση των κωδικό ή την ημερομηνία που επιθυμούμε, και να δούμε τα αντίστοιχα αποτελέσματα.



Εικόνα 17: *Επιλογή φίλτρων στο παράθυρο των Εκπαιδεύσεων*

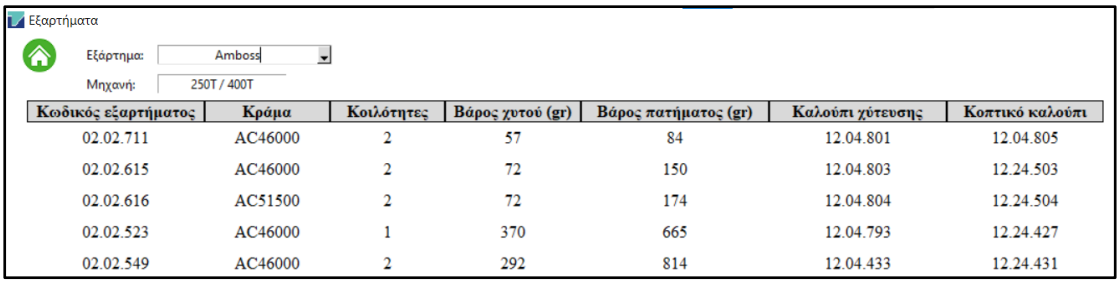
Πατώντας το πλήκτρο , πραγματοποιείται ανανέωση του παραθύρου. Τέλος αν πιέσουμε το πλήκτρο , επιλέγοντας κωδικό εκπαίδευσης, μπορούμε να δούμε ποιοι εργαζόμενοι έχουν παρακολουθήσει την συγκεκριμένη εκπαίδευση (εικόνα 18).



Εικόνα 18: *Παρακολούθηση εκπαιδεύσεων*

Εξαρτήματα

Επιλέγοντας από το κεντρικό μενού τα εξαρτήματα, βλέπουμε το παράθυρο που εμφανίζεται στην εικόνα 19.

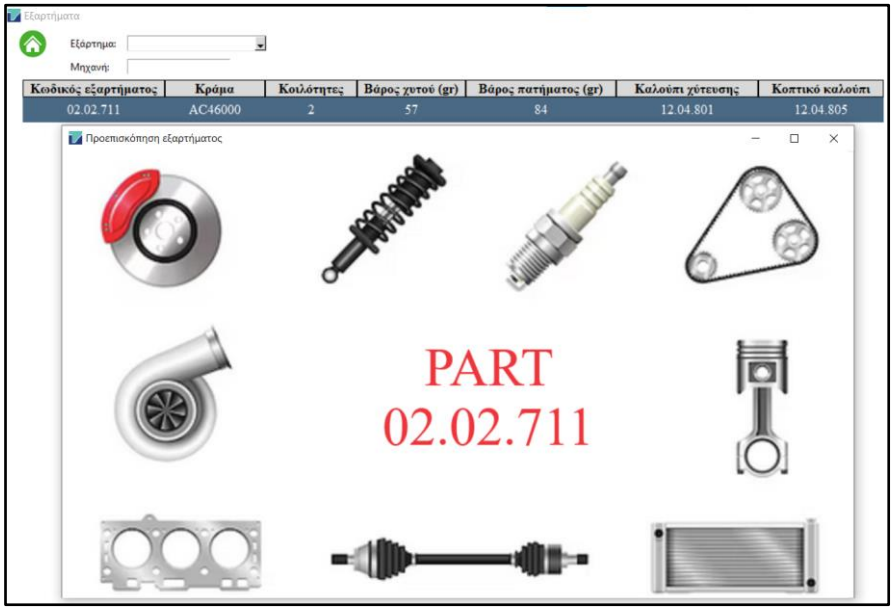


Κωδικός εξαρτήματος	Κράμα	Κοιλότητες	Βάρος χυτού (gr)	Βάρος πατήματος (gr)	Καλούπι χύτευσης	Κοπτικό καλούπι
02.02.711	AC46000	2	57	84	12.04.801	12.04.805
02.02.615	AC46000	2	72	150	12.04.803	12.24.503
02.02.616	AC51500	2	72	174	12.04.804	12.24.504
02.02.523	AC46000	1	370	665	12.04.793	12.24.427
02.02.549	AC46000	2	292	814	12.04.433	12.24.431

***Εικόνα 19**: Παράθυρο Εξαρτήματα*

Σε αυτό το module μπορούμε να δούμε χρήσιμες πληροφορίες για τα εξαρτήματα, όπως είναι ο κωδικός εξαρτήματος και το κράμα. Επίσης βλέπουμε πληροφορίες για την παραγωγή του εξαρτήματος όπως είναι η ονομασία του καλουπιού χύτευσης και του κοπτικού καλουπιού που χρησιμοποιούνται για την παραγωγή του εξαρτήματος αλλά και γενικές πληροφορίες όπως το βάρος του χυτού και του πατήματος. Επίσης υπάρχει η δυνατότητα, επιλέγοντας το όνομα του εξαρτήματος στο πεδίο εξάρτημα που βρίσκεται στην κορυφή της οθόνης, να δούμε σε ποια μηχανή ή μηχανές παράγεται το συγκεκριμένο εξάρτημα (εικόνα 19).

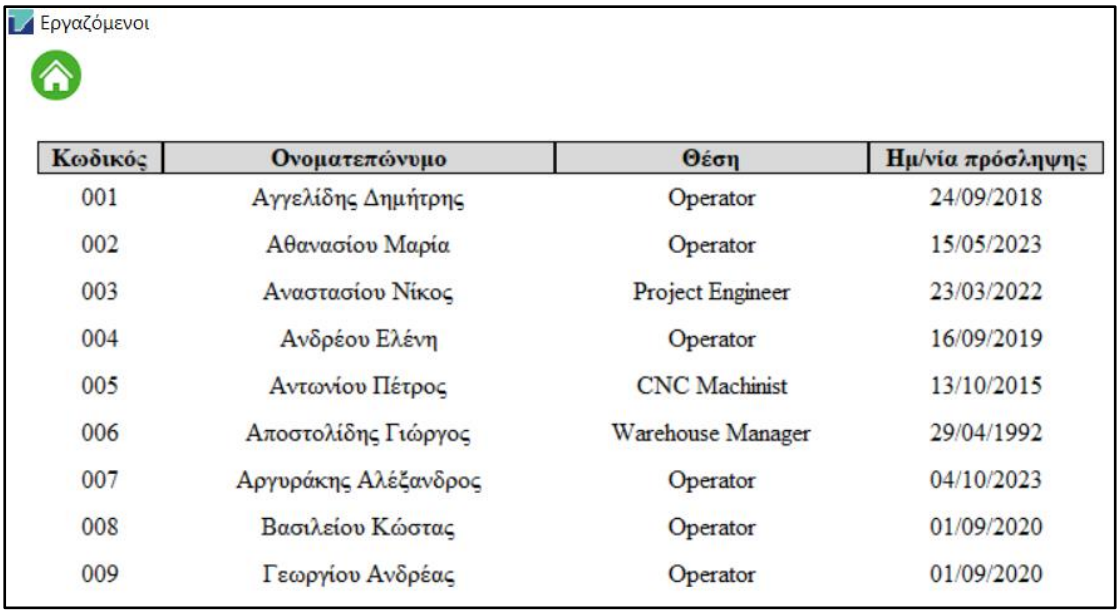
Στο ίδιο παράθυρο, πατώντας επάνω σε ένα κωδικό εξαρτήματος με διπλό κλικ, μπορούμε να δούμε μια προεπισκόπηση του εξαρτήματος (εικόνα 20).



***Εικόνα 20:** Προεπισκόπηση εξαρτήματος*

Εργαζόμενοι

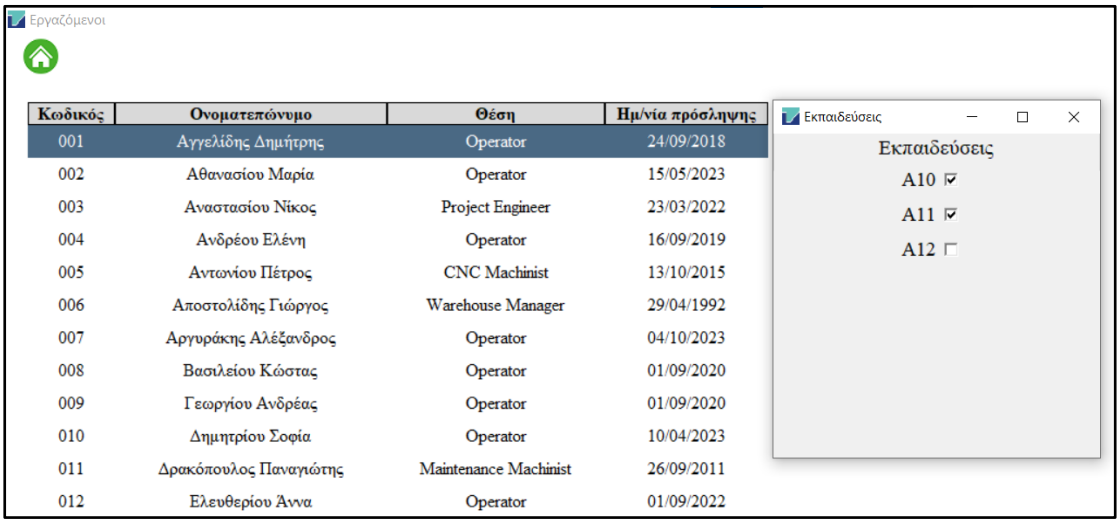
Πιέζοντας στο κεντρικό μενού το module εργαζόμενοι βλέπουμε το περιβάλλον της εικόνας 21.



Κωδικός	Ονοματεπώνυμο	Θέση	Ημ/νία πρόσληψης
001	Αγγελίδης Δημήτρης	Operator	24/09/2018
002	Αθανασίου Μαρία	Operator	15/05/2023
003	Αναστασίου Νίκος	Project Engineer	23/03/2022
004	Ανδρέου Ελένη	Operator	16/09/2019
005	Αντωνίου Πέτρος	CNC Machinist	13/10/2015
006	Αποστολίδης Γιώργος	Warehouse Manager	29/04/1992
007	Αργυράκης Αλέξανδρος	Operator	04/10/2023
008	Βασιλείου Κώστας	Operator	01/09/2020
009	Γεωργίου Ανδρέας	Operator	01/09/2020

Εικόνα 21: *Παράθυρο Εργαζόμενοι*

Σε αυτό το παράθυρο μπορούμε να δούμε χρήσιμες πληροφορίες για τους εργαζόμενους όπως κωδικός, ονοματεπώνυμο, θέση και ημερομηνία πρόσληψης. Επίσης, πατώντας με διπλό κλικ κάποιο όνομα, εμφανίζεται το παράθυρο που φαίνεται στην εικόνα 22, όπου μπορούμε να δούμε ποιες εκπαιδεύσεις έχει παρακολουθήσει ο κάθε εργαζόμενος, όπως επίσης και να προσθέσουμε ή να αφαιρέσουμε εκπαιδεύσεις.

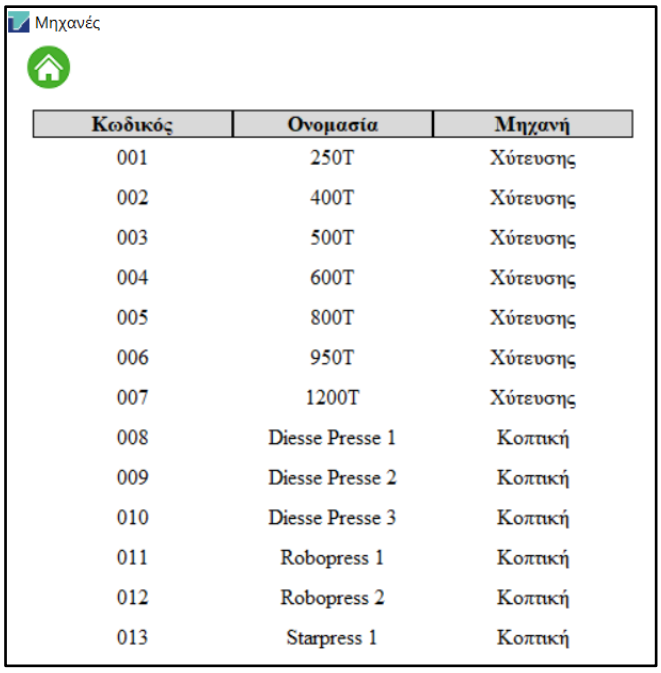


Κωδικός	Ονοματεπώνυμο	Θέση	Ημ/νία πρόσληψης
001	Αγγελίδης Δημήτρης	Operator	24/09/2018
002	Αθανασίου Μαρία	Operator	15/05/2023
003	Αναστασίου Νίκος	Project Engineer	23/03/2022
004	Ανδρέου Ελένη	Operator	16/09/2019
005	Αντωνίου Πέτρος	CNC Machinist	13/10/2015
006	Αποστολίδης Γιώργος	Warehouse Manager	29/04/1992
007	Αργυράκης Αλέξανδρος	Operator	04/10/2023
008	Βασιλείου Κώστας	Operator	01/09/2020
009	Γεωργίου Ανδρέας	Operator	01/09/2020
010	Δημητρίου Σοφία	Operator	10/04/2023
011	Δρακόπουλος Παναγιώτης	Maintenance Machinist	26/09/2011
012	Ελευθερίου Άννα	Operator	01/09/2022

Εικόνα 22: *Εκπαιδεύσεις εργαζομένων*

Μηχανές

Πιέζοντας από το κεντρικό μενού το πλήκτρο μηχανές, οδηγούμαστε στο παράθυρο που βλέπουμε στην εικόνα 23.



