ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ

ΤΜΗΜΑ ΔΙΟΙΚΗΣΗΣ ΕΠΙΧΕΙΡΗΣΕΩΝ ΑΓΙΟΥ ΝΙΚΟΛΑΟΥ

Ανάλυση - Σχεδίαση – Προγραμματισμός Συστήματος Διαχείρισης Πελατειακών Σχέσεων, Μετρήσεων και Χρεώσεων Μιας Εταιρείας Ενέργειας

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Εισηγητής: Γεώργιος - Ελευθέριος, Μουστακίδης, ΑΜ 233

Επιβλέπων: Μιχαλοδημητράκης, Νικόλαος, Εργαστηριακός Συνεργάτης

2018

TECHNOLOGICAL EDUCATION INSTITUTE OF CRETE

SCHOOL OF MANAGEMENT AND ECONOMICS

DEPARTMENT OF BUSINESS ADMINISTRATION (AGIOS NIKOLAOS)

Analysis - Design - Programming of Customer Relationship Management, Measurement and Charging System Of an Energy Company

---

DIPLOMA THESIS

Student : Georgios – Eleftherios, Moustakidis, Student Number 233

Supervisor: Nikolaos, Michalodimitrakis, Laboratory Demonstrator

2018

Υπεύθυνη Δήλωση *: Βεβαιώνω ότι είμαι συγγραφέας αυτής της πτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην πτυχιακή εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του Τμήματος Εμπορίας και Διαφήμισης του Τ.Ε.Ι. Κρήτης.*

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τους γονείς μου αλλά και άλλα μέλη της οικογένεια μου όπου με στηρίξαν στις σπουδές μου και κατ' επέκταση και εμένα τον ίδιο προσφέροντας μου το προνόμιο να μορφωθώ και να εξελιχθώ σαν άνθρωπος.

Τέλος θα ήθελα να ευχαριστώ τον καθηγητή μου Νικόλαο Μιχαλοδημητράκη για την απίστευτη στήριξη και εμπιστοσύνη που μου έδειξε. Είμαι πολύ τυχερός που συνεργάστηκα μαζί του. Με την καθοδήγηση του αλλά και τον ιδιαίτερο τρόπο που προσέγγισε ο ίδιος την συνεργασία μας κατάφερα να εξελιχθώ σε μεγάλο βαθμό. Πλέον αγνοώ οποιοδήποτε εμπόδιο βρω μπροστά μου. Λύσεις υπάρχουν για κάθε πρόβλημα το μόνο που χρειάζεται είναι καθαρή και δομημένη σκέψη.

Η επιμονή του καθηγητή για συνεχή βελτίωση και εξέλιξη άνοιξε νέους δρόμους σε αυτήν την πτυχιακή ξεπερνώντας κατά πολύ το επιθυμητό αποτέλεσμα.

ΠΕΡΙΛΗΨΗ

Το σκοπός της συγκεκριμένης πτυχιακής εργασίας είναι η δημιουργία ενός συστήματος για τη διαχείριση των πελατειακών σχέσεων της. Το σύστημα αυτό ονομάζεται Clover Software. Το σύστημα θα διαχειρίζεται πληροφορίες οι οποίες εμπλέκονται με την εταιρία. Ο χρήστης όπου θα διαχειρίζεται το συγκεκριμένο σύστημα θα μπορεί να διαχειριστή και να τροποποίηση οποιαδήποτε πληροφορία αφορά την εξυπηρέτηση της εταιρίας για μια ομαλή λειτουργία. Η ενέργειες που θα έχει ο κάθε χρήστης ορίζονται από τι θέση του στην εταιρία όπου εργάζεται. Οι δυνατότητες είναι πολλές αλλά η πολυπλοκότητα έχει περιοριστεί αρκετά προβάλλοντας στο χρήστη ένα αρκετά εύκολο και λειτουργικό σύστημα. Στο σύστημα συμπεριλαμβάνονται πολλές παροχές ·σχόλια σε κάθε κουμπί που υπάρχει μέσα στο σύστημα, ενημερωτικά message για πιθανά λάθη, πληροφορίες για τους κανόνες που περικλείουν το σύστημα αλλά και site με βιντεομαθήματα που κάνουν το χρήστη να εξοικειώνεται όλο και περισσότερο με το Clover Software.

Για την ανάπτυξη του συστήματος χρειάστηκε η μελέτη για την κατανόηση του κώδικα σε γλώσσα Python αλλά και του QT5 που απαιτήθηκε για την ανάπτυξη σε γραφικό περιβάλλον. Ενώ για την δημιουργία της βάσης η οποία θα περιέχει όλες εκείνες της πληροφορίες έγινε με την χρήση της SQLite. Η συνεργασία αυτών των δυο προγραμμάτων είχε ως αποτέλεσμα την δημιουργία αυτού του συστήματος.

Λέξεις Κλειδιά : CRM, Clover Software, Python, PyQt5, SQLite3

ABSTRACT

The focus of this diploma thesis is the creation of a client relations managing system. The system is named Clover Software. The system will manage specific company information. The user that manages the system is able to handle and edit any information that concerns the company's services to ensure smooth functionality. The actions of every user on the system are limited. The limitations depend on the position that the user has in the company. The system offers plenty of capabilities through a simple and user friendly graphical interface. Some of the futures of the graphical interface are: popup explanation for every button, informative error massages, information about for the rules of the system and a link to a web site that includes video tutorials so that the user can get comfortable using the clover software.

For the development of this system was required the extensive research and understanding of the Python programming language as well as PyQt5 that was needed for the development of the graphical user interface. In addition SQLite was used to create the database that includes all the information. This system was a result of the collaboration of the two programs above.

Key Words : CRM, Clover Software, Python, PyQt5, SQLite3

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

Περιεχόμενα

Κεφάλαιο 1 – Εισαγωγή και βασικές έννοιες

1.1 Αντικειμενοστρεφής προγραμματισμός

Τα σύγχρονα και ανταγωνιστικά περιβάλλοντα εργασίας οδήγησαν στην ανάπτυξη πολύπλοκων προϊόντων λογισμικού. Η κατασκευή τους και η συντήρησή τους απαιτούσε ειδικούς με γνώσεις σε τεχνικές διαχείρισης μνήμης, λειτουργικά συστήματα, πρότυπα επικοινωνίας αλλά και ειδικές γλώσσες προγραμματισμού. Η πολυπλοκότητα κατασκευής του λογισμικού δημιούργησε ταυτόχρονα προβληματισμούς που οδήγησαν στην αναζήτηση φιλικότερων τεχνικών και μεθόδων σχεδιασμού προγραμμάτων. Από την άλλη πλευρά η μοντελοποίηση του ανθρώπινου συλλογισμού μέσω της τεχνητής νοημοσύνης, έκανε φανερή την ανάγκη της δημιουργίας ενός διαφορετικού προγραμματιστικού περιβάλλοντος που θα ομαδοποιούσε στην ίδια οντότητα όλες τις πληροφορίες και ιδιότητες που αφορούσαν στην ίδια έννοια. Ο αντικειμενοστραφής σχεδιασμός προγραμμάτων προέκυψε από την προσπάθεια αντιμετώπισης αυτών των ζητημάτων. Χρησιμοποιώντας τον όρο αντικειμενοστραφής προγραμματισμός δεν αναφερόμαστε σε κάποιο συγκεκριμένο προϊόν ή μια γλώσσα προγραμματισμού, αλλά σε ένα διαφορετικό τρόπο προσέγγισης του ίδιου του υπολογιστικού προβλήματος. Σύμφωνα με την αντικειμενοστραφή θεωρεία ανάπτυξης εφαρμογών, ο τρόπος που θα αντιμετωπίσεις κάθε προβλήματος πρέπει να γίνεται με φυσική ερμηνεία και να μην στηρίζεται σε πολύπλοκα τεχνικά ζητήματα.

Η αντικειμενοστραφής σχεδίαση λαμβάνει σαν πρωτεύοντα δομικά στοιχεία ενός προγράμματος τα δεδομένα, από τα οποία δημιουργούνται με κατάλληλη μορφοποίηση τα αντικείμενα. Αυτή η σχεδίαση αποδείχθηκε ότι επιφέρει καλύτερα αποτελέσματα, αφού τα προγράμματα που δημιουργούνται είναι περισσότερο ευέλικτα, επαναχρησιμοποιήσιμα και περισσότερο φιλικά. Σύγχρονα αντικειμενοστραφή προγραμματιστικά περιβάλλοντα είναι η Visual C++, η Java, η Visual Basic, και η Delphi (Βακαλή κ.ά., 2010, σ.232-233).

Κατ' ουσίαν, ο αντικειμενοστραφής προγραμματισμός (Object-Oriented Programming ή OOP για συντομία) είναι ένας τρόπος οργάνωσης των προγραμμάτων που γράφουμε. Ο τρόπος αυτός οργάνωσης, δεν είναι φυσικά ούτε μοναδικός, ούτε καν ο βέλτιστος. Άλλοι τρόποι οργάνωσης είναι ο διαδικαστικός (procedural ή imperative) και ο συναρτησιακός (functional) προγραμματισμός. Συνηθίζεται, αυτοί οι τρόποι οργάνωσης ή τεχνικές

οργάνωσης των προγραμμάτων να ονομάζονται προγραμματιστικά παραδείγματα (programming paradigms) (Python Tutorial, 2012).

1.1.1 Ιστορικό

Αντικειμενοστρεφή προγραμματισμό (object-oriented programming), ή ΑΠ, ονομάζουμε ένα προγραμματιστικό παράδειγμα το οποίο εμφανίστηκε στα τέλη της δεκαετίας του 1960 και καθιερώθηκε κατά τη δεκαετία του 1990, αντικαθιστώντας σε μεγάλο βαθμό το παραδοσιακό υπόδειγμα του δομημένου προγραμματισμού (Wikipedia, 2016).

Οι περισσότερες αντικειμενοστρεφείς έννοιες εμφανίστηκαν αρχικά στη γλώσσα προγραμματισμού Simula 67, η οποία ήταν προσανατολισμένη στην εκτέλεση προσομοιώσεων του πραγματικού κόσμου. Οι ιδέες της Simula 67 επηρέασαν κατά τη δεκαετία του '70 την ανάπτυξη της Smalltalk, της γλώσσας που εισήγαγε τον όρο αντικειμενοστρεφής προγραμματισμός. Την ίδια περίπου εποχή και επίσης με επιρροές από τη Simula, ολοκληρωνόταν η ανάπτυξη της C++ ως μίας ισχυρής επέκτασης της δημοφιλούς γλώσσας προγραμματισμού C στην οποία είχαν μεταφερθεί αντικειμενοστρεφή χαρακτηριστικά. Η επιρροή της C++ καθ' όλη της δεκαετία του '80 ήταν καταλυτική με αποτέλεσμα τη σταδιακή κυκλοφορία αντικειμενοστρεφών εκδόσεων πολλών γνωστών διαδικαστικών γλωσσών προγραμματισμού. Κατά το πρώτο ήμισυ της δεκαετίας του '90 η βαθμιαία καθιέρωση στους μικροϋπολογιστές των γραφικών διασυνδέσεων χρήστη (GUI), για την ανάπτυξη των οποίων ο ΑΠ φαινόταν ιδιαιτέρως κατάλληλος, και η επίδραση της C++ οδήγησαν στην επικράτηση της αντικειμενοστρέφειας ως βασικού προγραμματιστικού υποδείγματος (Wikipedia, 2016).

1.2 Προγραμματιστικό Παράδειγμα

Προγραμματιστικό παράδειγμα ή προγραμματιστικό υπόδειγμα καλείται ένα θεμελιώδες στυλ προγραμματισμού υπολογιστών, με το οποίο ένα υπολογιστικό πρόβλημα και η αλγοριθμική λύση του προσεγγίζονται με συγκεκριμένες μεθόδους. Είναι δηλαδή ένα σύνολο εννοιών οι οποίες εκφράζουν έναν συγκεκριμένο τρόπο σκέψης. Αυτό έχει ως συνέπεια τη διαμόρφωση ενός προγράμματος όπου οι δημιουργοί λογισμικού αποφασίζουν πώς θα χρησιμοποιήσουν τις μεθόδους (Wikipedia, 2017).

Η επιλογή του προγραμματιστικού παραδείγματος το οποίο θα χρησιμοποιήσουμε εξαρτάται από το πρόβλημα το οποίο καλούμαστε να επιλύσουμε, αλλά και από τη γλώσσα προγραμματισμού την οποία χρησιμοποιούμε. Δεν επιτρέπουν όλες οι γλώσσες προγραμματισμού τη χρήση όλων των προγραμματιστικών παραδειγμάτων. Για παράδειγμα η γλώσσα Fortran επιτρέπουν τη δημιουργία μόνο διαδικαστικών προγραμμάτων. Υπάρχουν και άλλες που επιτρέπουν κυρίως τη δημιουργία μόνο συναρτησιακών προγραμμάτων, όπως η Lisp. Ενώ υπάρχουν και γλώσσες που επιτρέπουν την εφαρμογή περισσότερων από ένα προγραμματιστικά παραδείγματα. Η Python ανήκει σε αυτήν την κατηγορία καθώς σου επιτρέπει να γράψεις κώδικα που χρησιμοποιεί και τα τρία προγραμματιστικά παραδείγματα (Python Tutorial, 2012).

1.3 Βασικές Έννοιες Αντικειμενοστρέφιας

1.3.1 Κλάσεις (Classes)

Η κεντρική ιδέα στον αντικειμενοστρεφή προγραμματισμό είναι η κλάση (class). Η κλάση είναι ένας τύπος δεδομένων, ή αλλιώς η βάση μίας δομής δεδομένων με τα δικά της περιεχόμενα, μεταβλητές αλλά και διαδικασίες. Το περιεχόμενο μιας κλάσης μπορεί να δηλωθεί σαν δημόσια (public) είτε ως ιδιωτικά (private) (σε περίπτωση δήλωσης ενός περιεχομένου ως ιδιωτικό δεν θα είναι προσπελάσιμο από άλλες κλάσεις). Οι διαδικασίες των κλάσεων συνήθως καλούνται ως μέθοδοι (methods) και οι μεταβλητές τους ως *γνωρίσματα* (attributes) ή *πεδία* (fields). Μια κλάση δεν είναι τίποτα άλλο παρά ένας ορισμός. Αυτό που ορίζει είναι το ποιες ιδιότητες (attributes) και ποιες μεθόδους (methods) θα έχει ένα αντικείμενο. ιδανικά μία κλάση πρέπει να περιέχει μεθόδους οι οποίες καλούνται από το εξωτερικό πρόγραμμα, χωρίς να εξαρτώνται από άλλα δεδομένα ή κώδικα εκτός κλάσης. Σκοπός είναι επαναχρησιμοποίηση του κώδικα (Wikipedia, 2017).

1.3.2 Αντικείμενα (object)

Ένα αντικείμενο (object) είναι το στιγμιότυπο μίας κλάσης. Δηλαδή αυτή καθαυτή η δομή δεδομένων βασισμένη σε μια υπόσταση που προσφέρει η κλάση. Κάθε αντικείμενο είναι ευδιάκριτο και αυτάρκες, ενώ το σύνολο των χαρακτηριστικών ιδιοτήτων του προσδιορίζεται με λεπτομέρεια στη φυσική του υπόσταση. Είναι λοιπόν εμφανές ότι η περιγραφή ενός αντικειμένου καθορίζεται από τις τιμές των επιμέρους ιδιοτήτων του (Wikipedia, 2017).

1.3.3 Κληρονομικότητα (Inheritance)

Ίσως η πλέον κεφαλαιώδης έννοια του αντικειμενικοστραφούς προγραμματισμού είναι η κληρονομικότητα (inheritance). Κληρονομικότητα ονομάζεται η ιδιότητα των κλάσεων να επεκτείνονται σε νέες κλάσεις**.** Μια κλάση μπορεί να περιγράφεται γενικά και στη συνέχεια μέσω αυτής της κλάσης να οριστούν υποκλάσεις (subclasses) αντικειμένων. Η υποκλάση κληρονομεί και μπορεί να χρησιμοποιήσει όλα τα δεδομένα και τις μεθόδους που περιέχει η κλάση πρόγονος. Δηλαδή να μπορούν να επαναχρησιμοποιήσουν τις μεταβιβάσιμες μεθόδους και ιδιότητες της γονικής τους κλάσης αλλά και να προσθέσουν δικές τους (Wikipedia, 2017).

1.3.4 Πολλαπλή κληρονομικότητα

Πολλαπλή κληρονομικότητα είναι η δυνατότητα που προσφέρουν ορισμένες γλώσσες προγραμματισμού μία κλάση να κληρονομεί ταυτόχρονα περισσότερες από μία υπερκλάση. Από μία υποκλάση μπορούν να προκύψουν νέες υποκλάσεις που κληρονομούν από αυτήν, με αποτέλεσμα μία ιεραρχία κλάσεων που συνδέονται μεταξύ τους με σχέσεις κληρονομικότητας (Wikipedia, 2017).

Ένα παράδειγμα πολλαπλής κληρονομικότητας είναι το ακόλουθο:

File Edit Format Run SubCode Options Window Help

> ## - 1.0 + / 1.1 * RS RSP RA

```
1 class BaseClass1():
2     pass
3
4 class BaseClass2():
5     pass
6
7 class DerivedClass(BaseClass1, BaseClass2):
8     pass
9
```

1.3.5 Απλή και Πολλαπλή Κληρονομικότητα

Η διαφορά μεταξύ της απλής και της πολλαπλής κληρονομικότητας έγκειται στον αριθμό των υπερκλάσεων που έχει μία συγκεκριμένη υποκλάση. Αν κληρονομεί από μία μόνο υπερκλάση τότε μιλάμε για απλή κληρονομικότητα. Αν κληρονομεί από δύο ή περισσότερες υπερκλάσεις, τότε μιλάμε για πολλαπλή κληρονομικότητα (Python Tutorial, 2012).

1.3.6 Αφαιρετικότητα

Γενικά μπορούμε να πούμε πως ισχύει η παρακάτω εξίσωση:

Αφαιρετικότητα Δεδομένων = Ενθυλάκωση + Απόκρυψη

Η Ενθυλάκωση είναι η ομαδοποίηση των δεδομένων μαζί με τις μεθόδους που τις χρησιμοποιούν. Στη διαδικασία της ενθυλάκωσης η άμεση πρόσβαση στα δεδομένα είναι δυνατή αλλά είναι προτιμότερο να υπάρχει ελεγχόμενη πρόσβαση σε αυτά. Η Ενθυλάκωση συνήθως επιτυγχάνεται μέσω δύο ειδών μεθόδων για τα ιδιοχαρακτηριστικά:

Οι μέθοδοι που διαβάζουν και επιστρέφουν τις τιμές των πεδίων(μεταβλητών) ονομάζονται Μέθοδοι Ανάγνωσης (Getter methods), όπου δεν μπορεί να αλλάξει τι τιμή μιας μεταβλητής.

Οι μέθοδοι που μπορούν να αλλάξουν τις τιμές των μεταβλητών ονομάζονται Μέθοδοι Ορισμού (Setter methods)

```
class Robot:
    def __init__(self,name):
        self.name  =name
    def SetName(self,name):
        self.name = name
    def GetName(self):
        return self.name
x = Robot("Giorgos")
y = Robot("Nikos")
for rob in [x , y]:
    print("Hi, I'm ",rob.GetName(),"!")
```

Η Απόκρυψη είναι μια διαδικασία σύμφωνα με την οποία κάποια «εσωτερικά» δεδομένα και πληροφορίες είναι «κρυμμένα» στον κώδικα ώστε να αποτρέπει να αλλάξει την τιμή τους καταλάθος (χωρίς να το επιθυμούμε ευθέως).

Ιδιωτικά χαρακτηριστικά (private attributes): Χρησιμοποιούνται μόνο από τον ιδιοκτήτη τους(εντός της ίδιας της κλάσης όπου έχει οριστεί). Το χαρακτηριστικό (μεταβλητή) γίνεται ιδιοτικό με δύο χαρακτήρες της κάτω παύλας “__”.

Προστατευμένα χαρακτηριστικά(Protected attributes): Χρησιμοποιούνται κυρίως μέσω της κληρονομικότητας. Το χαρακτηριστικό (μεταβλητή) γίνεται προστατευμένο με ένα χαρακτήρα της κάτω παύλας “_”.

Δημόσια χαρακτηριστικά (public attributes): Χρησιμοποιούνται ελεύθερα.

Παρακάτω βλέπουμε μια κλάση όπου χρησιμοποιεί ένα ιδιωτικό χαρακτηριστικό:

```
class Clover:
    __sys = 0
    def run(self):
        self.__sys+= 5
        print(self.__sys)
counter=Clover()
counter.run()

print(counter.__sys)
5
Traceback (most recent call last):
  File "C:\Users\moust\Desktop\sdf.py",
line 9, in <module>
    print(counter.__sys)
AttributeError: 'Clover' object has no
attribute '__sys'
>>>
```

1.4 Τι είναι οι βάσεις δεδομένων;

Είναι ένα εργαλείο για τη συλλογή δεδομένων όπου οργανώνει και ταξινομεί με τέτοιο τρόπο ώστε να διευκολύνεται η διαδικασία εκτέλεσης λογικών πράξεων πάνω σε αυτή. Οι βάσεις δεδομένων μπορούν να αποθηκεύουν δεδομένα σχετικά με άτομα, προϊόντα, παραγγελίες ή οτιδήποτε άλλο. Δεδομένα που δε σχετίζονται μεταξύ τους και απλά έχουν αποθηκευτεί σ' έναν Η/Υ δεν αποτελούν μια Βάση Δεδομένων. Μια Βάση Δεδομένων πρέπει να αντικατοπτρίζει ένα περιβάλλον του πραγματικού κόσμου. Τα δεδομένα που αποθηκεύονται στη Βάση Δεδομένων πρέπει να έχουν λογική συνέχεια και νόημα.

1.4.1 Εισαγωγικές Έννοιες

Μια ηλεκτρονική βάση δεδομένων είναι ένα κοντέινερ αντικειμένων. Μία βάση δεδομένων μπορεί να περιέχει περισσότερους από έναν πίνακες. Για παράδειγμα, ένα σύστημα παρακολούθησης αποθήκης που χρησιμοποιεί τρεις πίνακες παρακολούθησης δεν είναι τρεις βάσεις δεδομένων, αλλά μία βάση δεδομένων που περιέχει τρεις πίνακες. Η Βάση Δεδομένων περιέχει τα δεδομένα που περιγράφουν κάθε φορά το «πρόβλημα», ενώ μπορεί να περιέχει και δεδομένα από την ιστορία του προβλήματος (Microsoft, 2018).

Μια βάση δεδομένων, λοιπόν, έχει κάποια πηγή από την οποία παράγονται τα δεδομένα, αλληλεπιδρά σε κάποιο βαθμό με γεγονότα του πραγματικού κόσμου και απευθύνεται σε ένα ακροατήριο που ενδιαφέρεται ενεργά για τα περιεχόμενό της. Το ακροατήριο αποτελείτε από

ανθρώπους οι οποίοι έχουν ως καθήκον τους να εισάγουν δεδομένα στην Βάση Δεδομένων. (Γεωργίου, 2011). Οι άνθρωποι αυτοί χρησιμοποιούν τη Βάση Δεδομένων και καλούνται χρήστες (users). Χρήστες μιας Βάσης Δεδομένων είναι όσοι γενικότερα φροντίζουν για την εύρυθμη λειτουργία Βάση Δεδομένων είτε δηλαδή για απόκτηση πληροφορίας είτε για τη συντήρηση της Βάσης Δεδομένων.

1.4.2 Ιστορική αναδρομή

Οι Βάσεις Δεδομένων θεωρητικά δεν προϋποθέτουν την ύπαρξη ηλεκτρονικού υπολογιστή. Οι πρώτες Βάσεις Δεδομένων δεν βασίζονταν στη χρήση ηλεκτρονικού υπολογιστή, αλλά σε χαρτί και οργανώνονταν με φακέλους ή καρτέλες.

Οι Βάσεις Δεδομένων εξελίχτηκαν πολύ στα χρόνια που πέρασαν. Από μικρές Βάσεις Δεδομένων που υπήρχαν οι οποίες αποτελούνταν από μικρούς αριθμούς απλών δεδομένων, σήμερα έχουν αναπτυχθεί Βάσεις Δεδομένων που χειρίζονται τεράστιο όγκο πολύπλοκων δεδομένων. Σήμερα μιλάμε για Βάσεις Δεδομένων της τάξης των Terabytes με δεδομένα τα οποία περιέχουν εικόνες, ή βίντεο και Συστήματα διαχείρισης Βάσεων Δεδομένων με δυνατότητες ενσωμάτωσης λειτουργιών για καλύτερο χειρισμό των δεδομένων, καθώς και για τρόπους εναλλακτικής οργάνωσης των δεδομένων, όπως οι αντικειμενοστραφείς Βάσεις Δεδομένων (Ξένος, Χριστοδουλάκης, 2000, σ.17-19).

1.4.3 Εφαρμογές Συστημάτων Βάσεων Δεδομένων

Οι Βάσεις Δεδομένων πλέων είναι ευρέος γνωστές στον επιχειρηματικό χώρο. Την χρήση των βάσεων θα την παρατηρήσουμε σε εταιρίες όπως:

- Τηλεπικοινωνίες: για διατήρηση των κλήσεων, δημιουργία μηνιαίων λογαριασμών κ.α.
- Πωλήσεις: για πληροφορίες πελατών, προϊόντων και πωλήσεων.
- Ανθρώπινοι πόροι: για πληροφορίες για εργαζόμενους, μισθούς, φόρους μισθοδοσίας και παροχές και για πληρωμές μισθών.
- Πανεπιστήμια: για πληροφορίες φοιτητών, εγγραφές σε μαθήματα και βαθμούς.

Η εξέλιξη των Βάσεων Δεδομένων και οι ανάγκες για δημιουργία όλο και περισσότερων Βάσεων Δεδομένων, οδήγησαν στη δημιουργία των Συστημάτων Διαχείρισης Βάσεων Δεδομένων (Database Management Systems ή DBMS). Το Σύστημα

Διαχείρισης Βάσεων Δεδομένων «φιλοξενεί» πολλές Βάσεις Δεδομένων. Πλέον το χρησιμοποιούν οι περσότερες επιχειρήσεις καθώς διευκολύνει την χρήση των δεδομένων μέσα στην βάση.

Σε αυτό το σημείο πρέπει να είναι ξεκάθαρη η διαφορά μεταξύ ενός Συστήματος Διαχείρισης Βάσεων Δεδομένων (ΣΔΒΔ) και μίας Βάσης Δεδομένων. Το Σύστημα Διαχείρισης Βάσεων Δεδομένων είναι λογισμικό το οποίο διευκολύνει τους χρήστες να χρησιμοποιούν Βάσεις Δεδομένων. Για παράδειγμα η mySQL η ή SQLite που χρησιμοποιούμε εμείς στο πρόγραμμα αποτελούν λογισμικά Διαχείρισης Βάσεων Δεδομένων. Αντίθετα η Βάση Δεδομένων υλοποιείται με τη βοήθεια ενός Συστήματος Διαχείρισης Βάσεων Δεδομένων. Η όλη διαδικασία χειρισμού και αποθήκευσης των δεδομένων διευκολύνεται από το Σύστημα Διαχείρισης Βάσεων Δεδομένων, το οποίο αναλαμβάνει τη μετατροπή των εντολών του χρήστη σε εντολές προς τον υπολογιστή με προκαθορισμένες από το σύστημα εντολές οι οποίες σχετίζονται με τον χειρισμό δεδομένων (Ξένος, Χριστοδουλάκης, 2000, σ.20-21).

Πλεονεκτήματα Σ.Δ.Β.Δ.

1. Ευκολία στη σχεδίαση και υλοποίηση
2. Δυνατότητα για τήρηση αντιγράφων ασφαλείας καθώς και ανάκαμψης της Βάσης Δεδομένων μετά από βλάβες.
3. Δυνατότητα περιγραφής περιορισμών ορθότητας για την οργάνωση των δεδομένων στη Βάση ή τις τιμές που μπορούν να λάβουν κάποια δεδομένα.
4. Δυνατότητα ταυτόχρονης προσπέλασης πολλών χρηστών στα δεδομένα και έλεγχο της προσπέλασης.
5. Δυνατότητα ταχύτατης εξαγωγής απαντήσεων σε απλές ερωτήσεις.
6. Ευελιξία σε τυχόν αλλαγές και γενικότερα ευκολία παρακολούθησης των αλλαγών του μοντέλου του πραγματικού κόσμου.
7. Υψηλή ποιότητα δεδομένων (Ξένος, Χριστοδουλάκης, 2000, σ.25-27).

Μειονεκτήματα διατήρησης πληροφοριών σε σύστημα επεξεργασίας αρχείων

Η διατήρηση των πληροφοριών μιας εταιρείας σ’ ένα σύστημα επεξεργασίας αρχείων έχει διάφορα μεγάλα μειονεκτήματα:

1. Επαναληπτικότητα και ασυνέπεια των δεδομένων.
2. Δυσκολία στην πρόσβαση των δεδομένων.
3. Απομόνωση των δεδομένων.

4. Προβλήματα ακεραιότητας.
5. Προβλήματα ατομικότητας
6. Προβλήματα ταυτόχρονης πρόσβασης
7. Προβλήματα ασφάλειας.

1.4.4 Τι είναι η γλώσσα SQL;

Η SQL (Structured Query Language) είναι μία γλώσσα υπολογιστών που σχεδιάστηκε για την αποστολή ερωτημάτων σε μια βάση δεδομένων (Wikipedia, 2017).

Μέσα από την χρήση της γλώσσας SQL μπορούν να οριστούν λειτουργίες όπως:

- Ορισμός του τύπου των δεδομένων που θα αποθηκεύει η βάση.
- Εισαγωγή, επεξεργασία και διαγραφή των αποθηκευμένων δεδομένων.
- Ανάκτηση των δεδομένων από την βάση με χρήση ερωτημάτων για χρήση από άλλες εφαρμογές.