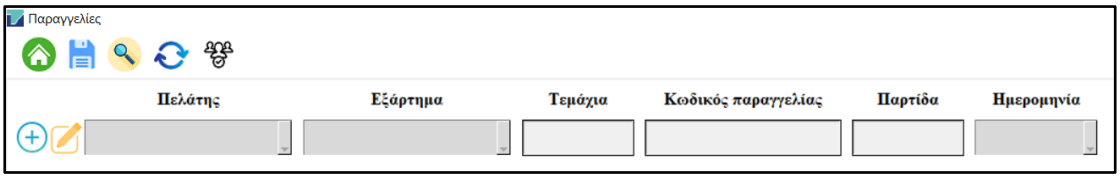
Κωδικός	Ονομασία	Μηχανή
001	250T	Χύτευσης
002	400T	Χύτευσης
003	500T	Χύτευσης
004	600T	Χύτευσης
005	800T	Χύτευσης
006	950T	Χύτευσης
007	1200T	Χύτευσης
008	Diesse Presse 1	Κοπτική
009	Diesse Presse 2	Κοπτική
010	Diesse Presse 3	Κοπτική
011	Robopress 1	Κοπτική
012	Robopress 2	Κοπτική
013	Starpress 1	Κοπτική

Εικόνα 23: *Παράθυρο Μηχανές*

Εδώ μπορούμε να δούμε τις διαθέσιμες μηχανές, τον προσωπικό τους κωδικό καθώς και την ονομασία τους. Επίσης μπορούμε να διακρίνουμε σε ποια κατηγορία ανήκει η κάθε μηχανή.

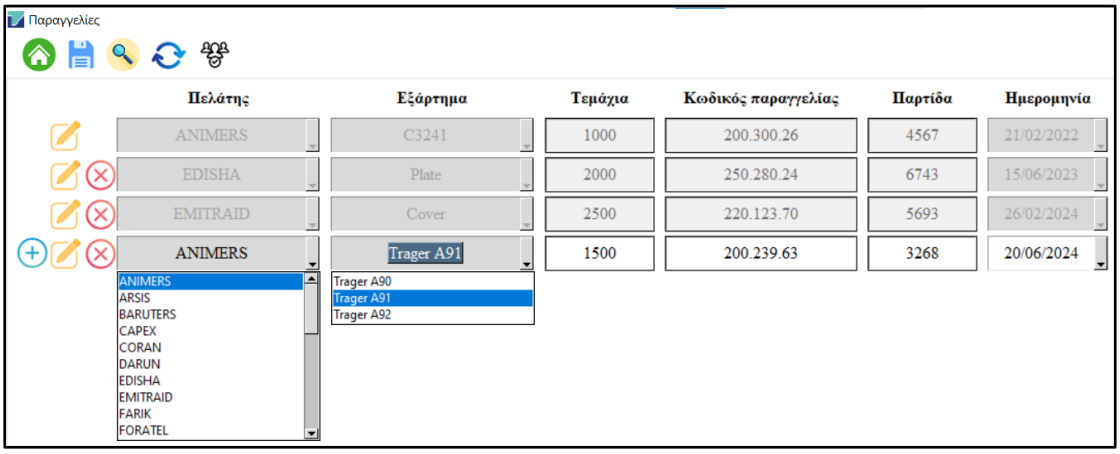
Παραγγελίες

Επιλέγοντας από το κεντρικό μενού το module παραγγελίες, εμφανίζεται το παράθυρο που βλέπουμε στην εικόνα 24.



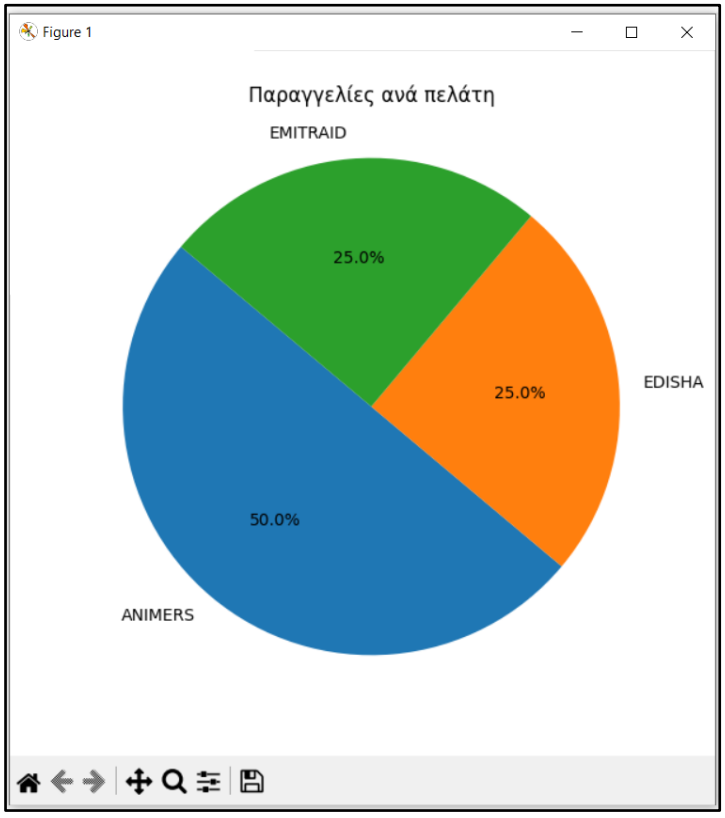
Εικόνα 24: *Παράθυρο Παραγγελίες*

Πατώντας το πλήκτρο της επεξεργασίας, επιλέγουμε αρχικά τον πελάτη. Στην συνέχεια στο πεδίο εξάρτημα, θα δούμε τα αντίστοιχα εξαρτήματα που αφορούν τον συγκεκριμένο πελάτη. Έπειτα συμπληρώνουμε χειροκίνητα τα υπόλοιπα πεδία όπως φαίνεται στην εικόνα 25.



***Εικόνα 25**: Συμπλήρωση πεδίων στις Παραγγελίες*

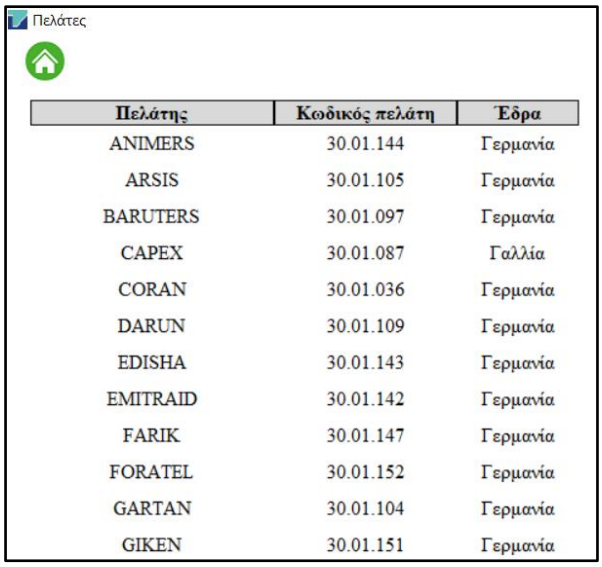
Μπορούμε επίσης να προσθέσουμε και να αφαιρέσουμε γραμμές όπως επίσης και να τροποποιήσουμε μία ήδη υπάρχουσα (βλέπε Αποθήκη). Ακόμα μπορούμε να φιλτράρουμε τα δεδομένα μας με βάση τον πελάτη, το εξάρτημα, την παρτίδα καθώς και την ημερομηνία (βλέπε Αποθήκη). Μια ακόμα δυνατότητα που έχουμε σε αυτό το παράθυρο είναι ότι πατώντας το πλήκτρο , εμφανίζεται ένα υποπαράθυρο στο οποίο μπορούμε να δούμε τα ποσοστά παραγγελιών ανά πελάτη (εικόνα 26).



***Εικόνα 26:** Ποσοστά παραγγελιών ανά πελάτη*

Πελάτες

Επιλέγοντας από το κεντρικό μενού τους πελάτες, βλέπουμε το περιβάλλον που φαίνεται στην εικόνα 27.



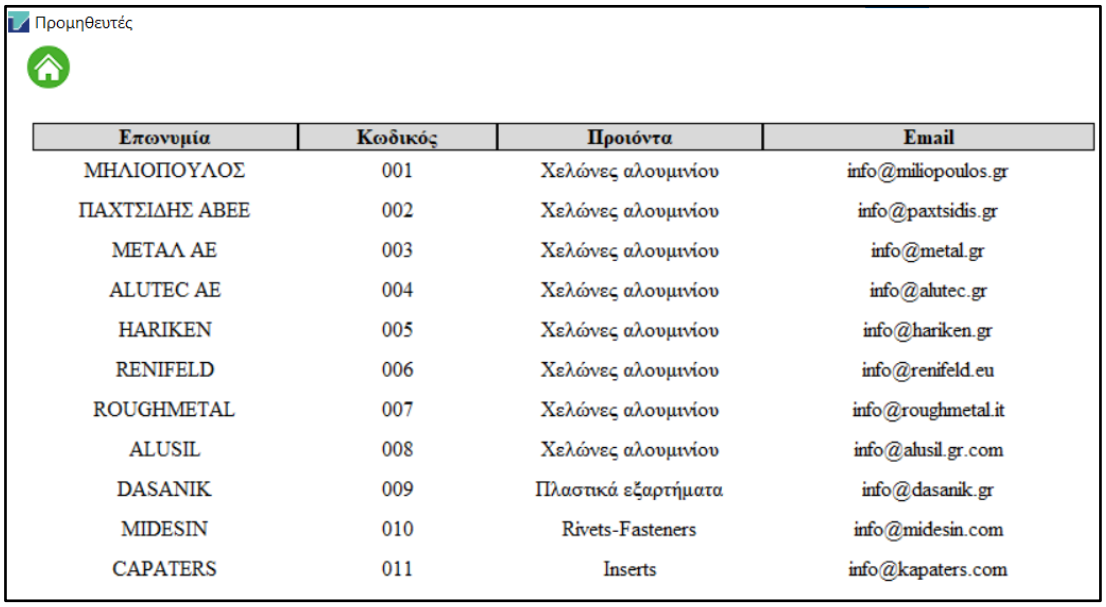
Πελάτης	Κωδικός πελάτη	Έδρα
ANIMERS	30.01.144	Γερμανία
ARSIS	30.01.105	Γερμανία
BARUTERS	30.01.097	Γερμανία
CAPEX	30.01.087	Γαλλία
CORAN	30.01.036	Γερμανία
DARUN	30.01.109	Γερμανία
EDISHA	30.01.143	Γερμανία
EMITRAID	30.01.142	Γερμανία
FARIK	30.01.147	Γερμανία
FORATEL	30.01.152	Γερμανία
GARTAN	30.01.104	Γερμανία
GIKEN	30.01.151	Γερμανία

Εικόνα 27: *Παράθυρο Πελάτες*

Σε αυτό το παράθυρο βλέπουμε πληροφορίες για τους πελάτες, και πιο συγκεκριμένα τον κωδικό του πελάτη, το όνομα καθώς και την έδρα που βρίσκεται.

Προμηθευτές

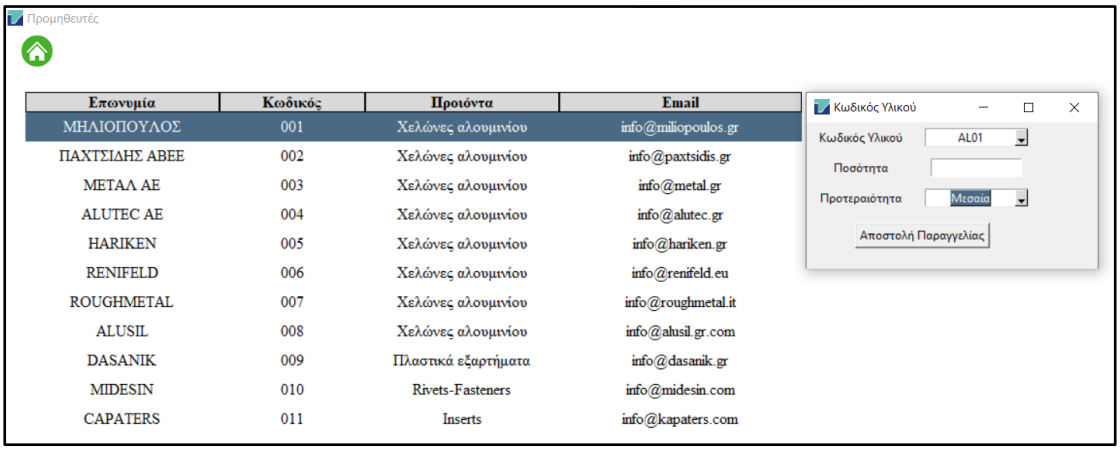
Πιέζοντας στο κεντρικό μενού το πλήκτρο των προμηθευτών, εμφανίζεται το παράθυρο της εικόνας 28.