Τα ερωτήματα της γλώσσας SQL περιλαμβάνουν διάφορους τύπους εντολών.
Ο Πίνακας 1.1 παρουσιάζει τις πιο συνηθισμένες δηλώσεις SQL (Ip.gr).

Δήλωση	Σκοπός
SELECT	Επιλέγει γραμμές από έναν πίνακα
INSERT	Εισάγει γραμμές σ' έναν πίνακα
DELETE	Διαγράφει γραμμές από έναν πίνακα
UPDATE	Ενημερώνει γραμμές ενός πίνακα
COMMIT	Εκτελεί μία συναλλαγή
ROLLBACK	Αναστρέφει μία συναλλαγή
RANT	Παραχωρεί δικαιώματα ασφαλείας
REVOKE	Ανακαλεί δικαιώματα ασφαλείας

Πίνακας 1.1

1.4.5 Η ιστορία της SQL

Η SQL βασίστηκε στη σχεσιακή άλγεβρα. Προτάθηκε στις αρχές του 1970 από τον ερευνητή Tedd Codd των εργαστηρίων IBM San Josi Research Laboratories. Δημιουργήθηκε με σκοπό να παρέχει μια ημι – φυσική γλώσσα για το IBM System Relational database system. Αρχικά οι σχεσιακές βάσεις δεδομένων είχαν πολλούς και πιο αποδοτικούς ανταγωνιστές π.χ. συστήματα διαχείρισης δεδομένων τα οποία βασίζονταν σε δικτυακά μοντέλα δεδομένων.

Ωστόσο, όλο και πιο αποδοτικά σχεσιακά συστήματα άρχισαν να εμφανίζονται στα τέλη του 1980 κι αυτό οδήγησε στην επικράτηση των σχεσιακών βάσεων δεδομένων και της SQL. Δυστυχώς, υπάρχουν πολλές διάλεκτοι της SQL που διαφέρουν ελαφρώς από το ένα σύστημα στο άλλο. Οι κυριότερες είναι οι τυποποιημένες ANSI/ISO SQL. Πρόκειται για την σταθερή εκδοχή της γλώσσας όπως αυτή έχει οριστεί από δύο αναγνωρισμένους οργανισμούς. Υπήρξαν δυο πρότυπα: το SQL-89 ή SQL1 και το SQL-92 ή SQL2. Οι περισσότερες εφαρμογές προσπαθούν να μείνουν πιστές στην SQL-92.

DML (Data Manipulation Language) DDL (Data Definition Language) DCL (Data Control Language)

Επιπλέον υπάρχουν πολλές παραλλαγές της γλώσσας SQL που κυκλοφορούν στην αγορά, όπου χαρακτηρίζονται από την ίδια δομή και την ίδια φιλοσοφία. Έτσι, μια τυπική γλώσσα SQL, θα περιλαμβάνει τις επόμενες δομικές μονάδες:

- **Γλώσσα Ορισμού Δεδομένων** (Data Definition Language, DDL): Η γλώσσα αυτή περιλαμβάνει εντολές που μας επιτρέπουν να υλοποιήσουμε πίνακες, σχέσεις ανάμεσα σε πίνακες, και γενικά όλη τη δομή μιας βάσης δεδομένων.
- **Γλώσσα χειρισμού δεδομένων** (Data Manipulation Language, DML): Η γλώσσα αυτή επιτρέπει τη διαχείριση των δεδομένων της εφαρμογής, όπως την εισαγωγή, διαγραφή, ανάκτηση και τροποποίηση δεδομένων.
- **Ορισμός όψεων της βάσης** (View Definition): Επιτρέπει τη δημιουργία όψεων της βάσης δεδομένων, ορίζονται ως εικονικοί πίνακες (virtual tables) οι οποίοι περιέχουν δεδομένα από έναν ή περισσότερους πίνακες της βάσης.
- **Ορισμός εξουσιοδοτήσεων** (Authorization): Επιτρέπει τη δημιουργία ομάδων χρηστών, και την απόδοση διαφορετικών δικαιωμάτων πρόσβασης σε κάθε έναν από αυτούς, προκειμένου η κάθε ομάδα χρηστών, να διαχειρίζεται μόνο τα δικά της δεδομένα.
- Δ**ιαχείρισης ακεραιότητας** (Integrity): Επιτρέπει το λεπτομερή έλεγχο των δεδομένων που καταχωρούνται στη βάση, έτσι ώστε να μην παραβιάζονται οι κανόνες ακεραιότητας (integrity constraints) που έχουμε ορίσει και οι οποίοι όταν τηρούνται, απομακρύνουν τον κίνδυνο καταχώρησης ασυνεπών δεδομένων (inconsistent data) (Μπαγκέρη).

1.5 Τι είναι η βάση δεδομένων SQLite 3

Το SQLite είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων που περιέχεται σε μια C προγραμματιστική βιβλιοθήκη. Σε αντίθεση με άλλα συστήματα διαχείρισης βάσης δεδομένων, το SQLite δεν είναι μια ξεχωριστή διεργασία που προσπελάζεται από μια εφαρμογή πελάτη, αλλά ένα ενσωματωμένο μέρος της. Το SQLite είναι μια δημοφιλής επιλογή ως ενσωματωμένη βάση δεδομένων για τοπική αποθήκευση ή αποθήκευση πελάτη σε λογισμικό εφαρμογής όπως ως φυλλομετρητής (Wikipedia, 2017).

1.5.1 Πληροφορίες για την SQLite 3

Το SQLite είναι συμβατό με ACID (Atomicity, Consistency, Isolation, Durability) και υλοποιεί το μεγαλύτερο μέρος του προτύπου SQL, χρησιμοποιώντας μια δυναμική και εβδομαδιαία τυπωμένη SQL σύνταξη που δεν εγγυάται την ακεραιότητα του τομέα.

Το SQLite έχει συνδέσεις με πολλές γλώσσες προγραμματισμού. Η μηχανή SQLite δεν έχει αυτόνομες διεργασίες με τις οποίες επικοινωνεί με κάποιο Συστήματα Διαχείρισης Βάσεων Δεδομένων. Έτσι η ενσωμάτωση της στο πρόγραμμα γίνεται μέσα από την σύνδεση μια SQLite βιβλιοθήκης. Το πρόγραμμα της εφαρμογής χρησιμοποιεί τη λειτουργικότητα του SQLite μέσα από απλές κλήσεις συνάρτησης, που μειώνουν την καθυστέρηση στην πρόσβαση της βάσης δεδομένων. Το SQLite αποθηκεύει την συνολική βάση δεδομένων (ορισμούς, πίνακες, δείκτες και τα ίδια τα δεδομένα) ως ένα μοναδικό διαλειτουργικό αρχείο στη μηχανή ενός οικοδεσπότη. Υλοποιεί αυτόν τον απλό σχεδιασμό με κλείδωμα όλου του αρχείου της βάσης δεδομένων κατά τη διάρκεια της εγγραφής. Το SQLite διαβάζει λειτουργίες που μπορεί να είναι πολυλειτουργικές, αν και οι εγγραφές μπορούν να γίνουν μόνο με τη σειρά.

Το SQLite υλοποιεί το μεγαλύτερο μέρος του προτύπου SQL-92 για το SQL, αλλά του λείπουν κάποια χαρακτηριστικά. Για παράδειγμα, έχει μερική υποστήριξη για εναύσματα βάσης δεδομένων και δεν μπορεί να γράψει σε προβολή (βάσης δεδομένων) (υποστηρίζει όμως στη θέση τους εναύσματα που παρέχουν αυτή τη λειτουργία). Αν και υποστηρίζει σύνθετα ερωτήματα, έχει ακόμα περιορισμένη υποστήριξη για ALTER TABLE, καθώς δεν μπορεί να τροποποιήσει ή να διαγράψει στήλες (Wikipedia, 2017).

1.6 Το σύστημα γραφικής διεπαφής Qt

1.6.1 Ιστορία της Qt

Το 1990 οι Haavard Nord και Eirik Chambe-Eng συνεργάζονταν σε μια εφαρμογή βάσης δεδομένων πάνω σε C ++ σε λειτουργικό Mac OS, Unix , και Windows. Οι ίδιοι ξεκίνησαν την ανάπτυξη του Qt το 1991.Το Qt πήρε την ονομασία αυτή λόγο που το Q φαινόταν ελκυστικό στη χρήση της γραμματοσειράς του Emacs του Haavand και το t από το Xt (X toolkit). Οι δύο πρώτες εκδόσεις του Qt την Qt / X11 για Unix και Qt / Windows για Windows. Να αναφερθεί ότι η πλατφόρμα σε Windows ήταν διαθέσιμη μόνο κάτω από μια ιδιόκτητη άδεια χρήσης, πράγμα που σήμαινε ότι ελεύθερες / ανοιχτές εφαρμογές που γράφονται στο Qt για X11 (Qt / X11 για Unix) δεν θα μπορούσαν να μεταφερθούν στα Windows χωρίς την αγορά του Qt (Wikipedia, 2018).

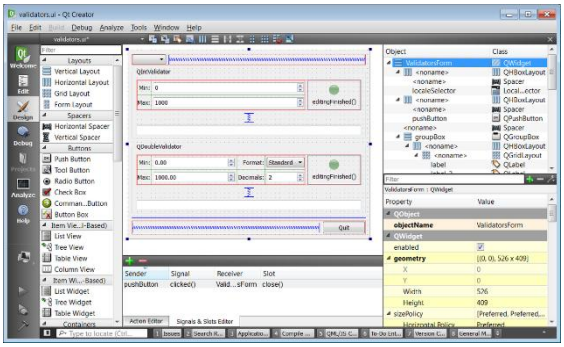
1.6.2 Qt Creator

Η ανάπτυξη σε αυτό που τελικά θα γινόταν το Qt Creator είχε ξεκινήσει από το 2007 ή νωρίτερα κάτω από τα μεταβατικά ονόματα Workbench και αργότερα Project Greenhouse. Ξεκίνησε με την απελευθέρωση του Qt Creator, έκδοση 1.0 τον Μάρτιο του 2009 και στη συνέχεια συνοδευόταν από το Qt 4.5 στο FBX SDK 2009.3 (Wikipedia, 2018).

Ο Qt Creator είναι ένα ενσωματωμένο περιβάλλον ανάπτυξης C ++, JavaScript και QML που είναι μέρος του SDK (Software Development kit) για το πλαίσιο ανάπτυξης εφαρμογών Qt GUI (graphical user interface). Ο επεξεργαστής κώδικα στο Qt Creator υποστηρίζει την επισήμανση σύνταξης για διάφορες γλώσσες. Εκτός από αυτό, ο επεξεργαστής κώδικα μπορεί να αναλύσει τον κώδικα σε γλώσσες C ++ και QML. Περιλαμβάνει έναν επεξεργαστή κώδικα όπου ενσωματώνει τον Qt Designer για το σχεδιασμό και την κατασκευή γραφικών διεπαφών χρήστη (GUI) από τα widgets Qt (Wikipedia, 2018).

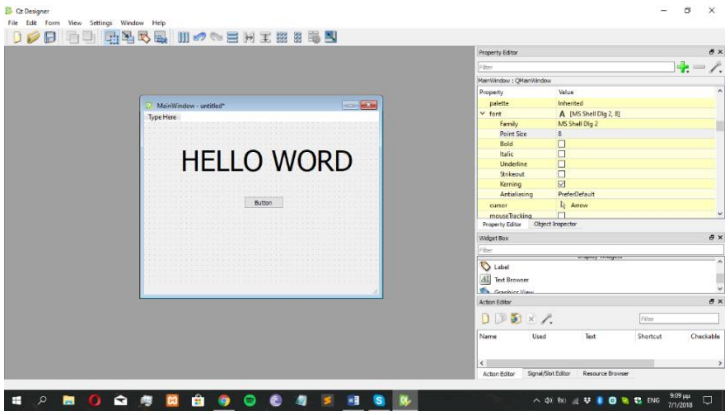
1.6.3 Qt Designer

Ο Qt Designer είναι ένα εργαλείο για το σχεδιασμό και την κατασκευή γραφικού περιβάλλοντος. Μέσα από την επεξεργασία του κώδικα μπορούν να συνδεθούν και να προσαρμοστούν τα widgets που έχουν



δημιουργηθεί. Η επεξεργασία αυτήν μπορεί να απευθύνεται στην ενεργοποίηση κάποιας ενεργείας η τροποποίηση του widget κ.α.

Το Qt Designer είναι ένα εργαλείο όπου σχεδιάζει και κτίζει το γραφικό περιβάλλον. Σου επιτρέπει να σχεδιάσεις widgets, dialogs ή ολόκληρη τη βασική φόρμα καθώς μέσα από το Qt ευκολά σύρεις από τον Editor όποιο widget θες να συμπεριλάβεις στο παράθυρο (φόρμα) που δημιουργείς.



Το Qt Designer χρησιμοποιεί αρχεία XML .ui για την προβολή του γραφικού περιβάλλοντος αλλά και για την εξοικονόμηση ενός μεγάλου μέρους του κώδικα που απαιτείτε για την κατασκευή μιας εφαρμογής. Το Qt συμπεριλαμβάνει επίσης την κλάση QUiLoader που επιτρέπει σε μια εφαρμογή να φορτώσει ένα .ui αρχείο και να δημιουργήσει αντίστοιχα δυναμικό περιβάλλον χρήστη. Το PyQt5 δεν εμπεριέχει την κλάση QUiLoader αλλά συμπεριλαμβάνει το μοντέλο uic Python. Όπως το QuiLoader, μπορεί να φορτώσει ένα αρχεία .ui για να δημιουργήσει ένα δυναμικό περιβάλλον. Έτσι και η χρήση της uic, μπορεί επίσης με κώδικα Python να παράγει ένα περιβάλλον χρήστη (Riverbank Computing, 2017).

1.6.4 Χρήση του παραγόμενου κώδικα

Ο κώδικας που παράγεται έχει την ίδια δομή με εκείνη που παράγεται από το Qt uic και μπορεί να χρησιμοποιηθεί με τον ίδιο τρόπο (Σε περίπτωση που δεν γίνει εξαγωγή του αρχείου). Ο κώδικας είναι δομημένος ως μία κλάση που προέρχεται από τον τύπο αντικειμένων Python. Το όνομα της κλάσης είναι το όνομα του αντικειμένου (MainWindow) που έχει οριστεί στο Designer με Ui_ prepended. Η κλάση εμπεριέχει μια μέθοδο που ονομάζεται setupUi (). Η μέθοδος αυτή βοήθα στον ορισμό της τάξης Qt ανάμεσα στα QDialog, QWidget ή QMainWindow για την προβολή τους στο περιβάλλον χρήστη. Μέσα από το Qt Designer ο χρήστης μπορεί να διαχειριστεί αλλά και να ορίσει τις διευθύνσεις του κάθε αντικειμένου. Στην συνέχεια μπορεί να τα μετατρέψει σε δυναμικά δίνοντας τους ενεργείς είτε με την χρήση της Python είτε από το ίδιο το Qt Designer. Έτσι η

τροποποίησης του περιβάλλοντος εξαρτάτε από τον ίδιο τον χρήστη (Riverbank Computing, 2016).

1.7 Η γλώσσα προγραμματισμού Python

1.7.1 Εισαγωγικά στην Python

Η Python είναι μια εύκολη στην εκμάθηση, δυναμική, αποδοτική, παραγωγική και επεκτάσιμη γλώσσα. Είναι μια ισχυρή γλώσσα προγραμματισμού. Έχει αποδοτικές δομές δεδομένων υψηλού επιπέδου και μια απλή αλλά αποτελεσματική προσέγγιση στον αντικειμενοστρεφή προγραμματισμό. Η κομψή σύνταξη της Python και οι δυναμικοί τύποι της, μαζί με τη λειτουργία της ως διερμηνευόμενης (αντί μεταγλωττιζόμενης) γλώσσας, την καθιστούν την ιδανική γλώσσα για δημιουργία σεναρίων εντολών και για ταχεία ανάπτυξη εφαρμογών σε πολλούς τομείς και στις περισσότερες πλατφόρμες. Είναι μια αρκετά δημοφιλείς γλώσσα προγραμματισμού και μπορεί να χρησιμοποιηθεί τόσο για εκπαιδευτικούς σκοπούς όσο και για την ανάπτυξη ολοκληρωμένων εφαρμογών. Κάθε μέρα γίνεται όλο και περισσότερο δημοφιλής (CyberPython).

Η Python υποστηρίζει και διαθέτει αποδοτικές δομές δεδομένων υψηλού επιπέδου, έτσι έχει μια αρκετά αποτελεσματική προσέγγιση στον αντικειμενοστρεφή προγραμματισμό. Επίσης υποστηρίζει διαδικαστικό αλλά και συναρτησιακό προγραμματισμό. Η σύνταξή της είναι απλή και οι τύποι της δυναμικοί. Είναι διερμηνευόμενη (interpreted) γλώσσα προγραμματισμού και μπορεί να χρησιμοποιηθεί για τη γρήγορη ανάπτυξη ολοκληρωμένων εφαρμογών. Επίσης μπορεί να για διαχείριση συστήματος υπολογιστή, ανάπτυξη εφαρμογών Διαδικτύου, επεξεργασία αρχείων κειμένου, επιστημονικές εφαρμογές, εκπαίδευση, ανάπτυξη παιχνιδιών, κ.λπ. Η Python υποστηρίζεται από τα περισσότερα Λειτουργικά Συστήματα (Windows, Unix, Linux, MacOS X, κ.λπ.). Διαθέτει πληθώρα έτοιμων βιβλιοθηκών. Οι βιβλιοθήκες μπορούν να επεκταθούν εύκολα. Τα προγράμματα σε Python είναι συμπαγή, ευανάγνωστα. Γράφονται και συντηρούνται γρηγορότερα σε σχέση με άλλες δημοφιλείς γλώσσες προγραμματισμού όπως οι C, C++ και Java (Αγγελιδάκης, 2015, σ.4).

Αναπτύσσεται με βάση το μοντέλο της κοινότητας προγραμματιστών που εργάζονται για την ανάπτυξή της όπου αναπτύσσεται ως ανοιχτό λογισμικό (open source). Η διαχείρισή της Python συντονίζεται από τον μη κερδοσκοπικό οργανισμό Python Software Foundation (PSF). Με την υποστήριξη από την ίδια κοινότητα των προγραμματιστών η κοινότητα αυτή αυξάνεται καθημερινά και η Python βρίσκεται σε υψηλή θέση στη λίστα των δημοφιλέστερων γλωσσών προγραμματισμού. Στην Python πέρα από τα πληθώρα των θετικών χαρακτηριστικών της οφείλουμε να αναφέρουμε και μειονεκτήματα της. Ο χρόνος εκτέλεσης των προγραμμάτων της Python μπορεί να μην είναι πάντα τόσο γρήγορος όσο είναι στις μεταγλωττιζόμενες (compiled) γλώσσες όπως η C και η C++. Αυτό οφείλεται στο ότι ένα πρόγραμμα σε Python δεν μεταγλωττίζεται σε δυαδικό κώδικα μηχανής που εκτελείται άμεσα από τον επεξεργαστή του υπολογιστή (Αγγελιδάκης, 2015, σ.5). Οι υπολογιστές μπορούν να τρέξουν μόνο προγράμματα τα οποία είναι γραμμένα σε γλώσσες χαμηλού επιπέδου. Έτσι, προγράμματα τα οποία είναι γραμμένα σε μία γλώσσα υψηλού επιπέδου πρέπει πρώτα να επεξεργαστούν για να μπορούν να τρέξουν. Αυτή η επιπλέον επεξεργασία παίρνει κάποιο χρόνο, το οποίο είναι ένα μικρό μειονέκτημα των γλωσσών υψηλού επιπέδου. Τα πλεονεκτήματα είναι τεράστια. Πρώτον, είναι πολύ ευκολότερο να προγραμματίζεις σε μία γλώσσα υψηλού επιπέδου. Προγράμματα γραμμένα σε μία γλώσσα υψηλού επιπέδου χρειάζονται λιγότερο χρόνο για να γραφτούν, είναι μικρότερα και διαβάζονται ευκολότερα, και είναι πιο πιθανό να είναι σωστά. Δεύτερον, οι γλώσσες υψηλού επιπέδου είναι φορητές, που σημαίνει ότι μπορούν να τρέξουν σε διαφορετικά είδη υπολογιστών με μερικές ή καθόλου τροποποιήσεις. Προγράμματα χαμηλού επιπέδου μπορούν να τρέξουν μόνο σε ένα είδος υπολογιστή και πρέπει να ξαναγραφτούν για να τρέξουν σε κάποιον άλλο (Downey, 2012, p.1). Έτσι, το μειονέκτημα χρόνου εκτέλεσης αντισταθμίζεται από τη εξοικονόμηση χρόνου που έχουμε κατά την ανάπτυξη μιας εφαρμογής σε Python.

1.7.2 Ιστορία της Python

Η δημιουργία της Python ξεκίνησε το 1989 από τον Ολλανδό Guido van Rossum στο ερευνητικό κέντρο Centrum Wiskunde & Informatica (CWI), το οποίο επιτελεί βασική έρευνα στο πεδίο των Μαθηματικών και της Επιστήμης Υπολογιστών. Το όνομα της γλώσσας προέρχεται από την ομάδα Άγγλων κωμικών Μόντυ Πάιθον που πρωταγωνιστούσε σε μια σειρά του BBC της Μεγάλης Βρετανίας τη δεκαετία του 70. Η Python θεωρείται

διάδοχος της γλώσσας προγραμματισμού ABC, μια και αυτή υπήρξε η βασική πηγή έμπνευσης για τη δημιουργία της. Το μοντέλο της Python σχεδιάστηκε με τέτοιο τρόπο ώστε να είναι ευέλικτα επεκτάσιμο, να παρέχει ενσωματωμένα στοιχεία (εντολές, τύπους αντικειμένων, κ.λπ.) αλλά και να δίνει τη δυνατότητα στους προγραμματιστές να προσθέτουν τα δικά τους στοιχεία ανάλογα με τις ανάγκες τους και το σύστημα που χρησιμοποιούν. Η πρώτη έκδοση κυκλοφόρησε το 1991 ενώ στις 16 Οκτωβρίου του 2000 κυκλοφόρησε η Python 2.0 (Αγγελιδάκης, 2015, σ.6).

	Version	Downloads
1	2.7	419,227,040
2	3.5	29,934,424
3	3.4	20,095,827
4	2.6	12,178,744
5	3.3	1,327,582
6	3.6	326,967

Πίνακας όπου παρατηρείται ο αριθμός των downloads ανά έκδοσης της Python.

1.7.3 Βασικά χαρακτηριστικά της Python

- Απλή

Η Python είναι μια αρκετά απλή και μινιμαλιστική γλώσσα Η ομοιότητα της Python με ψευδοκώδικα είναι ένα από τα πιο ισχυρά σημεία της. Καθώς δίνει την δυνατότητα στον προγραμματιστεί να επικεντρωθεί στη λύση του προβλήματος αντί στην ίδια τη γλώσσα που το εκτελεί.

- Εύκολη στην εκμάθηση

Η Python έχει μια πολύ απλή σύνταξη .

- Ελεύθερη και Ανοικτού Κώδικα

Η Python είναι ένα παράδειγμα ΕΛ/ΛΑΚ(Ελεύθερο Λογισμικό και Λογισμικό Ανοικτού Κώδικα ή open source). Το ΕΛ/ΛΑΚ βασίζεται στην ιδέα μιας κοινότητας που μοιράζεται τη γνώση. Σκοπός της Python κοινότητας είναι η συνεχείς βελτίωση για τη δημιουργία ενός όλο και καλύτερου λογισμικού.

- Γλώσσα υψηλού επιπέδου

Τέρμα πλέον τα segmentation faults .Στην Python δεν απαιτεί έλεγχο για τη διαχείριση της μνήμης που χρησιμοποιείται από τα προγράμματά σας.

- Φορητή

Λόγω του ανοικτού της κώδικα, μέσα στα χρόνια η Python έχει αλλαχθεί αρκετά προκειμένου να μπορεί να λειτουργεί. Έτσι με αρκετή προσοχή θα μεταθέτεται το κώδικα σας σε άλλες πλατφόρμες χωρίς σχεδόν να χρειαστεί καμία αλλαγή.

- Εκτεταμένες βιβλιοθήκες

Η Python διαθέτει μια τεράστια βιβλιοθήκη όπου μπορεί να παρέχει βοήθεια για τη δημιουργία προγραμμάτων σχετικά με κανονικές εκφράσεις, δημιουργία τεκμηρίωσης, δοκιμές μονάδων, νημάτωση (threads), βάσεις δεδομένων, περιηγητές ιστού, CGI, FTP, email, XML, XML-RPC, HTML, αρχεία WAV, κρυπτογράφηση, γραφικές διεπαφές χρήστη.

- Αντικειμενοστρεφής

Στις αντικειμενοστρεφείς γλώσσες, τα προγράμματα δομούνται πάνω σε αντικείμενα τα οποία συνδυάζουν δεδομένα και λειτουργίες. Η Python έχει έναν απλό αλλά ταυτόχρονα και πολύ ισχυρό τρόπο για αντικειμενοστρεφή προγραμματισμό, ειδικά όταν συγκρίνεται με άλλες γλώσσες προγραμματισμού, όπως η C++ ή η Java (Cyber Python).

Κεφάλαιο 2: Ανάλυση απαιτήσεων & προδιαγραφές

2.1 Τεχνικά Στοιχεία Υλοποίησης Εφαρμογής

Η υλοποίηση του προγράμματος θα γίνει με την χρήση της Python. Είναι μια αρκετά δημοφιλείς γλώσσα προγραμματισμού και μπορεί να χρησιμοποιηθεί για την ανάπτυξη ολοκληρωμένων εφαρμογών. Γι' αυτό το λόγο αποτελεί μια καλή επιλογή. Η σχεδίαση της βάσης δεδομένων θα γίνει με την χρήση της SQLite. Η SQLite είναι πολύ δημοφιλής και ένα αρκετά εύκολο Συστήματα Διαχείρισης Βάσεων Δεδομένων. Ο συνδυασμός αυτών των δυο εργαλείων θα είναι το κλειδί για την δημιουργία αυτού του συστήματος. Κάθε γραμμή κώδικα είναι σαν ένα μικρό κομμάτι LEGO. Ένα-ένα κομμάτι κτίζει τον πύργο που ονομάζετε Clover Software. Με σταθερά βήματα, υπομονή και με την σωστή χρήση των δυο εργαλείων ο στόχος επετεύχθη.

2.2 Βασικές απαιτήσεις

"Εταιρία παροχής ρεύματος. Μια εταιρία ρεύματος πουλά ρεύμα σε πελάτες. Θα έχει τρεις βάσεις στην Ελλάδα και σε κάθε μια από αυτές θα ανήκει ένας αριθμός Χ χρηστών. Οι βάσεις αυτές θα είναι Αθήνα, Μακεδονία, Κρήτη. Το σύστημα θα διαχειρίζεται αναλυτικά στοιχεία των πελατών αλλά και τις καταναλώσεις που έχουν κάνει. Στο τέλος θα υπολογίζονται τα συνολικά kw που καταναλωθήκαν από τον κάθε πελάτη καθώς και η συνολική χρέωση για τον κάθε πελάτη."

Αυτή ήταν η αρχική ιδέα (αρχικό σενάριο). Το πρώτο αρχείο δηλαδή που στάλθηκε προκειμένου να ξεκινήσουν οι διαδικασίες υλοποίησης του συστήματος. Έτσι αποφασίσαμε να κάνουμε αρχικά τη βάση δεδομένων και μετά το γραφικό περιβάλλον διεπαφής με το χρήστη.

Όσον αφορά τη βάση δεδομένων έγιναν πολλές προσπάθειες προκειμένου να υλοποιηθεί με τον πλέον αποδοτικό αλλά και συνοπτικό τρόπο. Τελικά καταλήξαμε πως η βάση πρέπει να είναι αρκετά απλή και κατανοητή προκειμένου να μπορούμε να διαχειριστούμε τα στοιχεία της με ευκολία, καθώς η απαίτηση αυτής της πτυχιακής ήταν να διαχειριστούμε ολόκληρο τον κύκλο ζωής και ανάπτυξης μίας εφαρμογής λογισμικού. Για αυτό το λόγο, η ευκολία στην κλήση του κάθε στοιχείου μέσα στη βάση δεδομένων είναι αρκετά σημαντική κυρίως όταν θα έπρεπε να τροποποιήσουμε τα στοιχεία με περίπλοκες διαδικασίες. Σκοπός ήταν να ελαχιστοποιήσουμε το κίνδυνο ώστε να μη χαθεί κάποια πληροφορία μέσα από τη διαχείριση

της πολυπλοκότητας των διαδικασιών. Στα διαγράμματα δεν μπορεί να φανεί αυτό αλλά η ροή των δεδομένων μέσα στο σύστημα είναι αρκετά περίπλοκη καθώς πολλά από αυτά τα στοιχεία φιλτράρονται με σκοπό να παράγουν νέα στοιχεία. Και για όλα τα παραπάνω καταλήξαμε στην παρακάτω ιδέα.

Δημιουργία μιας εταιρίας που αναπτύσσει CRM/Software συστήματα για άλλες εταιρίες. Μέσα στο σύστημα που θα φτιαχτεί, ο χρήστης θα μπορεί να αλληλοεπιδρά με αυτό, να παράγει οποιαδήποτε πληροφορία θελήσει ώστε να είναι ωφέλιμη για την εύκολη αναζήτηση και κατανόηση μεγάλου όγκου δεδομένων (παράδειγμα η αναζήτηση ή η προβολή διαγραμμάτων). Να μπορεί να ενημερώνεται για οποιαδήποτε πληροφορία σχετίζεται με την χρήση του συστήματος αλλά και τους περιορισμούς που έχει το ίδιο το σύστημα. Κάτι τέτοιο βοηθά τον χρήστη να μην μπερδεύεται στη χρήση της εφαρμογής. Επίσης αναφέρει τα λάθη του χρήστη αλλά και τις ενέργειες που δεν μπορεί να εκτελέσει για λόγους ασφαλείας της βάσης. Θέλαμε να κάνουμε ένα πρόγραμμα όπου θα είναι αρκετά εύκολο προς χρήση. Το widget να είναι εύκολο και κατανοητό στη χρήση του από τον χρήστη. Στόχος ήταν να ελαχιστοποιηθεί στο μέγιστο ο χρόνος εκτέλεσης μιας διαδικασίας.