Επωνυμία	Κωδικός	Προϊόντα	Email
ΜΗΛΙΟΠΟΥΛΟΣ	001	Χελώνες αλουμινίου	info@miliopoulos.gr
ΠΑΧΤΣΙΔΗΣ ΑΒΕΕ	002	Χελώνες αλουμινίου	info@paxtsidis.gr
ΜΕΤΑΛ ΑΕ	003	Χελώνες αλουμινίου	info@metal.gr
ALUTEC AE	004	Χελώνες αλουμινίου	info@alutec.gr
HARIKEN	005	Χελώνες αλουμινίου	info@hariken.gr
RENIFELD	006	Χελώνες αλουμινίου	info@renifeld.eu
ROUGHMETAL	007	Χελώνες αλουμινίου	info@roughmetal.it
ALUSIL	008	Χελώνες αλουμινίου	info@alusil.gr.com
DASANIK	009	Πλαστικά εξαρτήματα	info@dasanik.gr
MIDESIN	010	Rivets-Fasteners	info@midesin.com
CAPATERS	011	Inserts	info@kapaters.com

Εικόνα 28: *Παράθυρο Προμηθευτές*

Σε αυτό το module μπορούμε να δούμε τους προμηθευτές μας και τα διάφορα στοιχεία τους όπως κωδικός, προϊόντα που προμηθευόμαστε και το email επικοινωνίας. Επίσης πατώντας διπλό κλικ σε κάποιον προμηθευτή, εμφανίζεται το παράθυρο της εικόνας 29. Εκεί μπορούμε να επιλέξουμε το υλικό που θέλουμε να παραγγείλουμε, την ποσότητα καθώς και το πόσο επείγον είναι ή όχι. Στην συνέχεια, πατώντας το πλήκτρο της αποστολής παραγγελίας, αποστέλλεται η παραγγελία στον προμηθευτή μας.



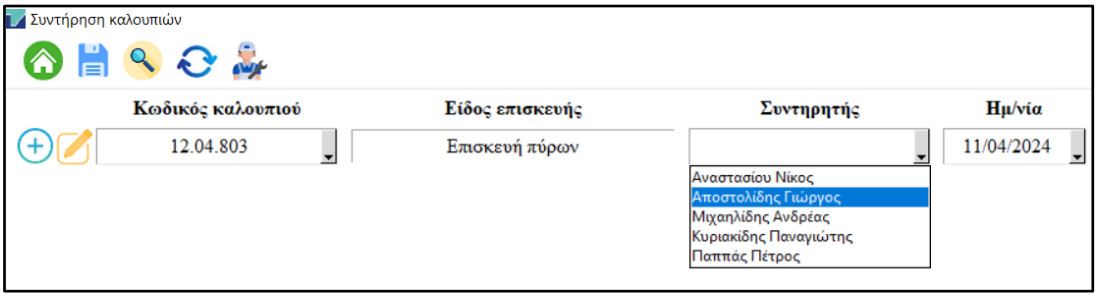
Εικόνα 29: *Αποστολή παραγγελίας*

Συντήρηση καλουπιών

Εάν στο κεντρικό μενού επιλέξουμε το module συντήρηση καλουπιών, βλέπουμε το παράθυρο της εικόνας 30.

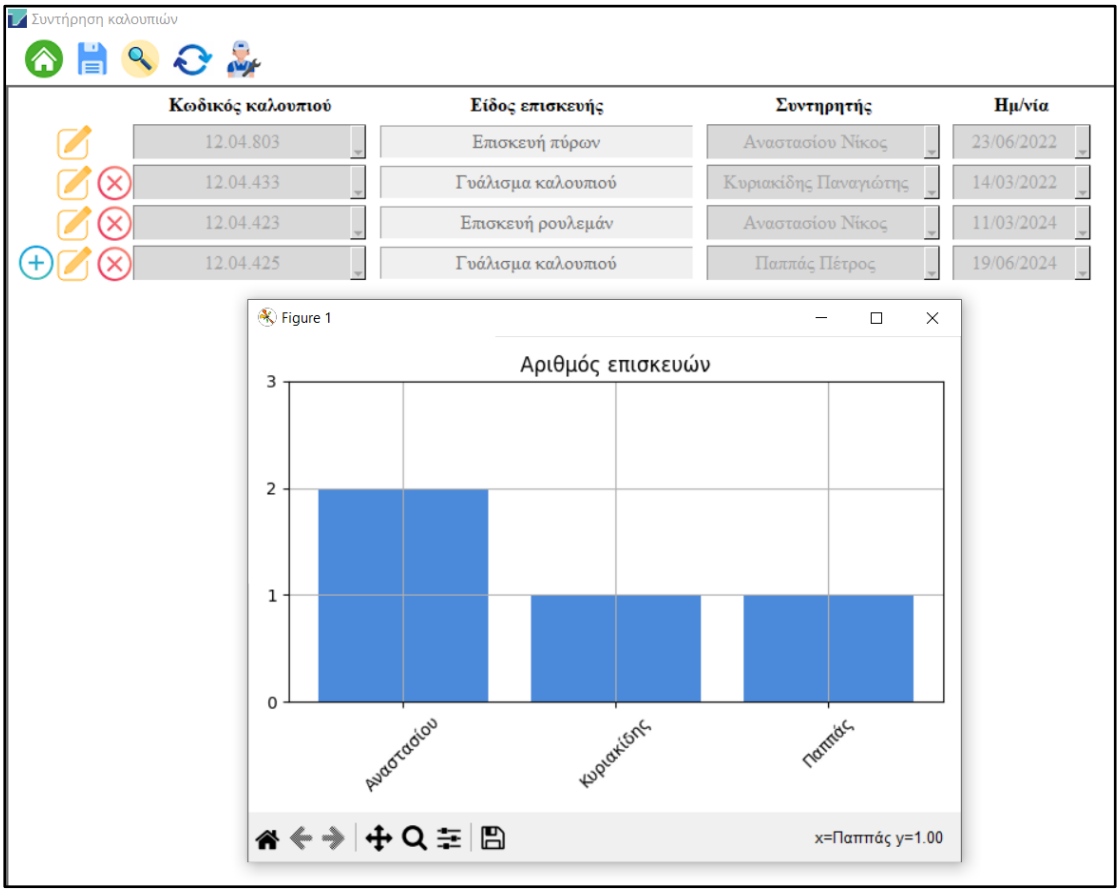
Εικόνα 30: *Παράθυρο Συντήρηση καλουπιών*

Πατώντας το πλήκτρο της επεξεργασίας, επιλέγουμε αρχικά τον κωδικό καλουπιού. Στην συνέχεια στο πεδίο είδος επισκευής, συμπληρώνουμε χειροκίνητα το λόγο για τον οποίο επισκευάστηκε το εν λόγω καλούπι. Έπειτα συμπληρώνουμε το όνομα του συντηρητή και τέλος την ημερομηνία της συντήρησης. Η παραπάνω διαδικασία, φαίνεται στην εικόνα 31.



Εικόνα 31: *Συμπλήρωση πεδίων στο παράθυρο Συντήρηση καλουπιών*

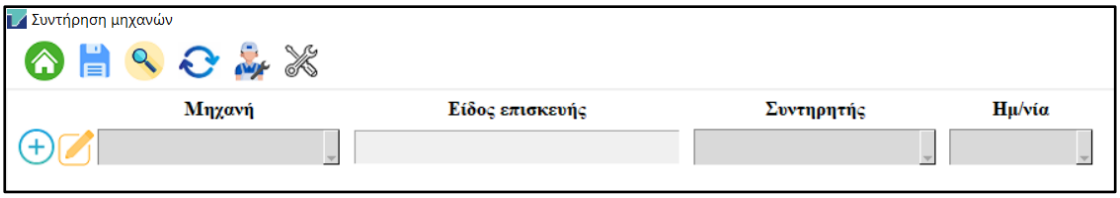
Μπορούμε επίσης να προσθέσουμε, να αφαιρέσουμε και να τροποποιήσουμε γραμμές (βλέπε Αποθήκη). Πατώντας το πλήκτρο του φίλτρου μπορούμε να φιλτράρουμε τα δεδομένα μας με βάση τον κωδικό καλουπιού, το όνομα του συντηρητή καθώς και την ημερομηνία. Τέλος επιλέγοντας το πλήκτρο , μπορούμε να δούμε το σύνολο των επισκευών που έχει πραγματοποιήσει κάθε συντηρητής (εικόνα 32).



Εικόνα 32: *Αριθμός επισκευών ανά συντηρητή*

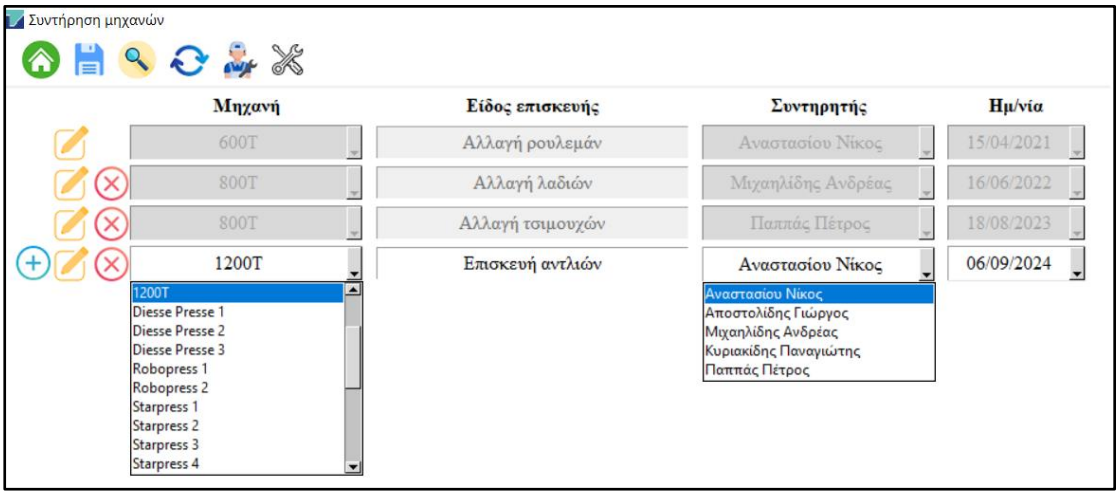
Συντήρηση μηχανών

Επιλέγοντας από το κεντρικό μενού την συντήρηση μηχανών, εμφανίζεται το παράθυρο που φαίνεται στην εικόνα 33.



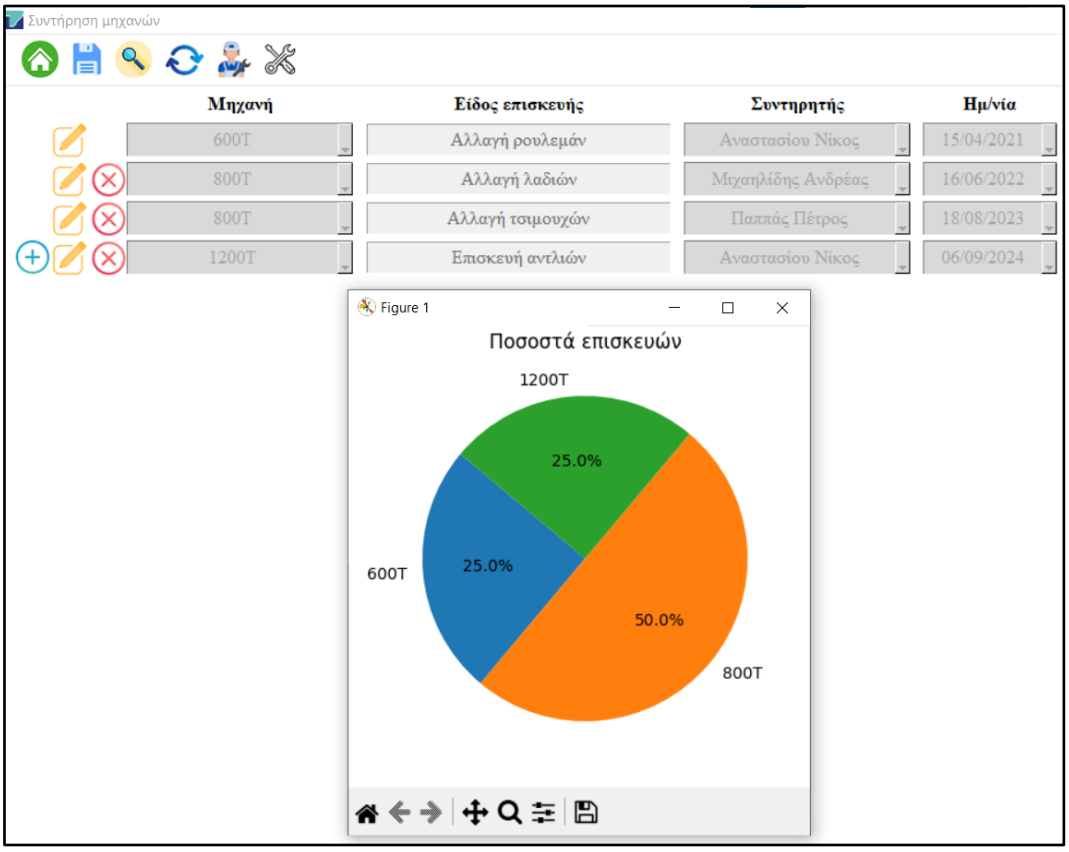
***Εικόνα 33**: Παράθυρο Συντήρηση μηχανών*

Εάν πατήσουμε το πλήκτρο της επεξεργασίας μπορούμε να επεξεργαστούμε τα διάφορα πεδία. Στο πεδίο μηχανή, έχουμε διαθέσιμους όλους τους κωδικούς μηχανών και στο πεδίο συντηρητής, επιλέγουμε το όνομα του συντηρητή που πραγματοποίησε την επισκευή από την διαθέσιμη λίστα. Τέλος συμπληρώνουμε χειροκίνητα το είδος της επισκευής και την ημερομηνία. Η παραπάνω διαδικασία φαίνεται στην εικόνα 34.



***Εικόνα 34:** Συμπλήρωση πεδίων στην Συντήρηση μηχανών*

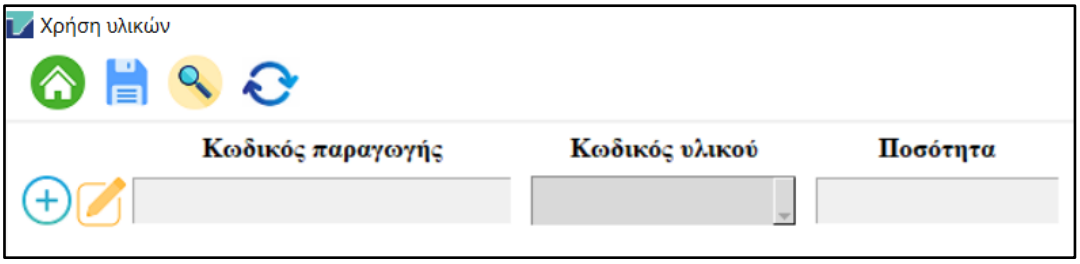
Μπορούμε επίσης να εισάγουμε και να διαγράψουμε γραμμές όπως και να τροποποιήσουμε τις ήδη υπάρχουσες (βλέπε Αποθήκη). Ακόμα μπορούμε να φιλτράρουμε τα δεδομένα μας με βάση το όνομα της μηχανής, το όνομα του συντηρητή και την ημερομηνία. Πατώντας το πλήκτρο , μπορούμε να δούμε το συνολικό αριθμό συντηρήσεων ανά συντηρητή (βλέπε συντήρηση καλουπιών). Τέλος, έχουμε ακόμα μία δυνατότητα, πιέζοντας το πλήκτρο , μπορούμε να δούμε τα ποσοστά επισκευών που έχουν πραγματοποιηθεί ανά μηχανή (εικόνα 35).



Εικόνα 35: *Ποσοστά επισκευών ανά μηχανή*

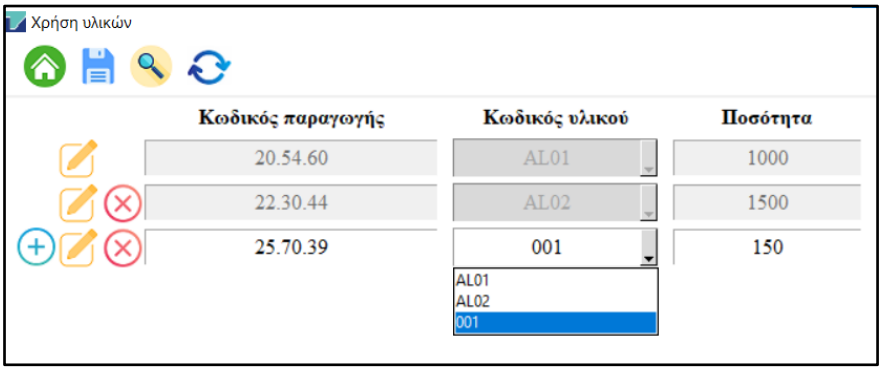
Χρήση υλικών

Επιλέγοντας από το κεντρικό μενού το module χρήση υλικών, βλέπουμε το παράθυρο που απεικονίζεται στην εικόνα 36.



Εικόνα 36: *Παράθυρο Χρήση υλικών*

Πατώντας το κουμπί της επεξεργασίας, συμπληρώνουμε χειροκίνητα τον κωδικό παραγωγής και επιλέγουμε κωδικό υλικού με βάση τις διαθέσιμες επιλογές που εμφανίζονται. Οι διαθέσιμες επιλογές αντλούνται από το module αποθήκη ενώ η ποσότητα που εμφανίζεται είναι η μέγιστη με βάση την διαθέσιμη ποσότητα που βρίσκεται στην αποθήκη. Τέλος συμπληρώνουμε την ποσότητα που θα χρειαστούμε. Η παραπάνω διαδικασία φαίνεται στην εικόνα 37.

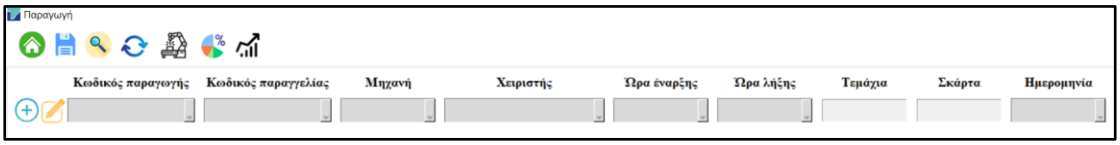


Εικόνα 37: *Συμπλήρωση πεδίων στην Χρήση*

Επιπρόσθετα, μπορούμε να προσθέσουμε, αφαιρέσουμε ή να τροποποιήσουμε γραμμές (βλέπε Αποθήκη). Τέλος μπορούμε να φιλτράρουμε τα δεδομένα μας με βάση τον κωδικό παραγγελίας ή τον κωδικό υλικού.

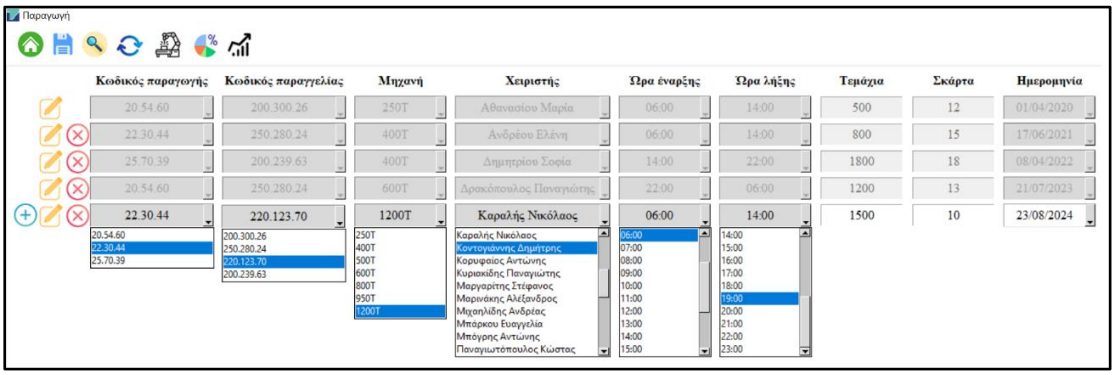
Παραγωγή

Στο κεντρικό μενού, επιλέγοντας το module παραγωγή, θα δούμε το περιβάλλον που φαίνεται στην εικόνα 38.



Εικόνα 38: *Παραγωγή*

Πιέζοντας το πλήκτρο της επεξεργασίας, μπορούμε να διαχειριστούμε τα διάφορα πεδία. Στον κωδικό παραγωγής εμφανίζονται οι κωδικοί παραγωγής που έχουμε εισάγει στο module χρήση υλικών ενώ στον κωδικό παραγγελίας αντίστοιχα εμφανίζονται οι κωδικοί παραγγελίας που έχουμε εισάγει στο module παραγγελίες. Στην συνέχεια συμπληρώνουμε τα πεδία μηχανές και χειριστής απο τις διαθέσιμες επιλογές, και τα υπόλοιπα πεδία τα συμπληρώνουμε χειροκίνητα (εικόνα 39).

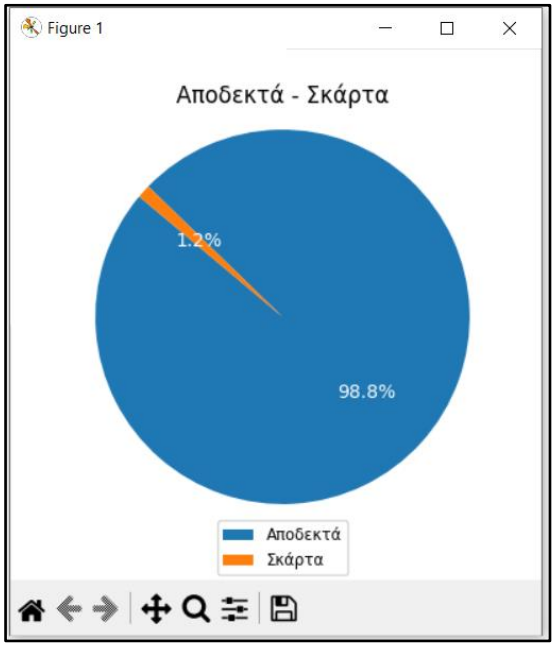


Εικόνα 39: *Συμπλήρωση πεδίων στην Παραγωγή*

Ακόμα, μπορούμε να προσθέσουμε, αφαιρέσουμε ή να τροποποιήσουμε γραμμές (βλέπε Αποθήκη). Μια ακόμα δυνατότητα που έχουμε σε αυτό το παράθυρο είναι πατώντας το πλήκτρο που βρίσκεται δεξιά από αυτό της ανανέωσης, να δούμε τα ποσοστά παραγωγικότητας κάθε μηχανής (εικόνα 40).

Εικόνα 40: *Παραγωγικότητα μηχανών*

Επίσης πιέζοντας το πλήκτρο δεξιά από αυτό της παραγωγικότητας μπορούμε να δούμε τι ποσοστό της παραγωγής μας ήταν τα σκάρτα (εικόνα 41).



Εικόνα 41: *Ποσοστά σκάρτων*

Τέλος, μια ακόμα δυνατότητα που έχουμε σε αυτό το παράθυρο, πιέζοντας το τελευταίο πλήκτρο στην κορυφή της οθόνης, να δούμε την ετήσια παραγωγή εξαρτημάτων (εικόνα 42).