Ο χρόνος είναι χρήμα, έτσι το σύστημα δεν θα έπρεπε να απαιτεί ιδιαίτερο χρόνος ή προσπάθεια για την εκμάθηση του. Με έναν απλό τρόπο ο χρήστης θα πρέπει να αποσπά μέσα από το σύστημα οποιαδήποτε πληροφορία σχετίζεται με τα καθήκοντα του. Έτσι μια καθυστέρηση στην όλη διαδικασία θα προκαλούσε σύγχυση. Το σύστημα είναι απλό, φτιαγμένο με σκοπό να εξυπηρετεί της απαιτήσεις του κάθε χρήστη. Είτε αυτές οι απαιτήσεις αφορούν το λογιστήριο, είτε τους πωλητές, είτε οποιοδήποτε άλλο τμήμα της επιχείρησης το οποίο εμπλέκεται ή σχετίζεται με το συγκεκριμένο προϊόν λογισμικού.

2.3 Ανάλυση απαιτήσεων & προδιαγραφών εφαρμογής

Η εφαρμογή θα πρέπει να καταγράφει τα στοιχεία των περιοχών (κωδικό και περιγραφή), των πελατών (ονοματεπώνυμο, email, τηλέφωνο κλπ), τα στοιχεία καταναλώσεων ηλεκτρικής ενέργειας (για διάφορα χρονικά διαστήματα) και τα στοιχεία χρεώσεων (για συγκεκριμένες χρονικές περιόδους). Αναλυτικότερα οι πίνακες με τα στοιχεία που τον πλαισιώνουν:

Area

- areaId (Κωδικός περιοχής - ακέραιος αριθμός)
- areaName (Όνομα περιοχής - Κείμενο/Συμβολοσειρά)

Customer

- customerId (Κωδικός πελάτη - ακέραιος αριθμός)
- firstName (Όνομα πελάτη - Κείμενο/Συμβολοσειρά)
- lastName (Επώνυμο πελάτη - Κείμενο/Συμβολοσειρά)
- address (Διεύθυνση πελάτη - Κείμενο/Συμβολοσειρά)
- postcode (Διεύθυνση πελάτη - Κείμενο/Συμβολοσειρά)
- city (Πόλη - Κείμενο/Συμβολοσειρά)
- pricePerKW (Τιμή χρέωσης ανά KW - Δεκαδικός αριθμός)
- areaId (κωδικός περιοχής - ακέραιος αριθμός)

Consumption

- consumeId (Κωδικός κατανάλωσης - ακέραιος αριθμός)
- startDate (Ημερομηνία έναρξης κατανάλωσης – Ημερομηνία)
- endDate (Ημερομηνία Λήξης κατανάλωσης – Ημερομηνία)
- totalKW (Συνολικά KW κατανάλωσης - Δεκαδικός αριθμός)
- customerId (Κωδικός πελάτη - ακέραιος αριθμός)

Charge

- chargeId (Κωδικός χρέωσης - ακέραιος αριθμός)
- chargeDate (Ημερομηνία χρέωσης – Ημερομηνία)
- KW (KW κατανάλωσης - Δεκαδικός αριθμός)
- totalPrice (Συνολική τιμή κατανάλωσης μιας χρέωσης - Δεκαδικός αριθμός)
- customerId (Κωδικός πελάτη - ακέραιος αριθμός)

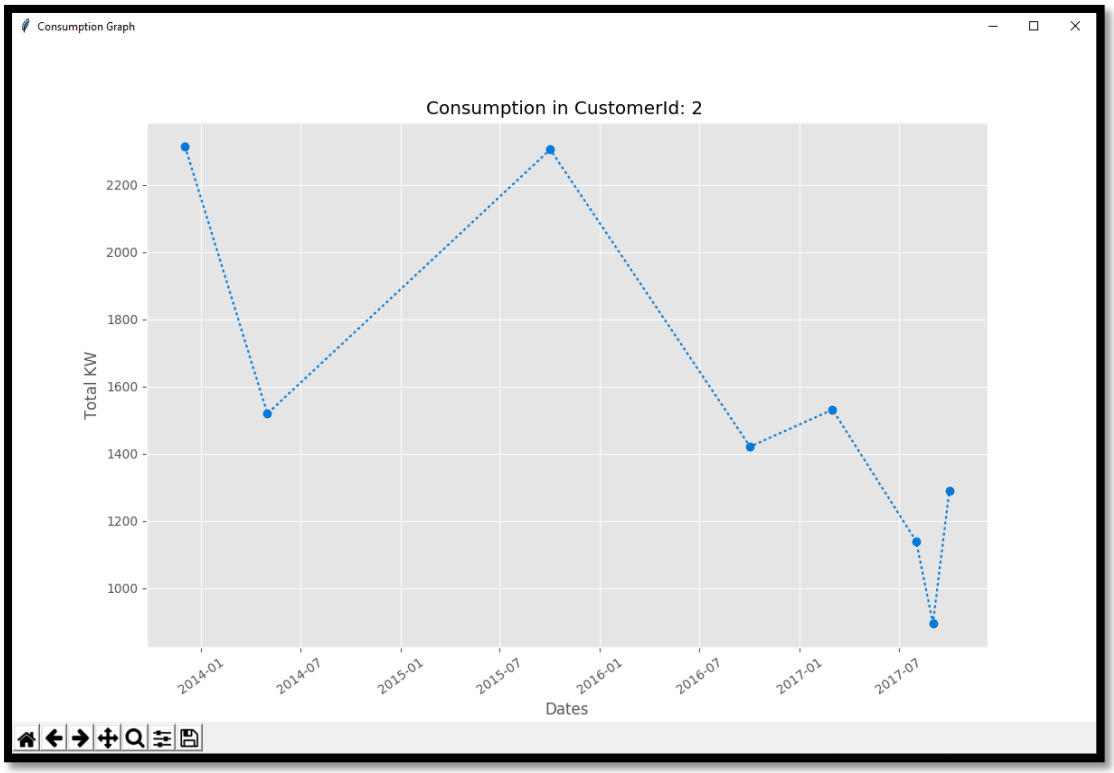
Για όλα αυτά τα στοιχεία θα πρέπει να γίνονται οι βασικές λειτουργίες της εισαγωγής νέων δεδομένων, η αναζήτηση δεδομένων με συγκεκριμένα φίλτρα αναζήτησης, η τροποποίηση και η διαγραφή επιλεγμένων δεδομένων μέσα από την βάση.

Ακόμη θα πρέπει να γίνονται διάφοροι έλεγχοι ορθότητας και ακεραιότητας των νέων δεδομένων προς εισαγωγή καθώς και η δημιουργία συγκεντρωτικών δεδομένων από συνδυασμένες δομές των αποθηκευμένων δεδομένων. Ειδικά για τις χρεώσεις θα πρέπει να γίνεται αυτόματα και με την ελάχιστη δυνατή απαίτηση από το χρήστη, η αναζήτηση, ομαδοποίηση και εξαγωγή συγκεκριμένων αθροισμάτων χρησιμοποιώντας δεδομένα αποθηκευμένα στη βάση δεδομένων που αφορούν τις καταναλώσεις του πελάτη. Θα πρέπει

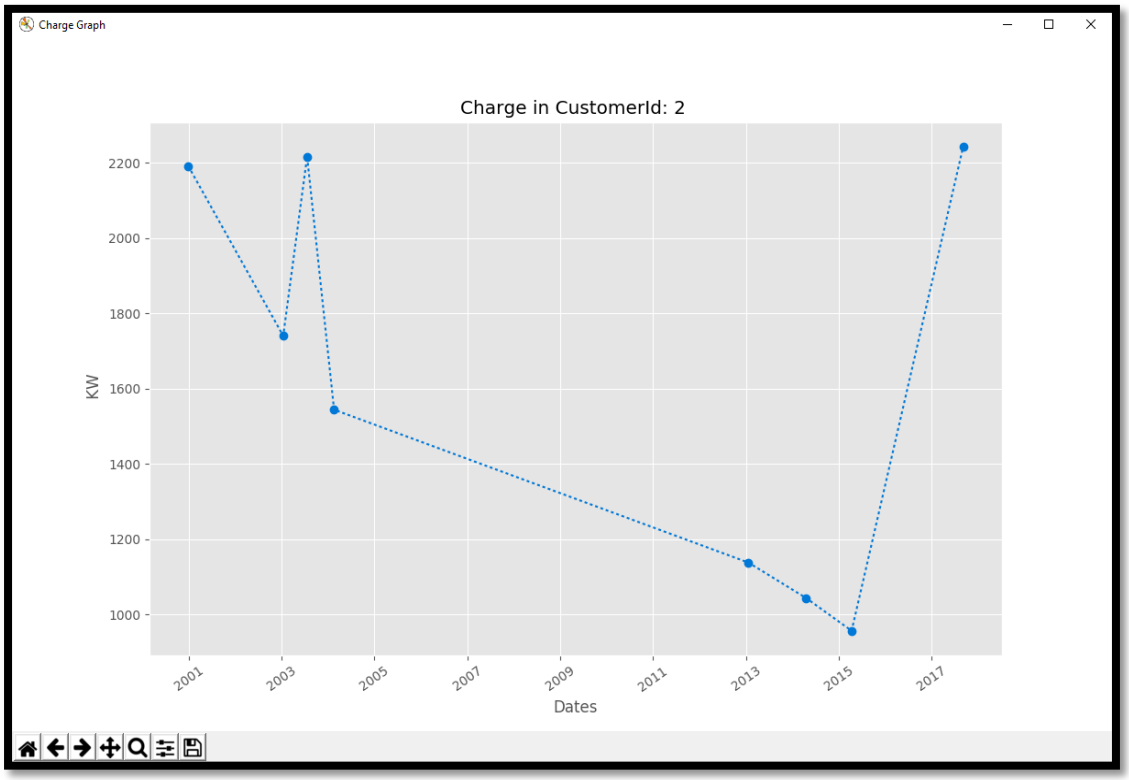
να μπορούν να εξαχθούν συνολικά αθροίσματα τα οποία θα μετασχηματίζονται σε απλές εγγραφές για τις χρεώσεις. Ακόμη θα πρέπει να γίνεται η δημιουργία συγκεντρωτικών αθροισμάτων καθώς και η απεικόνισή τους σε γραφήματα για εύκολη και γρήγορη εξαγωγή επιχειρηματικών συμπερασμάτων.

Ακόμη θα πρέπει να δημιουργούνται γραφήματα για τους πίνακες των καταναλώσεων και των χρεώσεως. Θα απεικονίζουν αναλυτικά τις χρονικές περιόδους που έγιναν οι καταναλώσεις η οι χρεώσεις στο πελάτη. Η διαδικασία αυτή θα γίνεται απλά επιλέγοντας το Id του πελάτη από τον αντίχτυπο πίνακα. Δηλαδή αν θες να δεις τις καταναλώσεις για τον πελάτη Χ ο χρήστης λογιστηρίου θα πάει στον πίνακα Consumption προκειμένου να προβάλει το πίνακα καταναλώσεων.

Διάγραμμα 1 Consumption



Διάγραμμα 2 Charge

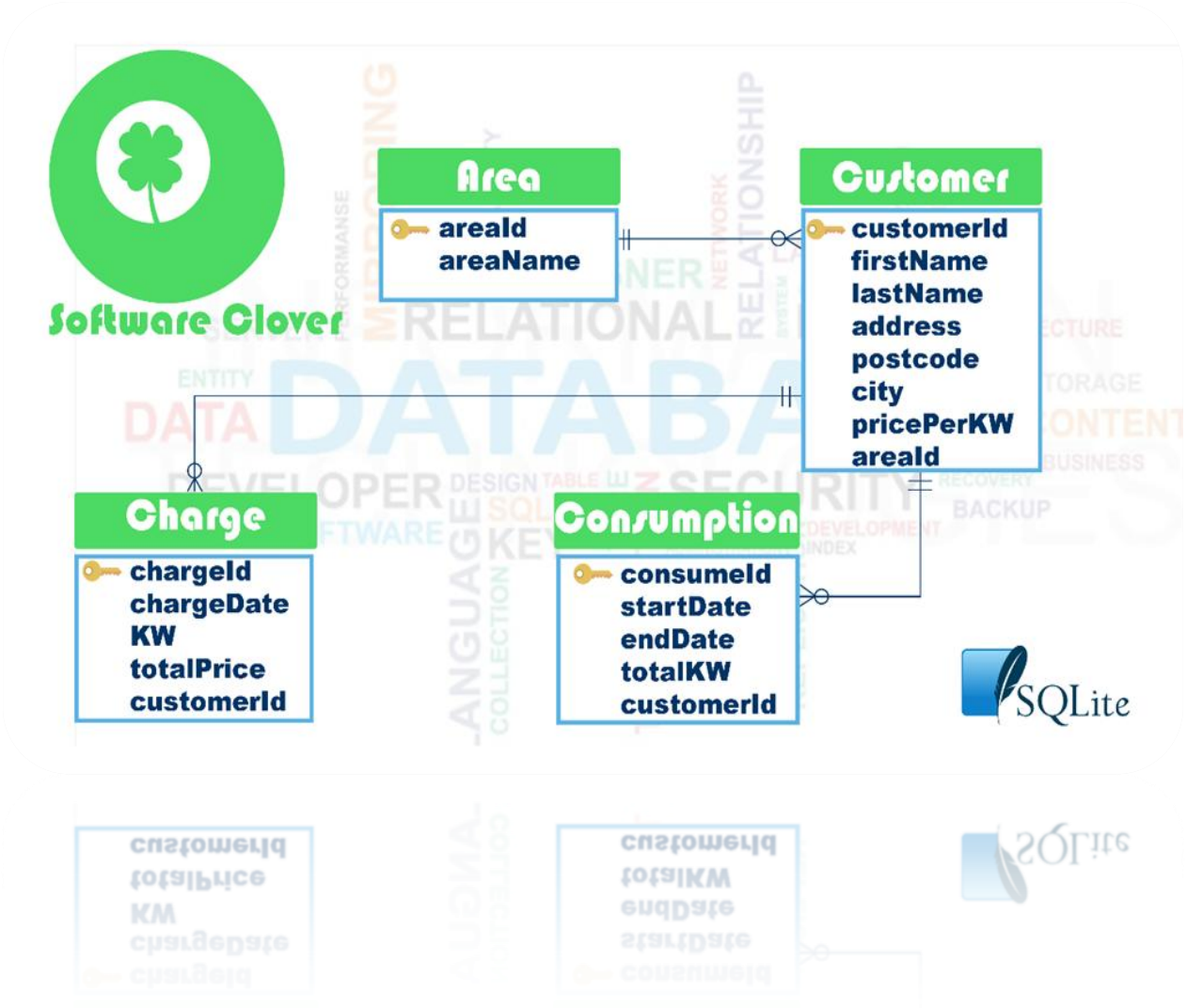


Στην περίπτωση όμως του πίνακα χρεώσεων ο χρήστης θα μπορεί επίσης να κάνει προβολή των συνολικών kw (Total used kw) που έχει καταναλώσει ο πελάτης αλλά και την αντίστοιχη συνολική τιμή (Total price) που έχει τεθεί να πληρώσει ο συγκεκριμένος πελάτης για όλη τη περίοδο όπου η συγκεκριμένη εταιρία παρέχει κάποιες υπηρεσίες για εκείνον. Επίσης θα μπορεί να γίνει η εξαγωγή δεδομένων σε Excel αρχείο. Για την προβολή όλων των παραπάνω το μόνο που θα χρειαστεί είναι να μεταβεί ο χρήστης στον πίνακα των χρεώσεων και στην συνέχεια να επιλέξει τον customer της επιλογής του.