Εικόνα 42: *Ετήσια παραγωγή*

Παράρτημα 2: Πηγαίος Κώδικας

```
import tkinter as tk
from tkinter import messagebox, ttk
from functools import partial
from tkcalendar import DateEntry
import uuid
import json
from datetime import datetime, timedelta
from tuples_file import ergazomenoi, mixanes, promitheutes, pelates, kaloupia, ulika
import sqlite3

window_stack = []

def create_database ():
    conn = sqlite3.connect('erp.db')
    c = conn. cursor ()

    c.execute ('''CREATE TABLE IF NOT EXISTS Αποθήκη
            (id TEXT PRIMARY KEY, Υλικό TEXT, Κωδικός_υλικού TEXT, Μονάδα_μέτρησης TEXT, Διαθέσιμη_ποσότητα
TEXT, Ελάχιστη_ποσότητα TEXT, Κωδικός_προμηθευτή TEXT)''')
    c.execute ('''CREATE TABLE IF NOT EXISTS Εκπαιδεύσεις
            (id TEXT PRIMARY KEY, Τίτλος TEXT, Κωδικός TEXT, Εκπαιδευτής TEXT, Ημερομηνία TEXT)''')
    c.execute ('''CREATE TABLE IF NOT EXISTS Παραγγελίες
            (id TEXT PRIMARY KEY, Πελάτης TEXT, Εξάρτημα TEXT, Τεμάχια TEXT, Κωδικός_παραγγελίας TEXT, Παρτίδα
TEXT, Ημερομηνία TEXT)''')
    c.execute ('''CREATE TABLE IF NOT EXISTS Παραγωγή
            (id TEXT PRIMARY KEY, Κωδικός_παραγωγής, Κωδικός_παραγγελίας TEXT, Μηχανή TEXT, Χειριστής TEXT,
Ώρα_έναρξης TEXT, Ώρα_λήξης TEXT, Τεμάχια INTEGER, Σκάρτα INTEGER, Ημερομηνία TEXT)''')
    c.execute ('''CREATE TABLE IF NOT EXISTS Χρήση
            (id TEXT PRIMARY KEY, Κωδικός_παραγωγής TEXT, Κωδικός_υλικού TEXT, Ποσότητα INTEGER)''')
    c.execute ('''CREATE TABLE IF NOT EXISTS Συμμετοχές
            (id TEXT PRIMARY KEY, Κωδικός_εκπαίδευσης TEXT, Συμμετέχοντες TEXT)''')
    c.execute ('''CREATE TABLE IF NOT EXISTS Συντήρηση
            (id TEXT PRIMARY KEY, Κωδικός_καλουπιού TEXT, Είδος_επισκευής TEXT, Συντηρητής TEXT, Ημερομηνία)
    conn. commit ()
    conn. close ()

create_database ()

def fetch_data(table_name):
    conn = sqlite3.connect('erp.db')
    c = conn. Cursor ()
    c.execute ("SELECT * FROM {}". format(table_name))
    records = c. fetchall ()
    conn. close ()
    return records

def fetch_data_apothiki ():
    return fetch_data("Αποθήκη")
```

```
def fetch_data_ekpaideuseis ():
    return fetch_data("Εκπαιδεύσεις")

def fetch_data_paraggelies ():
    return fetch_data("Παραγγελίες")

def fetch_data_paragogi ():
    return fetch_data("Παραγωγή")

def fetch_data_summetoxes ():
    return fetch_data("Συμμετοχές")

def fetch_data_suntirisi ():
    return fetch_data("Συντήρηση")

def fetch_data_xrisi ():
    return fetch_data("Χρήση")

def on_close ():
    if messagebox.askokcancel ("Τερματισμός", "Θέλετε σίγουρα να τερματίσετε το πρόγραμμα ;"):
        window. destroy ()

def create_submenu (title, zoomed=False):
    submenu = tk. Toplevel ()
    submenu. title(title)
    submenu. geometry (window. winfo_geometry () if not zoomed else "")
    submenu. protocol ("WM_DELETE_WINDOW", on_close)
    return submenu

def set_common_style ():
    style = ttk. Style ()
    style. theme_use("default")
    style. configure ("Treeview.Heading", font= ('Times New Roman', 12, 'bold'), bg="white", bd=3, relief="solid")
    style. layout ("Treeview", [('Treeview.treearea', {'sticky': 'nswe'})])

def create_nav_button (parent, command, col, img_path1):
    img = tk. PhotoImage (file=img_path1)
    btn = tk. Button (parent, image=img, command=command, bd=0, bg="white")
    btn. grid (row=0, column=col, padx = 5)

def password_window ():
    pwd_win = tk. Toplevel(window)
    pwd_win. protocol ("WM_DELETE_WINDOW", on_close)
    correct_passwords = [f"{i:03}" for i in range (1, 60)]
    pwd_var = tk. StringVar (pwd_win)
    attempts = 0
    last_attempt_time = datetime.min
    def on_ok ():
        nonlocal attempts, last_attempt_time
        if datetime.now () - last_attempt_time > timedelta(minutes=1):  attempts = 0
            enable_entry_and_buttons ()
```

```
            last_attempt_time = datetime.now ()
        password = pwd_var.get ()
        if password in correct_passwords and 0 <= (index: = int(password) - 1) < len(ergazomenoi):
            name = ergazomenoi[index][1]
            welcome_label.config (text=f"Καλώς ήρθες {name}!")
            pwd_win. destroy ()
        else:
            attempts += 1
            if attempts >= 3:
                messagebox. showerror ("Κλείδωμα", "Προσπαθήστε ξανά σε 1 λεπτό")

    label = tk. Label (pwd_win, text="Εισάγετε κωδικό χρήστη", font= ("Times New Roman", 20))
    entry = tk. Entry (pwd_win, textvariable=pwd_var, show="*", width=20, relief="solid", font = ("Times New Roman", 20))
    entry. focus_set ()

    pwd_win. bind('<Return>', lambda event=None: on_ok ())

    button_one = tk. Frame(pwd_win)
    button_one. pack(pady=10)
    ok_btn = tk. Button (button_one, text="OK", command=on_ok, bd=1, relief="solid", font= ("Times New Roman", 20))
    ok_btn. grid (row=0, column=0)
    cancel_btn = tk. Button (button_one, text="Ακύρωση", command=lambda: pwd_var.set (''), bd=1, relief="solid"))
    cancel_btn. grid (row=0, column=1, padx=10)

    portal_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\portal.png'
    portal_image = tk. PhotoImage(file=portal_path)
    portal_label = tk. Label (pwd_win, image=portal_image)
    portal_label. image = portal_image
    symbol_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\symbol.png'
    symbol_image = tk. PhotoImage(file=symbol_path)
    pwd_win. iconphoto (True, symbol_image)

def disconnect_user ():
    if messagebox. askokcancel ("Αποσύνδεση", "Θέλετε σίγουρα να αποσυνδεθείτε;"):
        global current_user_name
        current_user_name = ""
        window. withdraw ()
        password_window ()

def button_click(event):
    clicked_text = event. widget. cget ("text")
    window. withdraw ()

    submenus = {
        "Αποθήκη": create_apothiki_submenu,
        "Εκπαιδεύσεις": create_ekpaideuseis_submenu,
        "Εξαρτήματα": create_eksartimata_submenu,
        "Εργαζόμενοι": create_ergazomenoi_submenu,
        "Μηχανές": create_mixanes_submenu,
        "Παραγγελίες": create_paraggelies_submenu,
        "Παραγωγή": create_paragogi_submenu,
```

```
        "Πελάτες": create_pelates_submenu,
        "Προμηθευτές": create_promitheutes_submenu,
        "Συντήρηση": create_suntirisi_submenu,
        "Χρήση": create_xrisi_submenu,
        "Συμμετοχές": create_summetoxes_submenu}

    function_to_execute = submenus.get(clicked_text)
    if function_to_execute:
        function_to_execute ()

def go_home ():
    while window_stack:
        top_window = window_stack.pop ()
        top_window. withdraw ()
    window. state ('zoomed')
    window. geometry ("")

def create_apothiki_submenu ():
    apothiki_submenu = create_submenu("Αποθήκη")

    def save_data ():
        try:
            with sqlite3.connect('erp.db') as conn:
                conn. execute ("BEGIN TRANSACTION")
                    if not (uliko and kodikos_ulikou and monada_metrisis and diathesimi_posotita and elaxisti_posotita):
                        messagebox.showwarning("Προσοχή", "Συμπληρώστε όλα τα πεδία")
                        return
                    c.execute ('"SELECT id FROM Αποθήκη WHERE id=?"', (str(row_entries['id']),))
                    existing_record = c. fetchone ()
                    if existing_record:
                        c.execute('"UPDATE Αποθήκη SET Υλικό=?, Κωδικός_υλικού=?, Μονάδα_μέτρησης=?,
                         Διαθέσιμη_ποσότητα=?, Ελάχιστη_ποσότητα=?, Κωδικός_προμηθευτή=? WHERE id=?"',
                    else:
                        c.execute ('"INSERT INTO Αποθήκη
                                (id, Υλικό, Κωδικός_υλικού, Μονάδα_μέτρησης, Διαθέσιμη_ποσότητα, Ελάχιστη_ποσότητα,
                conn.execute("COMMIT")
                messagebox.showinfo("", "Τα δεδομένα αποθηκεύτηκαν με επιτυχία")
        except sqlite3.Error as e:
            messagebox. showerror ("Error", f"An error occurred: {e}")

    nav_frame = tk. Frame (apothiki_submenu, bg="white")
    nav_frame. pack (pady=5, padx=5, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    create_nav_button (nav_frame, save_data, 1, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\save.png')
    input_frame = tk. Frame (input_canvas, bg="white")
    input_canvas. create_window ((10, 0), window=input_frame, anchor='nw')
    input_frame. bind("<Configure>", lambda e: input_canvas. configure (scrollregion=input_canvas. bbox("all")))

    labels = ["Υλικό", "Κωδικός υλικού", "Μονάδα μέτρησης", "Διαθέσιμη ποσότητα", "Ελάχιστη ποσότητα", "Κωδικός
προμηθευτή"]
```

```
entry_widths_by_label={"Υλικό": 25, "Κωδικός υλικού":15, "Μονάδα μέτρησης":15, "Διαθέσιμη ποσότητα":20,"Ελάχιστη ποσότητα":20, "Κωδικός προμηθευτή":15}

rows = []

def toggle_edit (row_entries: dict) -> None:
    current_state = row_entries.get ('state', '')
    states = {'': ('readonly', 'disabled', 'viewing'), 'viewing': ('readonly', 'normal', 'editing'), 'editing': ('disabled', 'disabled')}
    new_state_combobox, new_state_entry, new_state = states.get (current_state, ('disabled', 'disabled', 'viewing'))
    row_entries['state'] = new_state

def clear_grid ():
    for entries in rows:
        for widget in entries. values ():
            if isinstance (widget, (tk. Entry, ttk. Combobox, tk. Button)):
                widget. grid_forget ()

def check_shortage(row_entries):
    diathesimi_posotita_entry = row_entries.get ("Διαθέσιμη ποσότητα")
    elaxisti_posotita_entry = row_entries.get ("Ελάχιστη ποσότητα")
    if diathesimi_posotita_str nd elaxisti_posotita_str:
        diathesimi_posotita = int(diathesimi_posotita_str)
        elaxisti_posotita = int(elaxisti_posotita_str)
    else:
            for widget in input_frame. grid_slaves ():
                if widget. grid_info () ['column'] == 20 and widget. grid_info () ['row'] = = rows. index (row_entries) + 2

def add_row (position2=None, data=None):
    entries = {'state': 'viewing', 'id': uuid. Uuid 4 ()}
        if current_label = = "Υλικό":
            combobox_values_uliko = [item [0] for item in ulika]
            current_entry = ttk. Combobox (input_frame, values=combobox_values_uliko, state="disabled", justify="center")
            current_entry.set ("")
            current_entry. bind("<<ComboboxSelected>>", update_kodikos_ulikou)
        elif current_label = = "Μονάδα μέτρησης":
            combobox_values_monada = ["kg", "τεμάχια"]
            current_entry = ttk. Combobox (input_frame, values=combobox_values_monada, state="disabled", justify="center")
            current_entry.set ("")
        elif current_label == "Κωδικός προμηθευτή":
            combobox_values_promitheutes = [item [1] for item in promitheutes]
            current_entry = ttk. Combobox (input_frame, values=combobox_values_promitheutes, state="disabled")
            current_entry.set ("")
        elif current_label in ["Διαθέσιμη ποσότητα", "Ελάχιστη ποσότητα"]:
            vcmd = input_frame. register(validate_numeric_input)
            current_entry = tk. Entry (input_frame, validate="key", validate command = (vcmd, '%d', '%S'), state="disabled")
        else:
            current_entry = tk. Entry (input_frame, font = ('Times New Roman', 12), width=width_value, state="disabled")

    plus_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\plus.png'
    plus_image = tk. PhotoImage (file=plus_path)
    entries ['+'] = tk. Button (input_frame, image=plus_image, command= partial (add_row, entries), bd=0, bg="white")
```

```
    entries ['+']. image = plus_image

    edit_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\edit.png'
    edit_image = tk. PhotoImage (file=edit_path)
    entries['Edit'] = tk. Button (input_frame, image=edit_image, command= partial (toggle_edit, entries), bd=0, bg="white")
    entries['Edit']. image = edit_image

    delete_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\delete.png'
    delete_image = tk. PhotoImage (file=delete_path)
    entries['X'] = tk. Button (input_frame, image=delete_image, command=lambda row_id=entries['id']: delete_row(entries)
    entries['X']. image = delete_image

    entries ["Διαθέσιμη ποσότητα"]. Bind ('<KeyRelease>', lambda event, entry=entries: check_shortage2(entry))
    entries ["Ελάχιστη ποσότητα"]. Bind ('<KeyRelease>', lambda event, entry=entries: check_shortage2(entry))

    if data:
        entries ['id'] = data [0]
        entries ["Υλικό"]. insert (0, data [1])
        entries ["Κωδικός υλικού"]. insert (0, data [2])
        entries ["Μονάδα μέτρησης"]. insert (0, data [3])
        entries ["Διαθέσιμη ποσότητα"]. insert (0, data [4])
        entries ["Ελάχιστη ποσότητα"]. insert (0, data [5])
        entries ["Κωδικός προμηθευτή"]. insert (0, data [6])

    if position2:
        try:
            idx2 = rows. index (position2)
            rows. insert (idx2 + 1, entries)
    else:
        rows. append (entries)
    redraw_rows ()

def delete_row(entries):
    response = messagebox. Askyesno ("Προσοχή", "Είστε σίγουροι ότι θέλετε να διαγραφεί;")
        if row_id:
            c.execute ("DELETE FROM Αποθήκη WHERE id =?", (str_row_id,))
        except Exception as e:
            messagebox. showerror ("Error", f"An error occurred: {e}")
        finally:
        for widget in input_frame. grid_slaves ():
            if widget. grid_info () ['column'] == 20 and widget. grid_info () ['row'] == rows. index(entries) + 2:
        rows. remove(entries)
        redraw_rows ()

def redraw_rows ():
    for idx3, entries in enumerate(rows):
        for label_idx, current_label in enumerate(labels):
            entries[current_label]. grid (row=idx3 + 2, column=3 * label_idx + 3, sticky='nsew', padx = (5, 2), pady=2, ipadx=10)
                label = tk. Label (input_frame, text=current_label, bg="white", font = ('Times New Roman', 12, 'bold'))
                label. grid (row=0, column=3 * label_idx + 3, sticky='ns', columnspan=2, pady = 2, padx = (5, 2))
        entries['Edit']. grid (row=idx3 + 2, column=1, padx=2)
```

```
            entries['X']. grid (row=idx3 + 2, column=2)
            check_shortage(entries)

    def add_initial_row_if_empty ():
    records = fetch_data_apothiki ()
    for record in records:
        add_row(data=record)

    window_stack. Append (apothiki_submenu)

def create_ekpaideuseis_submenu ():
    ekpaideuseis_submenu = create_submenu("Εκπαιδεύσεις")

    def save_data ():
        with sqlite3.connect('erp.db') as conn:
            if not all(cleaned_data[key] for key in ['Τίτλος', 'Κωδικός', 'Εκπαιδευτής', 'Ημερομηνία']):
                messagebox.showwarning("Προσοχή", "Συμπληρώστε όλα τα πεδία")
            else:
                c.execute ('''INSERT INTO Εκπαιδεύσεις
                        (id, Τίτλος, Κωδικός, Εκπαιδευτής, Ημερομηνία) VALUES (?, ?, ?, ?, ?)''',
                conn.commit()
            messagebox.showinfo("", "Τα δεδομένα αποθηκεύτηκαν με επιτυχία")

    nav_frame = tk. Frame (ekpaideuseis_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    create_nav_button (nav_frame, save_data, 1, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\save.png')
    input_canvas. create_window ((10, 0), window=input_frame, anchor='nw')
    input_frame. bind ("<Configure>", lambda e: input_canvas. Configure (scrollregion=input_canvas. bbox ("all")))

    labels = ["Τίτλος", "Κωδικός", "Εκπαιδευτής", "Ημερομηνία"]
    entry_widths_by_label = {"Τίτλος": 40, "Κωδικός": 15, "Εκπαιδευτής": 25, "Ημερομηνία": 10}

    rows = []

    def toggle_edit (row_entries: dict) -> None:
        new_state = 'normal' if row_entries.get ('state', '') == 'viewing' else 'disabled'
        row_entries['state'] = 'editing' if row_entries.get ('state', '') == 'viewing' else 'viewing'

    def clear_grid ():
        for entries in rows:
            for widget in entries. Values ():
                if isinstance (widget, (tk. Entry, tk. Button)):
                    widget. grid_forget ()

    def add_row (position2=None, _added_by_plus_button=False, data=None):
        entries = {'state': 'viewing'}
            if current_label == "Ημερομηνία":
                widget = DateEntry (input_frame, font = ('Times New Roman', 12), state="disabled", date_pattern='dd/mm/yyyy')
                entries[current_label] = widget
            else:
                widget = tk. Entry (input_frame, font = ('Times New Roman', 12), state="disabled", justify="center")
```

```
    edit_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\edit.png'
    edit_image = tk. PhotoImage(file=edit_path)
    entries['Edit'] = tk. Button (input_frame, image=edit_image, command=partial (toggle_edit, entries), bd=0, bg="white")
    entries['Edit']. image = edit_image

    plus_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\plus.png'
    plus_image = tk. PhotoImage(file=plus_path)
    entries ['+'] = tk. Button (input_frame, image=plus_image, command=partial (add_row, entries), bd=0, bg="white")
    entries ['+']. image = plus_image

    delete_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\delete.png'
    delete_image = tk. PhotoImage(file=delete_path)
    entries['Delete'] = tk. Button (input_frame, image=delete_image, command=lambda row_id=entries['id']: delete_row(entries)
    entries['Delete']. image = delete_image

    if data:
        entries ['id'] = data [0]
        entries ["Τίτλος"]. insert (0, data [1])
        entries ["Κωδικός"]. insert (0, data [2])
        entries ["Εκπαιδευτής"]. insert (0, data [3])
        entries ["Ημερομηνία"]. set_date (data [4])
    if position2:
        idx2 = rows. index(position2)
    else:
        rows. append(entries)

def delete_row(entries):
    response = messagebox.askyesno("Προσοχή", "Είστε σίγουροι ότι θέλετε να διαγραφεί ;")
    if response:
        conn = sqlite3.connect('erp.db')
        if row_id:
            c.execute ("DELETE FROM Εκπαιδεύσεις WHERE id =?", (str_row_id,))
        except Exception as e:
            messagebox. showerror ("Error", f"An error occurred: {e}")

def redraw_rows ():
    for idx3, entries in enumerate(rows):
        for label_idx, current_label in enumerate(labels):
            entries[current_label]. Grid (row=idx3 + 2, column=3 * label_idx + 3, sticky='nsew', columnspan=2)
            if 'Edit' in entries:
                entries['Edit']. grid (row=idx3 + 2, column=1)
            else:
                entries['Delete']. grid_remove ()

def add_initial_row_if_empty ():
    if not rows and not fetch_data_ekpaideuseis ():
records = fetch_data_ekpaideuseis ()
for record in records:

window_stack. append(ekpaideuseis_submenu)
```

```
def create_eksartimata_submenu ():
    def update_mixani (_):
        selected_eksartima = eksartima_combobox.get ()
        for item in kaloupia:
            if item [1] == selected_eksartima:
                mixani_entry. delete (0, "end")
                mixani_entry. insert (0, item [3])

    eksartimata_submenu = create_submenu("Εξαρτήματα")
    nav_frame = tk. Frame (eksartimata_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')

    tree = ttk. Treeview (container_frame,
                columns = ("Κωδικός εξαρτήματος", "Κράμα", "Κοιλότητες", "Βάρος χυτού (gr)", "Βάρος πατήματος (gr)",
"Καλούπι χύτευσης", "Κοπτικό καλούπι"), show="headings", style="Treeview")

    columns_info = {"Κωδικός εξαρτήματος": 170, "Κράμα": 120, "Κοιλότητες": 110, "Βάρος χυτού (gr)": 130, "Βάρος πατήματος
(gr)": 180, "Καλούπι χύτευσης": 170, "Κοπτικό καλούπι": 150}

    for column, width in columns_info. Items ():
        tree. Heading (column, text=column)
        tree. Column (column, anchor="center", width=width, stretch=False)
    tree. Configure (yscrollcommand=v_scrollbar.set, xscrollcommand=h_scrollbar.set)
    eksartima_values = sorted (set (item [1] for item in kaloupia))
    eksartima_label = tk. Label (nav_frame, text="Εξάρτημα:", bg="white")
    mixani_entry = tk. Entry (nav_frame, justify="center")

    indices_to_retrieve = [2, 4, 5, 6, 7, 8, 9]

    for item in kaloupia:
        values_to_insert = [item[i] for i in indices_to_retrieve]
        tree. Insert ("", "end", values=values_to_insert)

    window_stack. append(eksartimata_submenu)

def create_ergazomenoi_submenu ():
    ergazomenoi_submenu = create_submenu("Εργαζόμενοι")
    nav_frame = tk. Frame (ergazomenoi_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    tree_frame. pack (padx=10, pady=10, side="left", fill="both", expand=True)
    tree = ttk. Treeview (tree_frame, columns = ("Κωδικός", "Ονοματεπώνυμο", "Θέση", "Ημ/νία πρόσληψης"), show='headings')

    headings = {"Κωδικός": ("Κωδικός", 80), "Ονοματεπώνυμο": ("Ονοματεπώνυμο", 250), "Θέση": ("Θέση", 200),"Ημ/νία
πρόσληψης": ("Ημ/νία πρόσληψης", 150)}

    scrollbar = ttk. Scrollbar (tree_frame, orient="vertical", command = tree. yview)
    scrollbar. Pack (side="right", fill="y")
    tree. Configure (yscrollcommand=scrollbar.set)

    window_stack. Append (ergazomenoi_submenu)
```

```
def create_mixanes_submenu ():
    mixanes_submenu = create_submenu("Μηχανές")
    set_common_style ()
    nav_frame = tk. Frame (mixanes_submenu, bg="white")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    tree = ttk. Treeview (tree_frame, columns = ("Κωδικός", "Ονομασία", "Μηχανή"), show="headings")
    columns_info = [("Κωδικός", "center", 150), ("Ονομασία", "center", 150), ("Μηχανή", "center", 150)]

    for col_name, col_anchor, col_width in columns_info:
        tree. heading (col_name, text=col_name)
        tree. column (col_name, anchor=col_anchor, width=col_width)

    scrollbar = ttk. Scrollbar (tree_frame, orient="vertical", command= tree. yview)
    scrollbar. Pack (side="right", fill="y")
    tree. configure(yscrollcommand=scrollbar.set)
    for item in mixanes:
        tree. insert ("", "end", values=item)
    tree. pack (padx=10, pady=10, side="left", fill="y")

    window_stack. append(mixanes_submenu)

def create_paraggelies_submenu ():
    paraggelies_submenu = create_submenu("Παραγγελίες")

    def save_data ():
        try:
            conn = sqlite3.connect('erp.db')
            c = conn. cursor ()
                if not all ((pelatis, exartima, temaxia, kodikos_paraggelias, partida, imerominia)):
                    messagebox.showwarning("Προσοχή", "Συμπληρώστε όλα τα πεδία")
                else:
                    id1 = str (uuid. uuid 4 ())
                    c.execute ('''INSERT INTO Παραγγελίες (id, Πελάτης, Εξάρτημα, Τεμάχια, Κωδικός_παραγγελίας, Παρτίδα,
Ημερομηνία) VALUES (?, ?, ?, ?, ?, ?, ?)''',
        except sqlite3.Error as e:
            messagebox. showerror ("Database Error", f"An error occurred: {str(e)}")
        finally:
            if conn:
        messagebox.showinfo("", "Τα δεδομένα αποθηκεύτηκαν με επιτυχία")

    nav_frame = tk. Frame (paraggelies_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    create_nav_button (nav_frame, save_data, 1, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\save.png')
    input_canvas. create_window ((10, 0), window=input_frame, anchor='nw')
    input_frame. Bind ("<Configure>", lambda e: input_canvas. configure (scrollregion=input_canvas. bbox ("all")))

    labels = ["Πελάτης", "Εξάρτημα", "Τεμάχια", "Κωδικός παραγγελίας", "Παρτίδα", "Ημερομηνία"]
    entry_widths_by_label = {"Πελάτης": 20, "Εξάρτημα": 20, "Τεμάχια": 10, "Κωδικός παραγγελίας": 20, "Παρτίδα": 10,
"Ημερομηνία": 10}
    for i, label_text in enumerate(labels):
```

```
    lbl = tk. Label (input_frame, text=label_text, bg="white", font = ('Times New Roman', 12, 'bold'))
    lbl. grid (row=0, column=3 * i + 3, sticky='nsew', columnspan=3, pady=5, padx = (0, 10))

rows = []

def toggle_edit (row_entries: dict) -> None:
    state = row_entries.get ('state', '')
    new_state = 'editing' if state == 'viewing' else 'viewing'
    new_state_combobox = 'readonly' if new_state == 'editing' else 'disabled'
    new_state_entry = 'normal' if new_state == 'editing' else 'disabled'

def clear_grid ():
    for entries in rows:
        for widget in entries. values ():
            if isinstance (widget, (tk. Entry, ttk. Combobox, tk. Button)):
                widget. grid_forget ()

def update_eksartima_combobox (pelatis_combobox, row_entries=None, triggered_by_pelatis=True):
    eksartima_combobox = row_entries.get("Εξάρτημα")
    eksartima_values = [item [1] for item in kaloupia if item [0] = = pelatis]

def add_row (position2=None, data=None):
        if current_label == "Πελάτης":
            combobox_values_pelatis = [item [0] for item in pelates]
            current_entry = ttk. Combobox (input_frame, values=combobox_values_pelatis, state="disabled", justify="center")
            current_entry. Bind ("<<ComboboxSelected>>", lambda event
        else:
            if current_label in ["Ημερομηνία"]:
                current_entry = DateEntry (input_frame, font = ('Times New Roman', 12), date_pattern='dd/mm/yyyy')

    plus_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\plus.png'
    plus_image = tk. PhotoImage (file=plus_path)
    entries ['+'] = tk. Button (input_frame, image=plus_image, command = partial (add_row, entries), bd=0, bg="white")
    entries ['+']. image = plus_image

    edit_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\edit.png'
    edit_image = tk. PhotoImage (file=edit_path)
    entries['Edit'] = tk. Button (input_frame, image=edit_image, command = partial (toggle_edit, entries), bd=0, bg="white")
    entries['Edit']. image = edit_image

    delete_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\delete.png'
    delete_image = tk. PhotoImage (file=delete_path)
    entries['Delete'] = tk. Button (input_frame, image=delete_image, command=lambda: delete_row(entries), bd=0, bg="white")
    entries['Delete']. image = delete_image
    if data:
        entries ['id'] = data [0]
        entries ["Πελάτης"]. set (data [1])
        entries ["Εξάρτημα"]. set (data [2])
        entries ["Τεμάχια"]. insert (0, data [3])
        entries ["Κωδικός παραγγελίας"]. insert (0, data [4])
        entries ["Παρτίδα"]. insert (0, data [5])
```

```
            entries["Ημερομηνία"]. Insert (0, data [6])

        if position2:
            idx2 = rows. index (position2)
            rows. insert (idx2 + 1, entries)
        else:
            rows. append (entries)
        redraw_rows ()