2.4 Ανάλυση Σχήματος Βάσης Δεδομένων

Σύμφωνα με τις παραπάνω απαιτήσεις και γενικές προδιαγραφές, σχεδιάστηκε το παρακάτω διάγραμμα οντοτήτων συσχετίσεων για τις βασικές οντότητες της βάσης δεδομένων.

2.4.1 Διάγραμμα Οντοτήτων Συσχετίσεων



Αποφασίστηκε η βάση δεδομένων να έχει την πιο απλή δομή που θα μπορούσαμε να έχουμε, ακολουθώντας το παρακάτω σχεσιακό σχήμα για να επιτύχουμε μία λειτουργικά και σχεδιαστικά σωστή και ταυτόχρονα εύκολη στη διαχείριση βάση δεδομένων. Εφαρμόσαμε την πρώτη κανονική μορφή ώστε να επιτύχουμε το καλύτερο δυνατό αποτέλεσμα, δεν προχωρήσαμε σε άλλες κανονικές μορφές μιας και δε θέλαμε να γίνει το αποτέλεσμα ιδιαίτερα δύσκολο στη χρήση.

2.4.2 Σχεσιακό Σχήμα

Εταιρία παροχής ρεύματος – Σχεσιακό Σχήμα

Area (areaId, areaName)

Customer (customerId, firstName, lastName, address, postcode, city, pricePerKW, areaId)

Consumption (consumeId, startDate, endDate, totalKW, customerId)

Charge (chargeId, chargeDate, KW, totalPrice, customerId)

Παράλληλα με το παραπάνω σχεσιακό σχήμα υπάρχουν και οι παρακάτω περιορισμοί (ακεραιότητας και όχι μόνο) όσον αφορά τις βασικές οντότητες της βάσης δεδομένων.

Περιορισμοί:

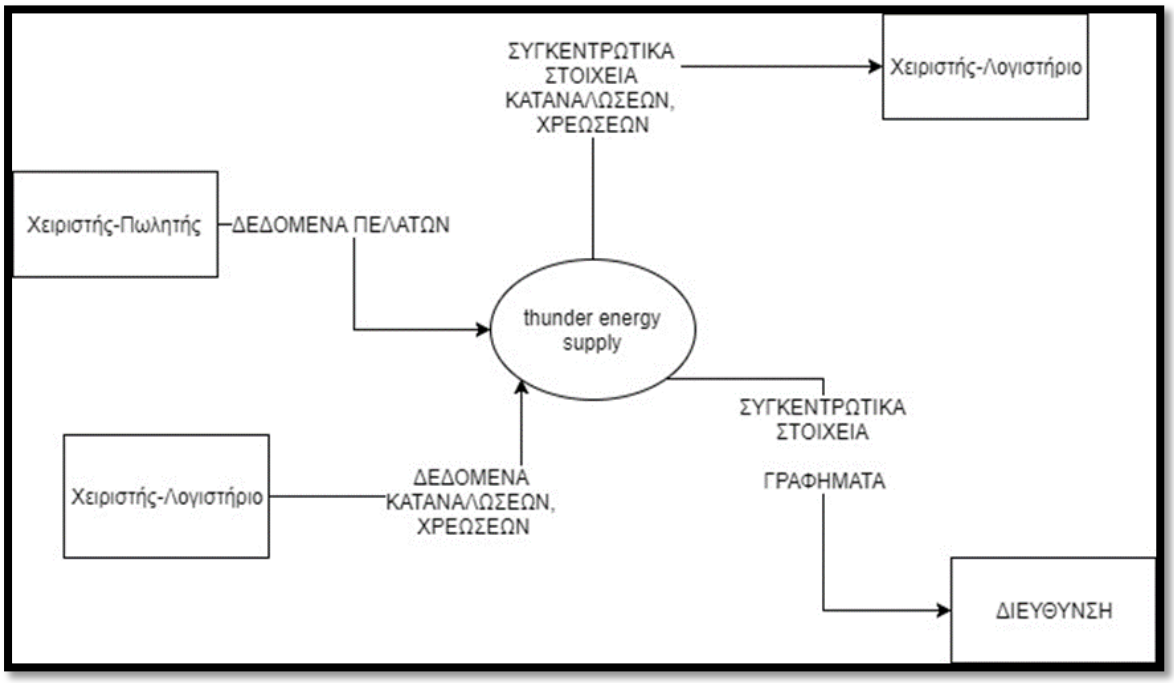
- Δεν επιτρέπεται 1 πελάτες να ανήκει σε 2 περιοχές
- Μία χρέωση μπορεί να περιλαμβάνει πολλές περιόδους κατανάλωσης
- Η ημερομηνία μίας χρέωσης πρέπει να συμπίπτει με μία καταληκτική ημερομηνία κατανάλωσης
- Δημιουργία μοναδικότητας συνδυασμένων στοιχείων μέσα στην βάση προκειμένου να αποτρέψουμε την περίπτωση διπλής εισαγωγής τιμών μέσα στην βάση δεδομένων.

Area_add	CREATE UNIQUE INDEX Area_add ON Area(areaName)
Charge_add	CREATE UNIQUE INDEX Charge_add ON Charge(chargeDate,KW)
Consumption_add	CREATE UNIQUE INDEX Consumption_add ON Consumption(startDate,endDate,totalKW,customerId)
Customer_add	CREATE UNIQUE INDEX Customer_add ON Customer(lastName,phone)

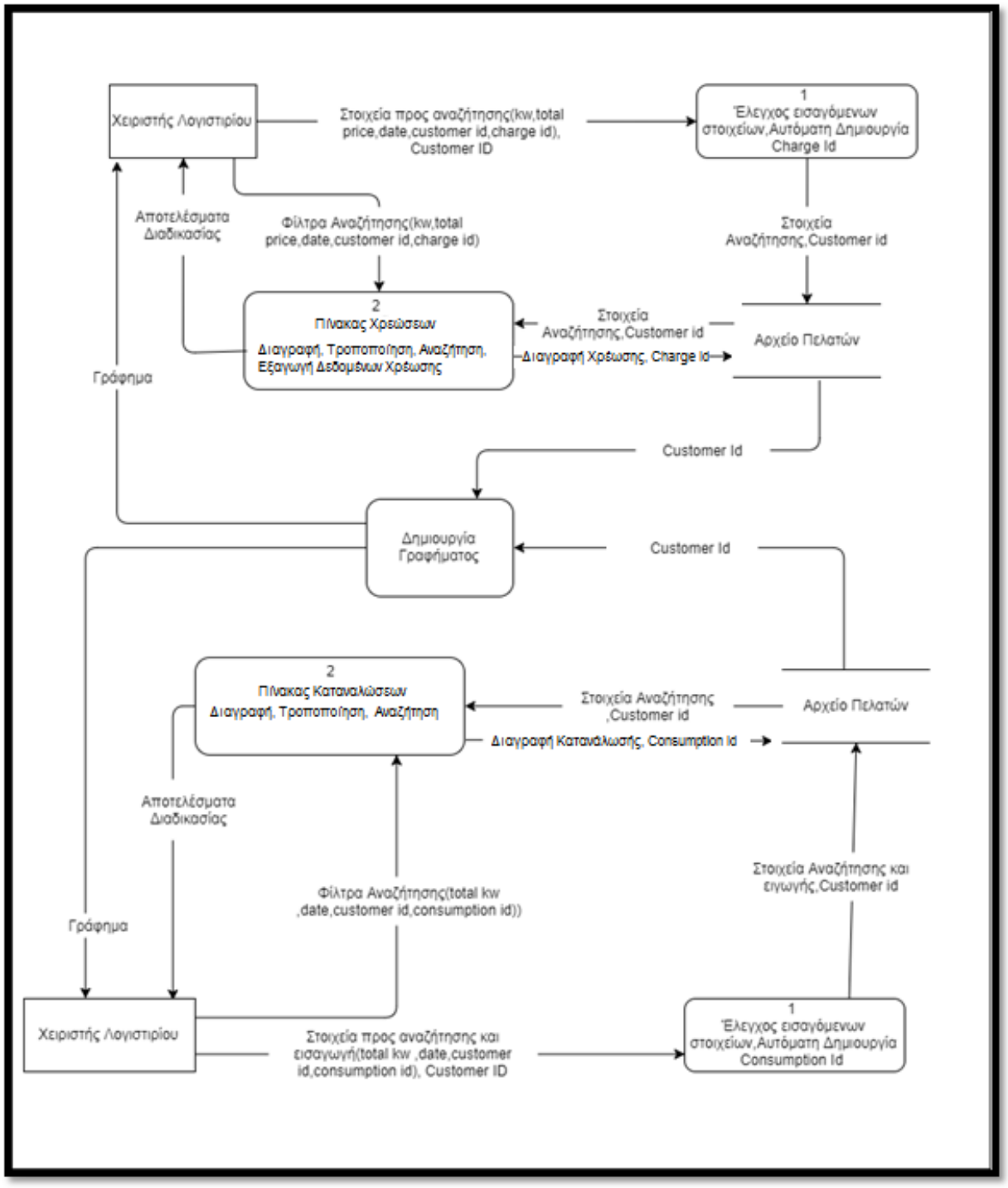
2.5 Ανάλυση διαδικασιών διαχείρισης πληροφορίας

2.5.1 Διαγράμματα Ροής Δεδομένων Επιπέδου 0

- Οι πράκτορες (Χειριστής-Πωλητής, Χειριστής-Λογιστηρίου) εισάγουν τα στοιχεία στο σύστημα ενημερώνοντας το.
- Οι πράκτορες (Χειριστής-Λογιστηρίου, Διεύθυνση) λαμβάνουν τα συγκεντρωτικά στοιχεία από το σύστημα.



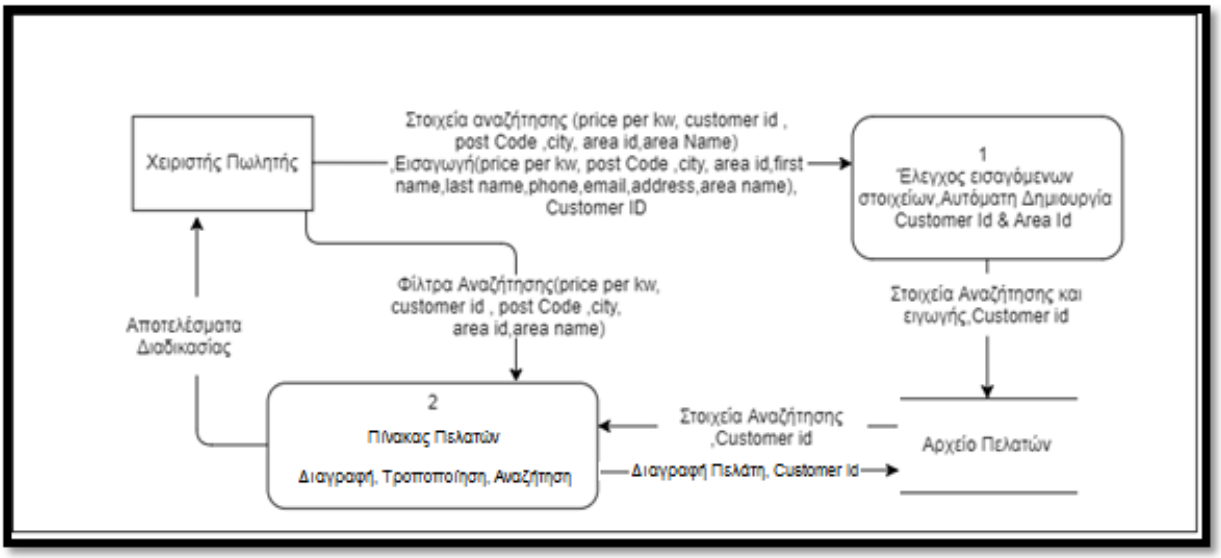
2.5.2 Διαγράμματα Ροής Δεδομένων Επιπέδου 1



Βασικές διαδικασίες Χειριστής Λογιστηρίου

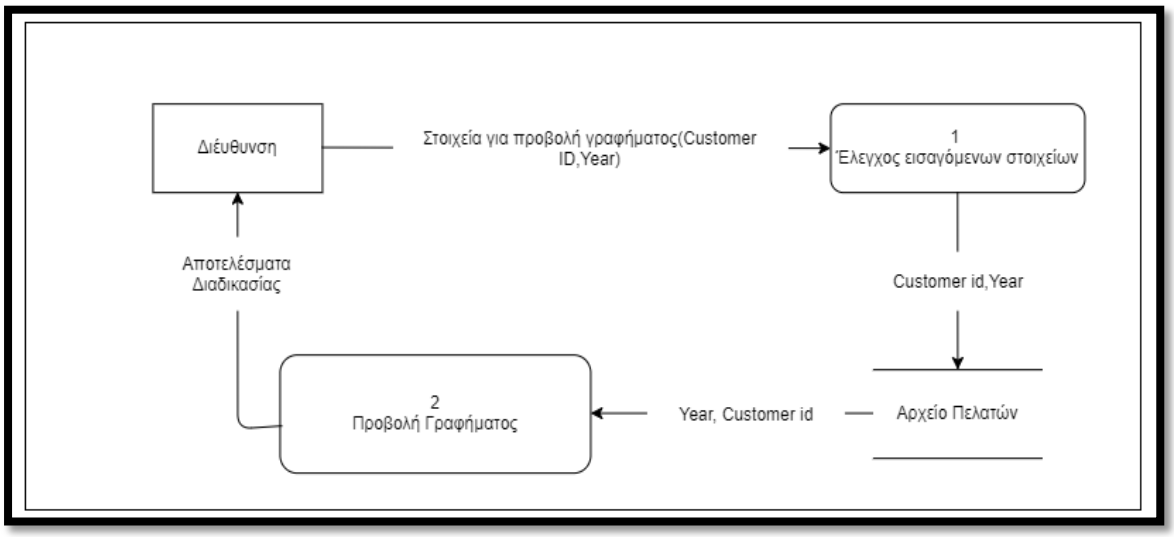
1. Τα στοιχεία εισάγονται από το Λογιστήριο.
2. Τα στοιχεία ελέγχονται από την βάση προκειμένου να γίνει η ταυτοποίηση και η έγκριση τους από αυτή.

3. Τα στοιχεία στέλνονται από τη βάση για τροποποίηση ,αναζήτηση ,εισαγωγή και διαγραφή προβολή γραφημάτων αλλά εισάγονται και τα φίλτρα αναζήτησης πελατών (αν υπάρχουν).
4. Τα αποτελέσματα αποστέλλονται πείσω στον χρήστη



Βασικές διαδικασίες Χειριστή Πωλητή

1. Τα στοιχεία εισάγονται από το τον χειρίστη.
2. Τα στοιχεία ελέγχονται από την βάση προκειμένου να γίνει η ταυτοποίηση και η έγκριση τους από αυτή.
3. Τα στοιχεία στέλνονται από τη βάση για τροποποίηση ,αναζήτηση ,εισαγωγή και διαγραφή αλλά εισάγονται και τα φίλτρα αναζήτησης πελατών (αν υπάρχουν).
4. Τα αποτελέσματα αποστέλλονται πείσω στον χρήστη



Βασικές διαδικασίες Διεύθυνσης

1. Τα στοιχεία εισάγονται από το τη διεύθυνση.
2. Τα στοιχεία ελέγχονται από την βάση προκειμένου να γίνει η ταυτοποίηση και η έγκριση τους από αυτή.
3. Τα στοιχεία στέλνονται από την βάση στην διαδικασία για να γίνει η προβολή των αποτελεσμάτων με γραφική παράσταση.
4. Τα αποτελέσματα αποστέλλονται πείσω στη διεύθυνση.

Κεφάλαιο 3: Η εφαρμογή λογισμικού clover software

Στις επόμενες σελίδες γίνεται μία σύντομη παρουσίαση της εφαρμογής λογισμικού που κατασκευάστηκε χρησιμοποιώντας τις παραπάνω αναλύσεις των βασικών απαιτήσεων που είχαμε. Ας δούμε μία-μία τις βασικές καρτέλες (tabs) της εφαρμογής πιο αναλυτικά.

3.1 Καρτέλα Home

Η καρτέλα Home είναι η τελευταία καρτέλα που προέκυψε στο σύστημα. Η όλη δημιουργία της οφείλεται στο γεγονός ότι έπρεπε να υπάρχει μία καρτέλα που να παρουσιάζει κάποια δεδομένα με τη μορφή διαγραμμάτων. Με την προβολή των δεδομένων αυτών ο χρήστης μπορεί να συγκρίνει τα στοιχεία των πελατών, αλλά και να δει ανά χρονιά τα κέρδη που είχε η εταιρία. Έτσι στο σύστημα έχουν τοποθετηθεί τέσσερεις γραμμές διαχείρισης (Edit Line).

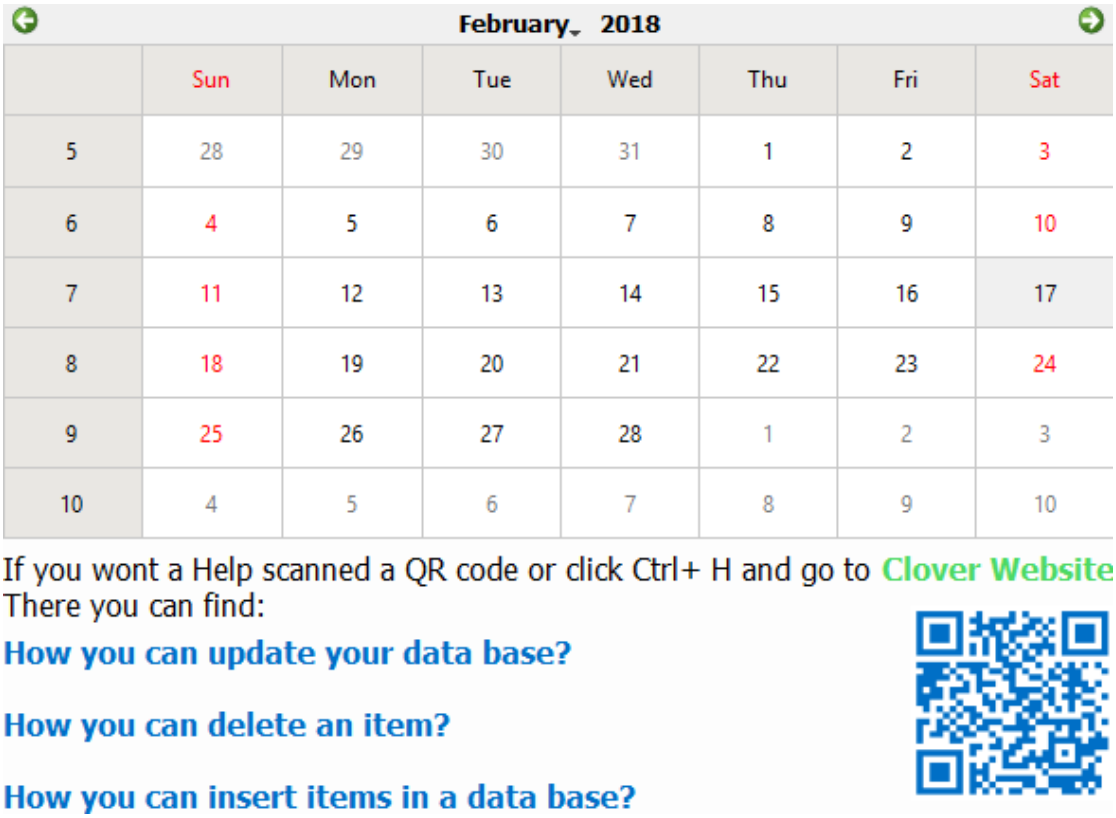
Εικόνα 3.1.01

Στις τρείς πρώτες γραμμές (Εικόνα 3.1.01) ο χρήστης πληκτρολογεί τους κωδικούς (Id) των πελατών σε περίπτωση που συμπληρωθούν και οι τρείς γραμμές διαχείρισης τότε θα εμφανιστεί ένα διάγραμμα όπου θα συγκρίνονται οι καταναλώσεις των πελατών. Στην τέταρτη γραμμή διαχείρισης ο χρήστης αναγράφει μια συγκεκριμένη χρονιά.

Εικόνα 3.1.02

Στην συνέχεια εμφανίζονται τα κέρδη τα οποία αντιστοιχούν στην αντίστοιχη χρονιά που πληκτρολογήθηκε (Εικόνα 3.1.02).

Στην καρτέλα Home ο χρήστης μπορεί επίσης να βρει και ένα QR code που παραπέμπει στην ιστοσελίδα του cover software(Εικόνα 3.1.03).



Εικόνα 3.1.03

Επίσης μπορεί να βρει και άλλες χρήσιμες πληροφορίες για τον τρόπο εκτέλεση διάφορων διαδικασιών στις υπόλοιπες καρτέλες. Ακόμη υπάρχει ένα εικονίδιο που αναγράφει τα κέρδη της εταιρίας thunder ανά χρόνια. Κάνοντας κάποιος click πάνω στην εικόνα τότε θα εμφανιστεί ένα διάγραμμα με τα πιο πρόσφατα στοιχεία της κάθε περιόδου της εταιρίας.

3.2 Καρτέλα Area

Η καρτέλα Area είναι η καρτέλα που αντιστοιχεί στον κύριο πίνακα της βάσης δεδομένων που αναγράφονται οι διάφορες περιοχές. Αυτές οι περιοχές αντιστοιχούν σε ονόματα σημείων πάνω στο χάρτη της Ελλάδας. Σε κάθε ένα από αυτά τα σημεία αντιστοιχούν πελάτες αθροιστικά περισσότεροι από ένας. Σε περίπτωση που διαγράφουν όλοι οι πελάτες από μια περιοχή η πληροφορία αυτή μπορεί θα συνεχίσει να υπάρχει. Καθώς για λογούς ασφάλειας δεν επιτρέπεται η διαγράφη οποιοδήποτε σημείου από τη βάση.

Ενέργειες που μπορούν να πραγματοποιηθούν είναι η Αναζήτηση, η Εισαγωγή και η Τροποποίηση αναλυτικότερα:

Αναζήτηση

Εικόνα 3.2.01

Η αναζήτηση (Εικόνα 3.2.01) μπορεί να γίνει είτε βάση του πεδίου AreaId, είτε βάση του πεδίου AreaName, είτε από τον συνδυασμό και των δύο αυτών πεδίων.

Εισαγωγή

Εικόνα 3.2.02

Ο χρήστης εισάγει την περιγραφή για την περιοχή της επιλογής του αλλά στην συνέχεια το Area Id δημιουργείται χρησιμοποιώντας μια αυτοματοποιημένη διαδικασία της βάσης δεδομένων. Έτσι μπορούμε να εξασφαλίσουμε τη μοναδικότητα που πρέπει να έχει το πεδίο AreaId μέσα στο σύστημα. Με κάθε μια εισαγωγή (Εικόνα 3.2.02) περιοχής που πραγματοποιείται αυτόματα ενημερώνεται και το εισαγωγικό πλαίσιο στην καρτέλα του customer για την νέα πειοχή που εισάχθηκε στο σύστημα.

Τροποποίηση

Εικόνα 3.2.03

Ο χρήστης μπορεί να τροποποιήσει τις περιοχές (Εικόνα 3.2.03)που έχουν καταχωρηθεί μέσα στο σύστημα στην περίπτωση που έχει προκύψει κάποιο λάθος κατά την εισαγωγή στοιχείων τους μέσα στο σύστημα.

3.3 Καρτέλα Customer

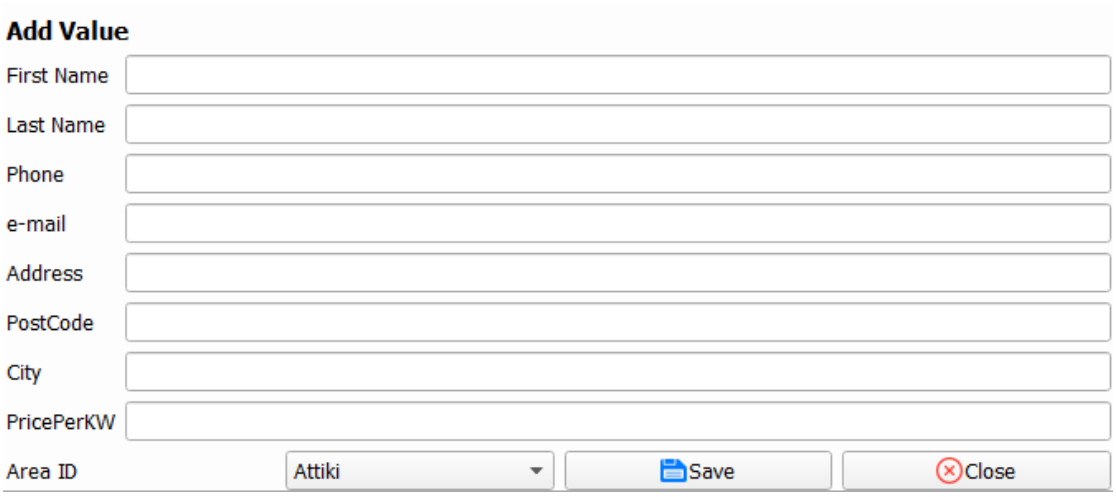
Η καρτέλα Customer είναι η βασικότερη καρτέλα από όλες. Το Customer tab είναι η καρτέλα που στον κύριο πίνακά της αναγράφονται όλες οι πληροφορίες σχετικά με τους πελάτες. Οι ενέργειες που μπορούν να πραγματοποιηθούν είναι η Αναζήτηση, η Εισαγωγή, η Διαγραφή και η Τροποποίηση. Αναλυτικότερα:

Αναζήτηση

Εικόνα 3.3.01

Η αναζήτηση (Εικόνα 3.3.01) μπορεί να γίνει με βάση τον κωδικό πελάτη στου πεδίο CustomerId. Τα αποτελέσματα της αναζήτησης μπορεί να προκύψουν από τον συνδυασμό αναζήτησης τεσσάρων ή και λιγότερων πληροφοριών. Σε περίπτωση που ο χρήστης συμπληρώσει περισσότερες από τέσσερις γραμμές εισαγωγής κειμένου (Edit Line) η αναζήτηση δεν μπορεί να πραγματοποιηθεί.

Εισαγωγή



Εικόνα 3.3.02

Στην εικόνα (Εικόνα 3.3.02) παρατηρείτε της διάφορες γραμμές εισαγωγής κειμένου όπου με την πλήρη ολοκλήρωσή του μπορεί ο χρήστης να εισάγει στοιχεία στο σύστημα. Σε οποιαδήποτε άλλη περίπτωση σφάλματος λόγο ελλιπής καταχώρισης η λάθους βάσει κάποιων περιορισμών που έχουν οριστεί στο σύστημα· η εισαγωγή δεν θα πραγματοποιηθεί. Τότε θα εμφανιστεί ένα προειδοποιητικό μήνυμα ανάλογα με το λάθος που έχει προκύψει κατά την εισαγωγή των στοιχειών.

Τροποποίηση

Εικόνα 3.3.03

Η τροποποίηση που μπορεί να γίνει είναι μόνο βάση του πεδίου Price Per KW καθώς κατά την διάρκεια χρήσης του συστήματος μπορεί να προκύψει κάποια νέα συμφωνία με τον πελάτη σχετικά με τη νέα/τροποποιημένη χρέωση που θα έχει ανά KW. Για την εκτέλεση της διαδικασίας το μόνο που απαιτείται είναι η επιλογή του εκάστοτε πεδίου κωδικού πελάτη (Customer ID) όπου θα γίνει η τροποποίηση. Η επιλογή αυτή θα γίνει μέσα από τον πίνακα που βρίσκεται στην καρτέλα Customer. Στη συνέχεια μέσα στην γραμμή εισαγωγής κειμένου (Εικόνα 3.3.03) θα πρέπει να συμπληρωθεί η νέα τιμή. Ύστερα save και η διαδικασία θα ολοκληρωθεί με επιτυχία.

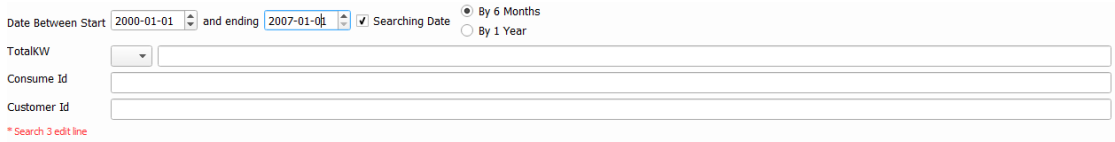
Διαγραφή

Η διαγραφή μπορεί να γίνει βάση του πεδίου κωδικού πελάτη Customer Id. Σε περίπτωση που διαγραφεί κάποιος πελάτης από το σύστημα τότε αυτόματα θα διαγραφεί και οποιαδήποτε πληροφορία εμπλέκεται με αυτόν. Έτσι αν ο χρήστης επιλέξει να διαγράψει ένα πελάτη με ένα συγκεκριμένο Id τότε αυτόματα θα διαγραφούν όλες οι πληροφορίες που αντιστοιχούν στον συγκεκριμένο Customer με το συγκεκριμένο Id και από τον πίνακα Consumption, αλλά και από τον πίνακα Charge.