    def delete_row (row_entries: dict) -> None:
        response = messagebox.askyesno("Προσοχή", "Είστε σίγουροι ότι θέλετε να διαγραφεί ;")
        if response:
            row_id = row_entries.get('id')
            if row_id:
                c.execute ("DELETE FROM Παραγγελίες WHERE id=?", (row_id,))
            clear_grid ()
            redraw_rows ()

    def redraw_rows ():
        for idx3, entries in enumerate(rows):
            entries['Edit']. grid (row=idx3 + 2, column=1, pady=entry_padding_y)
            if idx3! = 0:
                entries['Delete']. Grid (row=idx3 + 2, column=2, pady=entry_padding_y)
            else:
                entries['Delete']. grid_forget ()

    def add_initial_row_if_empty ():
        if not rows and not fetch_data_paraggelies ():
    for record in records:
        add_row(data=record)

    window_stack. Append (paraggelies_submenu)


def create_paragogi_submenu ():
    paragogi_submenu = create_submenu("Παραγωγή")

    def save_data ():
        with sqlite3.connect('erp.db') as conn:
            for row1 in rows:
                row_id = row1.get('id')
                if not row_id:
                    row_id = str (uuid. uuid 4())

             if not (kodikos_paragogis and kodikos_paraggelias and mixani and xiristis and ora_enarksis and ora_liksis and temaxia
and skarta and imerominia):
                    messagebox.showwarning("Προσοχή", "Συμπληρώστε όλα τα πεδία")
                if existing:
                    c.execute ('"UPDATE Παραγωγή
                                SET Κωδικός_παραγωγής=?, Κωδικός_παραγγελίας=?, Μηχανή=?, Χειριστής=?, Ώρα_έναρξης=?,
                                Ώρα_λήξης=?, Τεμάχια=?, Σκάρτα=?, Ημερομηνία=?  WHERE id=?"',
                else:
```

```
                '''INSERT INTO Παραγωγή (id, Κωδικός_παραγωγής, Κωδικός_παραγγελίας, Μηχανή, Χειριστής,
                Ώρα_έναρξης, Ώρα_λήξης, Τεμάχια, Σκάρτα, Ημερομηνία) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)''',
            conn.commit()
        messagebox.showinfo("", "Τα δεδομένα αποθηκεύτηκαν με επιτυχία")

    nav_frame = tk. Frame (paragogi_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    create_nav_button (nav_frame, save_data, 1, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\save.png')
    input_frame. Bind ("<Configure>", lambda e: input_canvas. configure (scrollregion=input_canvas. bbox ("all")))

    labels = ["Κωδικός παραγωγής", "Κωδικός παραγγελίας", "Μηχανή", "Χειριστής", "Ώρα έναρξης", "Ώρα λήξης", "Τεμάχια",
"Σκάρτα", "Ημερομηνία"]
    entry_widths_by_label = {"Κωδικός παραγωγής": 15, "Κωδικός παραγγελίας": 15, "Μηχανή": 10, "Χειριστής": 20,
                             "Ώρα έναρξης": 10, "Ώρα λήξης": 10, "Τεμάχια": 10, "Σκάρτα": 10, "Ημερομηνία": 10}

    rows = []

    def toggle_edit (row_entries: dict) -> None:
        new_state_combobox = 'readonly' if row_entries.get ('state', '') == 'viewing' else 'disabled'
        new_state_entry = 'normal' if row_entries.get ('state', '') == 'viewing' else 'disabled'
        dateentry_state = row_entries.get ('state', '')! = 'viewing'
        row_entries['state'] = 'editing' if row_entries.get ('state', '') == 'viewing' else 'viewing'

    def clear_grid ():
        for entries in rows:
            for widget in entries. values ():
                if isinstance (widget, (tk. Entry, ttk. Combobox, tk. Button)):
                    widget. grid_forget ()

    def add_row (position2=None, data=None):
            if current_label == "Μηχανή":
                combobox_values_mixani = [item [1] for item in mixanes [:7] if item [1]]
                current_entry = ttk. Combobox (input_frame, values=combobox_values_mixani, state="disabled")
            elif current_label == "Χειριστής":
                combobox_values_xeiristis = []
                current_entry = ttk. Combobox (input_frame, values=combobox_values_xeiristis, state="disabled")
            elif current_label == "Κωδικός παραγωγής":
                conn = sqlite3.connect('erp.db')
                c = conn. cursor ()
                c.execute ("SELECT DISTINCT Κωδικός_παραγωγής FROM Χρήση")
                codes = c. fetchall ()
                combobox_values_kodikos = [code [0] for code in codes]
                current_entry = ttk. Combobox (input_frame, values=combobox_values_kodikos)
            elif current_label = = "Κωδικός παραγγελίας":
                conn = sqlite3.connect('erp.db')
                c = conn. cursor ()
                c.execute ("SELECT DISTINCT Κωδικός_παραγγελίας FROM Παραγγελίες")
                codes = c. fetchall ()
                combobox_values_kodikos = [code [0] for code in codes]
```

```
        else:
            current_entry = tk. Entry (input_frame, font = ('Times New Roman', 12), width=width_value, state="disabled")

    plus_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\plus.png'
    plus_image = tk. PhotoImage(file=plus_path)
    entries ['+'] = tk. Button (input_frame, image=plus_image, command = partial (add_row, entries), bd=0, bg="white")
    entries ['+']. image = plus_image

    edit_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\edit.png'
    edit_image = tk. PhotoImage (file=edit_path)
    entries['Edit'] = tk. Button (input_frame, image=edit_image, command = partial (toggle_edit, entries), bd=0, bg="white")
    entries['Edit']. image = edit_image


    delete_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\delete.png'
    delete_image = tk. PhotoImage (file=delete_path)
    entries['X'] = tk. Button (input_frame, image=delete_image, command=lambda row_id1=entries['id']: delete_row(entries))
    entries['X']. image = delete_image

    if data:
        entries ['id'] = data [0]
        entries ["Κωδικός παραγωγής"]. insert (0, data [1])
        entries ["Κωδικός παραγγελίας"]. insert (0, data [2])
        entries ["Μηχανή"]. insert (0, data [3])
        entries ["Χειριστής"]. insert (0, data [4])
        entries ["Ώρα έναρξης"]. insert (0, data [5])
        entries ["Ώρα λήξης"]. insert (0, data [6])
        entries ["Τεμάχια"]. insert (0, data [7])
        entries ["Σκάρτα"]. insert (0, data [8])
        entries ["Ημερομηνία"]. insert (0, data [9])

    if position2:
        try:
            idx2 = rows. index (position2)
            rows. insert (idx2 + 1, entries)
        except ValueError:
            rows. append (entries)
    else:
        rows. append (entries)

def delete_row(entries):
    response = messagebox. askyesno ("Προσοχή", "Είστε σίγουροι ότι θέλετε να διαγραφεί;")
    if response:
        conn = sqlite3.connect('erp.db')
        if row_id:
            str_row_id = str(row_id)
            try:
                c.execute ("DELETE FROM Παραγωγή WHERE id =?", (str_row_id,))
            except Exception as e:
                messagebox. Showerror ("Error", f"An error occurred: {e}")

def redraw_rows ():
```

```
        for idx3, entries in enumerate(rows):
            for label_idx, current_label in enumerate(labels):
                entries[current_label]. grid (row=idx3 + 2, column=3 * label_idx + 3, sticky='nsew', columnspan=2, padx = (0, 10))
                if idx3 = = 0:
                    label = tk. Label (input_frame, text=current_label, bg="white", font = ('Times New Roman', 12, 'bold'))
                    label. grid (row=0, column=3 * label_idx + 3, sticky='ns', columnspan=2, pady=2, padx = (0, 10))
            entries['Edit']. grid (row=idx3 + 2, column=1)

    def add_initial_row_if_empty ():
        if not rows and not fetch_data_paragogi ():
    records = fetch_data_paragogi ()
    for record in records:
        add_row(data=record)

    window_stack. Append (paragogi_submenu)

def create_pelates_submenu ():
    pelates_submenu = create_submenu("Πελάτες")
    nav_frame = tk. Frame (pelates_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    tree = ttk. Treeview (tree_frame, columns = ("Πελάτης", "Κωδικός πελάτη", "Έδρα"), show="headings")
    columns_info = [("Πελάτης", "center", 200), ("Κωδικός πελάτη", "center", 150), ("Έδρα", "center", 100)]

    for col_name, col_anchor, col_width in columns_info:
        tree. heading (col_name, text=col_name)
        tree. Column (col_name, anchor=col_anchor, width=col_width)

    scrollbar = ttk. Scrollbar (tree_frame, orient="vertical", command=tree. yview)
    scrollbar. Pack (side="right", fill="y")
    tree. Configure (yscrollcommand=scrollbar.set)

  pelates. Sort (key=lambda x: x [0])
    for item in pelates:
        tree. Insert ("", "end", values=item)

    window_stack. Append (pelates_submenu)

def create_promitheutes_submenu ():
    promitheutes_submenu = create_submenu("Προμηθευτές")
    nav_frame = tk. Frame (promitheutes_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    tree = ttk. Treeview (tree_frame, columns = ("Επωνυμία", "Κωδικός", "Προϊόντα", "Email"), show="headings")
    columns_info = [("Επωνυμία", "center", 200), ("Κωδικός", "center", 150), ("Προϊόντα", "center", 200), ("Email", "center", 250)]

    for col_name, col_anchor, col_width in columns_info:
        tree. Heading (col_name, text=col_name)
        tree. Column (col_name, anchor=col_anchor, width=col_width)

    scrollbar = ttk. Scrollbar (tree_frame, orient="vertical", command = tree. yview)
    scrollbar. Pack (side="right", fill="y")
```

```
    tree. configure (yscrollcommand=scrollbar.set)

    for item in promitheutes:
      tree. Insert ("", "end", values=item)
       tree. Pack (pady=20, padx=10, side='left', fill="y")

    window_stack. Append (promitheutes_submenu)

def create_summetoxes_submenu ():
    summetoxes_submenu = create_submenu("Συμμετοχές")

    def save_data ():
                if existing_record:
                   c.execute ('"UPDATE Συμμετοχές SET Συμμετέχοντες=? WHERE id=?"',
                         (summetexontes, existing_record [0]))
                else:
                   id1 = uuid. uuid 4()
                   c.execute ('"INSERT INTO Συμμετοχές
                            (id, Κωδικός_εκπαίδευσης, Συμμετέχοντες) VALUES (?,  ?, ?)"',
                           (str(id), kodikos_ekpaideusis, summetexontes))
            messagebox. showinfo ("", "Τα δεδομένα αποθηκεύτηκαν με επιτυχία")
        except sqlite3.Error as e:
            messagebox. Showerror ("Error", f"An error occurred: {e}")

    nav_frame = tk. Frame (summetoxes_submenu, bg="white")
    nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
    create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
    create_nav_button (nav_frame, save_data, 1, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\save.png')
    input_frame. bind ("<Configure>", lambda e: input_canvas. configure (scrollregion=input_canvas. bbox ("all")))

    labels = ["Κωδικός εκπαίδευσης", "Συμμετέχοντες"]
    entry_widths_by_label = {"Κωδικός εκπαίδευσης": 20, "Συμμετέχοντες": 30}

    rows = []

    def clear_grid ():
        for entries in rows:
            for widget in entries. values ():
                if isinstance (widget, (tk. Entry, ttk. Combobox, tk. Button)):
                    widget. grid_forget ()

    def toggle_selection_color(listbox):
        selected_indices = listbox. Curselection ()
            if i in selected_indices:
                listbox. Itemconfig (i, {'bg': 'white', 'fg': 'black'})
            else:
                listbox. Itemconfig (i, {'bg': 'white', 'fg': 'black'})

    def add_row (position2=None, data=None):
        entries = {'state': 'viewing', 'id': uuid. uuid 4()}
            if current_label = = "Κωδικός εκπαίδευσης":
                ekpaideusi_values = [item [2] for item in fetch_data_ekpaideuseis ()]
```

```
            entry_frame. grid (row=1, column=2, sticky="nsew", padx=5)
            current_entry = ttk. Combobox (entry_frame, values=ekpaideusi_values, justify="center", height=5)
        elif current_label == "Συμμετέχοντες":
            summetexontes_values = [item [1] for item in ergazomenoi if item [1]]
            current_entry = tk. Listbox (input_frame, selectmode='multiple', font = ('Times New Roman', 12), height=30)

    if data:
        entries ['id'] = data [0]
        entries ["Κωδικός εκπαίδευσης"]. insert (0, data [1])
    if position2:
            idx2 = rows. index (position2)
            rows. insert (idx2, entries)
    else:
        rows. append (entries)
    redraw_rows ()

  def redraw_rows ():
    for idx3, entries in enumerate(rows):
        label_ke = tk. Label (input_frame, text="Κωδικός εκπαίδευσης", bg="white", font = ('Times New Roman', 12, 'bold'))
        label_ke. Grid (row=0, column=2, sticky='nsew')
        label_summetexontes = tk. Label (input_frame, text="Συμμετέχοντες", bg="white", font = ('Times New Roman', 12, 'bold'))
        label_summetexontes. Grid (row=0, column=3, sticky='nsew')
        toggle_selection_color(entries["Συμμετέχοντες"])

  def add_initial_row_if_empty ():
    if not rows and not fetch_data_summetoxes ():
  records = fetch_data_summetoxes ()
  for record in records:
    add_row(data=record)

  window_stack. Append (summetoxes_submenu)

def create_suntirisi_submenu ():
  suntirisi_submenu = create_submenu("Συντήρηση")

  def save_data ():
            if existing_record:
                c.execute ('"UPDATE Συντήρηση SET Κωδικός_καλουπιού =? Είδος_επισκευής=?, Συντηρητής=?,
Ημερομηνία=?
            else:
                id1 = uuid. uuid 4()
                c.execute ('"INSERT INTO Συντήρηση
                        (id, Κωδικός_καλουπιού, Είδος_επισκευής, Συντηρητής, Ημερομηνία)  VALUES (?, ?, ?, ?, ?)"',
        messagebox.showinfo("", "Τα δεδομένα αποθηκεύτηκαν με επιτυχία")
    except sqlite3.Error as e:
        messagebox. Showerror ("Error", f"An error occurred: {e}")

  nav_frame = tk. Frame (suntirisi_submenu, bg="white")
  nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
  create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
  create_nav_button (nav_frame, save_data, 1, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\save.png')
```

```
input_frame. Bind ("<Configure>", lambda e: input_canvas. configure (scrollregion=input_canvas. bbox ("all")))

labels = ["Κωδικός καλουπιού", "Είδος επισκευής", "Συντηρητής", "Ημ/νία"]
entry_widths_by_label = {"Κωδικός καλουπιού": 20, "Είδος επισκευής": 30, "Συντηρητής": 15, "Ημ/νία": 10}

rows = []

def toggle_edit (row_entries: dict) -> None:
    current_state = row_entries. get ('state', '')
    states = {'': ('readonly', 'disabled', 'viewing'), 'viewing': ('readonly', 'normal', 'editing') 'editing': ('disabled', 'disabled')}
    new_state_combobox, new_state_entry, new_state = states. Get (current_state, ('disabled', 'disabled', 'viewing'))
    row_entries['state'] = new_state
def clear_grid ():
        for widget in entries. values ():
            if isinstance (widget, (tk. Entry, ttk. Combobox, tk. Button)):
                widget. grid_forget ()

def add_row (position2=None, data=None):
        if current_label == "Συντηρητής":
            combobox_values_xeiristis = []
            current_entry = ttk. Combobox (input_frame, values=combobox_values_xeiristis, state="disabled", justify="center")
        elif current_label == "Ημ/νία":
            current_entry = DateEntry (input_frame, font = ('Times New Roman', 12), width=width_value, justify="center")
        elif current_label == "Κωδικός καλουπιού":
            kaloupia_values = [item [8] for item in kaloupia]
            current_entry = ttk. Combobox (input_frame, values=kaloupia_values)
        else:
            current_entry = tk. Entry (input_frame, font = ('Times New Roman', 12), width=width_value, state="disabled")
        entries[current_label] = current_entry

    plus_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\plus.png'
    plus_image = tk. PhotoImage (file=plus_path)
    entries ['+'] = tk. Button (input_frame, image=plus_image, command = partial (add_row, entries), bd=0, bg="white")
    entries ['+']. image = plus_image


    edit_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\edit.png'
    edit_image = tk. PhotoImage (file=edit_path)
    entries['Edit'] = tk. Button (input_frame, image=edit_image, command = partial (toggle_edit, entries), bd=0, bg="white")
    entries['Edit']. image = edit_image

    delete_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\delete.png'
    delete_image = tk. PhotoImage (file=delete_path)
    entries['X'] = tk. Button (input_frame, image=delete_image, command=lambda row_id=entries['id']: delete_row(entries))
    entries['X']. image = delete_image

    if data:
        entries ['id'] = data [0]
        entries ["Κωδικός καλουπιού"]. insert (0, data [1])
        entries ["Είδος επισκευής"]. insert (0, data [2])
        entries ["Συντηρητής"]. set (data [3])
        entries ["Ημ/νία"]. set_date (data [4])
```

```
        if position2:
            try:
                idx2 = rows. index (position2)
             rows. insert (idx2 + 1, entries)
            except ValueError:
                rows. append (entries)
        else:
            rows. append (entries)
        redraw_rows ()

    def delete_row(entries):
        response = messagebox.askyesno("Προσοχή", "Είστε σίγουροι ότι θέλετε να διαγραφεί ;")
        if response:
            conn = sqlite3.connect('erp.db')
            if row_id:
                str_row_id = str(row_id)
                 c.execute ("DELETE FROM Συντήρηση WHERE id =?", (str_row_id,))
                except Exception as e:
                    messagebox. Showerror ("Error", f"An error occurred: {e}")

    def redraw_rows ():
        for idx3, entries in enumerate(rows):
            for label_idx, current_label in enumerate(labels):
                entries[current_label]. grid (row=idx3 + 2, column=3 * label_idx + 3, sticky='nsew', columnspan=2)
                if idx3 = = 0:
                    label = tk. Label (input_frame, text=current_label, bg="white", font = ('Times New Roman', 12, 'bold'))
                    label. grid (row=0, column=3 * label_idx + 3, sticky='ns', columnspan=2, pady=2, padx = (0, 10))
            entries['Edit']. grid (row=idx3 + 2, column=1)

    def add_initial_row_if_empty ():
        if not rows and not fetch_data_suntirisi ():
    records = fetch_data_suntirisi ()
    for record in records:
        add_row(data=record)

    window_stack. Append (suntirisi_submenu)
def create_xrisi_submenu ():
    xrisi_submenu = create_submenu("Χρήση")

    def save_data ():
                if existing_record:
                    c.execute ('"UPDATE Χρήση SET Κωδικός_παραγωγής =? Κωδικός_υλικού=?, Ποσότητα=? WHERE id=?"',
                            (kodikos_paragogis, kodikos_ulikou, posotita, str(row_entries['id'])))
                else:
                    c.execute ('"INSERT INTO Χρήση
                                (id, Κωδικός_παραγωγής, Κωδικός_υλικού, Ποσότητα) VALUES (?, ?, ?, ?)"',
                    c.execute ('"SELECT Διαθέσιμη_ποσότητα FROM Αποθήκη WHERE Κωδικός_υλικού=?"', (kodikos_ulikou,))
                    current_quantity = int (c. fetchone () [0])
                    if existing_record:
                        c.execute ('"SELECT Ποσότητα FROM Χρήση WHERE id=?"', (str(row_entries['id']),))
```

```
                old_posotita = int (c. fetchone () [0])
                difference = old_posotita - posotita
                new_quantity = current_quantity + difference
            else:
                new_quantity = current_quantity - posotita
        messagebox.showinfo("", "Τα δεδομένα αποθηκεύτηκαν με επιτυχία")
    except sqlite3.Error as e:
        messagebox. Showerror ("Error", f"An error occurred: {e}")

nav_frame = tk. Frame (xrisi_submenu, bg="white")
nav_frame. Pack (pady=5, padx=10, anchor='nw', fill="x")
create_nav_button (nav_frame, go_home, 0, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\home.png')
create_nav_button (nav_frame, save_data, 1, 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\save.png')
input_frame. bind ("<Configure>", lambda e: input_canvas. configure (scrollregion=input_canvas. bbox ("all")))

labels = ["Κωδικός παραγωγής", "Κωδικός υλικού", "Ποσότητα"]
entry_widths_by_label = {"Κωδικός παραγωγής": 25, "Κωδικός υλικού": 15, "Ποσότητα": 15}

rows = []

def toggle_edit (row_entries: dict) -> None:
    current_state = row_entries. get ('state', '')
    states = {'': ('readonly', 'disabled', 'viewing'), 'viewing': ('readonly', 'normal', 'editing'), 'editing': ('disabled', 'disabled')}
    new_state_combobox, new_state_entry, new_state = states.get (current_state, ('disabled', 'disabled', 'viewing'))
    row_entries['state'] = new_state

def clear_grid ():
    for entries in rows:
        for widget in entries. values ():
            if isinstance (widget, (tk. Entry, ttk. Combobox, tk. Button)):
                widget. grid_forget ()

def add_row (position2=None, data=None):
        if current_label == "Κωδικός υλικού":
            conn = sqlite3.connect('erp.db')
            c.execute ("SELECT DISTINCT Κωδικός_υλικού FROM Αποθήκη")
            combobox_values_kodikos = [code [0] for code in codes]
            current_entry = ttk. Combobox (input_frame, values=combobox_values_kodikos, width=width_value, state="disable")
            current_entry. Bind ("<<ComboboxSelected>>", lambda event,
        elif current_label == "Ποσότητα":
            current_entry = tk. Entry (input_frame, font = ('Times New Roman', 12), width=width_value, state="disabled)
        else:
            current_entry = tk. Entry (input_frame, font = ('Times New Roman', 12), width=width_value, state="disabled")
        entries[current_label] = current_entry

    plus_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\plus.png'
    plus_image = tk. PhotoImage (file=plus_path)
    entries ['+'] = tk. Button (input_frame, image=plus_image, command = partial (add_row, entries), bd=0, bg="white")
    entries ['+']. image = plus_image

    edit_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\edit.png'
```

```
    edit_image = tk. PhotoImage (file=edit_path)
    entries['Edit'] = tk. Button (input_frame, image=edit_image, command = partial (toggle_edit, entries), bd=0, bg="white")
    entries['Edit']. image = edit_image

    delete_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\delete.png'
    delete_image = tk. PhotoImage (file=delete_path)
    entries['X'] = tk. Button (input_frame, image=delete_image, command=lambda row_id=entries['id']: delete_row(entries))
    entries['X']. image = delete_image

    if data:
        entries ['id'] = data [0]
        entries ["Κωδικός παραγωγής"]. insert (0, data [1])
        entries ["Κωδικός υλικού"]. insert (0, data [2])
        entries ["Ποσότητα"]. insert (0, data [3])

    if position2:
        try:
            idx2 = rows. index (position2)
            rows. insert (idx2 + 1, entries)
        except ValueError:
            rows. append (entries)
    else:
        rows. append (entries)
    redraw_rows ()

def update_quantity (combobox_entry, row_entries):
    selected_value = combobox_entry.get ()
    if selected_value:
        conn = sqlite3.connect('erp.db')
        c.execute ("SELECT Διαθέσιμη_ποσότητα FROM Αποθήκη WHERE Κωδικός_υλικού =?", (selected_value,))
        max_quantity = c. fetchone () [0]

def delete_row(entries):
    response = messagebox. Askyesno ("Προσοχή", "Είστε σίγουροι ότι θέλετε να διαγραφεί;")
    if response:
        conn = sqlite3.connect('erp.db')
        row_id = entries.get('id')
        if row_id:
            str_row_id = str(row_id)
            try:
                c.execute ("DELETE FROM Χρήση WHERE id =?", (str_row_id,))
            except Exception as e:
                messagebox. Showerror ("Error", f"An error occurred: {e}")

def redraw_rows ():
    for idx3, entries in enumerate(rows):
        for label_idx, current_label in enumerate(labels):
            entries[current_label]. grid (row=idx3 + 2, column=3 * label_idx + 3, sticky='nsew', columnspan=2, padx = (0, 10))
            if idx3 = = 0:
                label = tk. Label (input_frame, text=current_label, bg="white", font = ('Times New Roman', 12, 'bold'))
                label. grid (row=0, column=3 * label_idx + 3, sticky='ns', columnspan=2, pady=2, padx = (0, 10))
```

```
        entries['Edit']. grid (row=idx3 + 2, column=1)

    def add_initial_row_if_empty ():
        if not rows and not fetch_data_xrisi ():
    records = fetch_data_xrisi ()
    for record in records:
        add_row(data=record)

    window_stack. Append (xrisi_submenu)

window = tk. Tk ()
window. Title ("VIORAL")
background_color = "white"
window. Configure (bg=background_color)
window. protocol ("WM_DELETE_WINDOW", on_close)

new_times_font = ("Times New Roman", 16)
buttons_text = ["Αποθήκη", "Εκπαιδεύσεις", "Εξαρτήματα", "Εργαζόμενοι", "Μηχανές", "Παραγγελίες", "Παραγωγή", "Πελάτες",
            "Προμηθευτές", "Συμμετοχές", "Συντήρηση", "Χρήση"]

main_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\main.png'
image = tk. PhotoImage (file=main_path)
main_label = tk. Label (window, image=image)
main_label. pack (expand=True)

tim_lekai_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\tim_lekai.png'
tim_lekai_image = tk. PhotoImage (file=tim_lekai_path)
left_frame = tk. Frame (window, bg=background_color)
left_frame. Place (anchor="nw")

welcome_label = tk. Label (window, font = ("Times New Roman", 16), padx=20, pady=10, bg=background_color)
welcome_label. place (relx=1, rely=0, anchor='ne')

icon_path = 'C:\\Users\\Lenovo\\Desktop\\erp\\images\\logout.png'
icon_image = tk. PhotoImage (file=icon_path)

logout_button = tk. Button (window, image=icon_image, bd=0, command=disconnect_user, bg=background_color)
logout_button. image = icon_image
logout_button. Pack (anchor='w', padx=10, pady=10)

for row, button_text in enumerate(buttons_text):
    button = tk. Button (left_frame, text=button_text, font = ("Times New Roman", 16), bg=background_color, fg="black", bd=0)
    button. Grid (row=row, column=0, padx=10, pady=7, sticky='w')

    button. Bind ("<Button-1>", button_click)
    button. Bind ("<Enter>", on_enter)
    button. Bind ("<Leave>", on_leave)

window. withdraw ()
password_window ()
window. mainloop ()
```