3.4 Καρτέλα Consumption

Η καρτέλα Consumption αποτελεί την καρτέλα που στον κύριο πίνακά της αναγράφονται όλες οι πληροφορίες σχετικά με τι καταναλώσεις του κάθε πελάτη. Οι ενέργειες που μπορούν να πραγματοποιηθούν είναι η Αναζήτηση, η Εισαγωγή, η Διαγραφή, η Προβολή Διαγράμματος και η Τροποποίηση. Αναλυτικότερα:

Αναζήτηση



Εικόνα 3.4.01

Η αναζήτηση (Εικόνα 3.4.01) μπορεί να γίνει βάση του πεδίου κωδικού Date, TotalKW, Consumption, ConsumId. Τα αποτελέσματα της αναζήτησης μπορεί να προκύψουν από το συνδυασμό αναζήτησης τριών ή και λιγότερων καταγραφών. Σε περίπτωση που ο χρήστης συμπληρώσει περισσότερες από τρείς γραμμές εισαγωγής κειμένου (Edit Line) τότε η αναζήτηση δε μπορεί να πραγματοποιηθεί.

Εισαγωγή

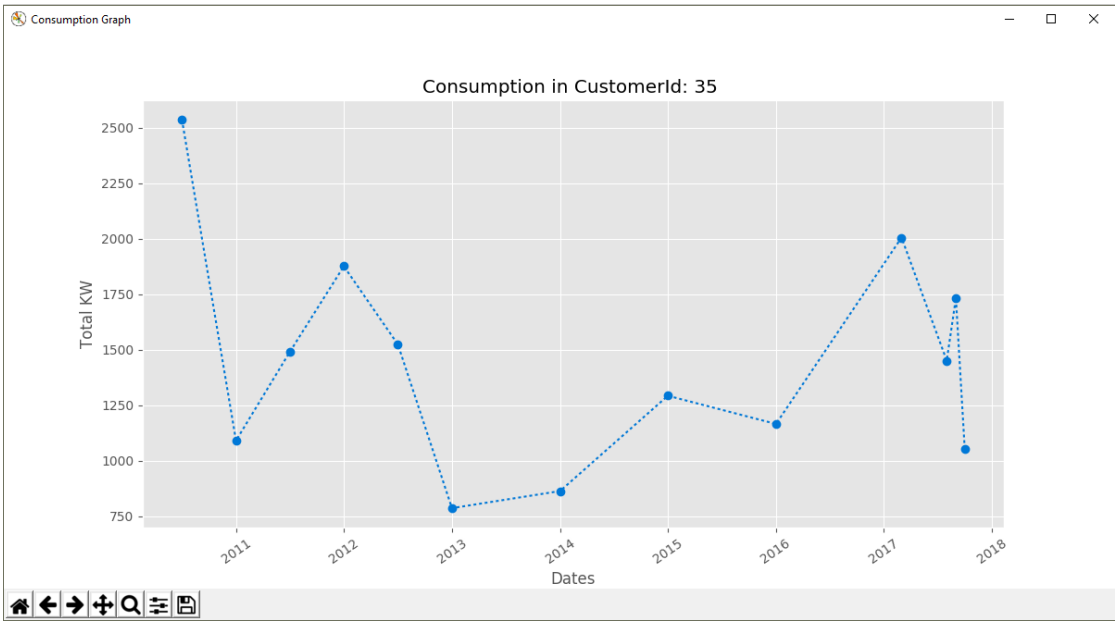
Εικόνα 3.4.02

Στην εικόνα (Εικόνα 3.4.02)παρατηρείτε τις γραμμές εισαγωγής κειμένου που με την πλήρη ολοκλήρωσή του μπορεί ο χρήστης να εισάγει στοιχεία στο σύστημα. Σε οποιαδήποτε άλλη περίπτωση σφάλματος λόγο ελλιπής καταχώρισης η λάθους βάσει κάποιων περιορισμών που έχουν οριστεί στο σύστημα· η εισαγωγή δεν θα πραγματοποιηθεί. Τότε θα εμφανιστεί ένα προειδοποιητικό μήνυμα ανάλογα με το λάθος που έχει προκύψει κατά την εισαγωγή των στοιχειών.

Διαγραφή

Η διαγραφή μπορεί να γίνει βάσει του πεδίου ConsumId. Σε περίπτωση που διαγραφεί κάποια κατανάλωση από ένα συγκεκριμένο πελάτη, τότε δε θα υπάρχει η δυνατότητα επανάκτησης της πληροφορίας. Κατά την διάρκεια αυτής της διαδικασίας και επειδή γίνεται αυτόματα η αποθήκευση των αλλαγών στη βάση δεδομένων, εμφανίζεται ένα προειδοποιητικό μήνυμα προκειμένου να επαληθευτεί η ενέργεια και να δοθεί μία ρητή έγκριση της ενέργειας από το χρήστη.

Προβολή Διαγράμματος



Εικόνα 3.4.03

Για την προβολή των διαγραμμάτων θα χρειαστεί η επιλογή ενός συγκεκριμένου πελάτη μέσω του πεδίου Customer ID. Η επιλογή αυτή θα γίνει μέσα από τον πίνακα που βρίσκεται στην καρτέλα Consumption. Στον πίνακα αναγράφονται όλες οι τιμές καταναλώσεων που είχε κάνει ο πελάτης ανά χρονική περίοδο. Έτσι μπορούν εύκολα να εντοπίζονται οι διακυμάνσεις που έχουν οι τιμές κατανάλωσης ανά χρονική περίοδο. Τα διαγράμματα αυτά είναι χρήσιμα σε μακροχρόνιο επίπεδο καθώς θα μπορούν να εντοπιστούν εύκολα οι περίοδοι όπου υπάρχει αύξηση της κατανάλωσης από το χρήστη (Εικόνα 3.4.03).

Τροποποίηση

Εικόνα 3.4.04

Η τροποποίηση που μπορεί να γίνει είναι μόνο βάση του πεδίου Total KW καθώς κατά την διάρκεια χρήσης του συστήματος μπορεί να προκύψουν λάθη από τις μετρήσεις των υπαλλήλων. Για την εκτέλεση της διαδικασίας το μόνο που απαιτείται είναι η επιλογή του

κωδικού της κατανάλωσης (Consumption Id) για την οποία θα γίνει η τροποποίηση. Η επιλογή αυτή γίνεται μέσα από τον πίνακα που βρίσκεται στην καρτέλα Consumption. Στην συνέχεια μέσα στην γραμμή εισαγωγής κειμένου (Εικόνα 3.4.04) θα πρέπει να συμπληρωθεί η νέα τιμή και τέλος να γίνει η οριστικοποίηση των αλλαγών πατώντας το κουμπί αποθήκευσης.

3.5 Καρτέλα Charge

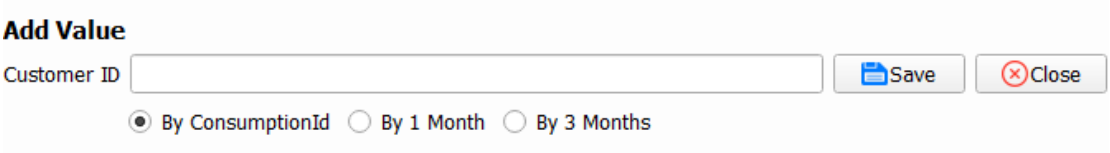
Οι ενέργειες που μπορούν να πραγματοποιηθούν είναι η Αναζήτηση, η Εισαγωγή, η Διαγραφή, η Προβολή Διαγράμματος, η Εξαγωγή Στοιχείων excel και η Τροποποίηση αναλυτικότερα:

Αναζήτηση

Εικόνα 3.5.01

Η αναζήτηση (Εικόνα 3.5.01) μπορεί να γίνει βάση των πεδίων Charge Id (Κωδικός Χρέωσης), Customer Id (Κωδικός Πελάτη), KW, Total Price (Συνολική Τιμή) και χρεώσεων που πραγματοποιήθηκαν σε κάποιο συγκεκριμένο χρονικό διάστημα. Τα αποτελέσματα της αναζήτησης μπορεί να προκύψουν από τον συνδυασμό αναζήτησης τριών ή και λιγότερων πληροφοριών. Σε περίπτωση που ο χρήστης συμπληρώσει περισσότερες από τρείς γραμμές εισαγωγής κειμένου (Edit Line) η αναζήτηση δε μπορεί να πραγματοποιηθεί.

Εισαγωγή



Εικόνα 3.5.02

Στην εικόνα (Εικόνα 3.5.02) παρατηρείτε μόνο μία γραμμή εισαγωγής κειμένου. Σε αυτήν τη γραμμή το μόνο που πρέπει να κάνει ο χρήστης είναι να εισάγει τον πελάτη όπου θέλει να κάνει τη χρέωση του για μια συγκεκριμένη περίοδο κατανάλωσης. Η ημερομηνίες αυτές θα

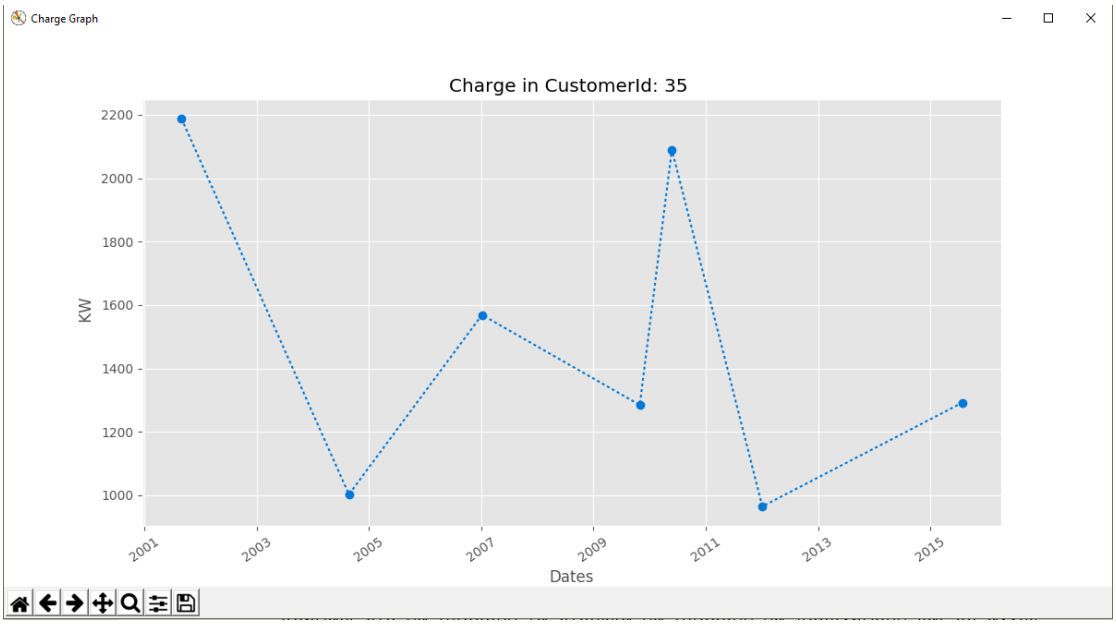
περιλαμβάνουν κάποια χρονικά διαστήματα μετρήσιμα από την τρέχουσα ημερομηνία. Με βάση αυτών των συγκεκριμένων χρονικών διαστημάτων θα προσαρμόζεται η εκάστοτε χρέωση. Καθώς έτσι ο χρήστης θα εισάγει στοιχεία στο σύστημα.

Σε οποιαδήποτε άλλη περίπτωση η διαδικασία εισαγωγής δε θα πραγματοποιηθεί και θα εμφανιστεί ένα προειδοποιητικό μήνυμα ανάλογα με το εκάστοτε λάθος που έχει προκύψει κατά την εισαγωγή των στοιχειών.

Διαγραφή

Η διαγραφή μπορεί να γίνει βάσει του πεδίου ChargeId (Κωδικός Χρέωσης). Σε περίπτωση που διαγραφεί κάποια χρέωση τότε δε θα υπάρχει η δυνατότητα επανάκτησης της διαγραμμένης πληροφορίας. Κατά τη διάρκεια αυτής της διαδικασίας, και επειδή γίνεται αυτόματα η αποθήκευση των αλλαγών στη βάση δεδομένων, εμφανίζεται ένα προειδοποιητικό μήνυμα προκειμένου να επαληθευτεί η έγκριση της ενέργειας από τον χρήστη.

Προβολή Διαγράμματος

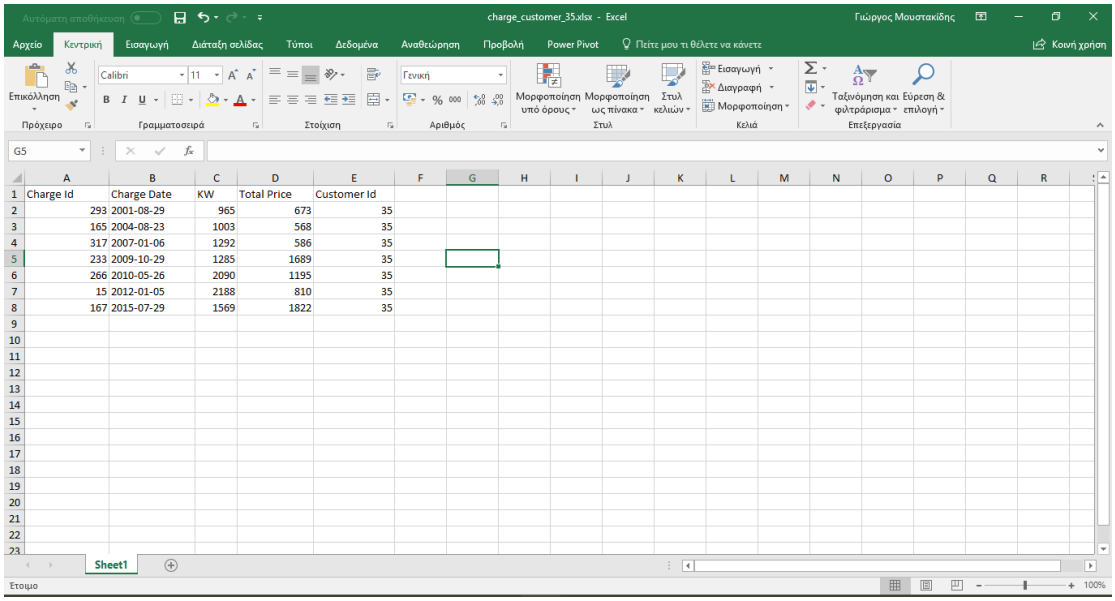


Εικόνα 3.5.03

Για την προβολή των διαγραμμάτων (Εικόνα 3.5.03) χρειάζεται η επιλογή του συγκεκριμένου κωδικού πελάτη Customer ID. Η επιλογή αυτή γίνεται μέσα από τον πίνακα που βρίσκεται στην καρτέλα Charge. Στον πίνακα αναγράφονται όλες οι τιμές χρεώσεων που

είχε δημιουργήσει ο πελάτης ανά μία χρονική περίοδο. Έτσι μπορούν εύκολα να εντοπίζονται οι διακυμάνσεις που έχουν οι τιμές ανά χρονική περίοδο. Τα διαγράμματα αυτά θα είναι χρήσιμα σε μακροχρόνιο επίπεδο καθώς θα μπορούν να εντοπιστούν εύκολα οι περίοδοι όπου υπάρχει μια απότομη αύξηση, μείωση της χρέωσης όπου μπορεί να προκύψει για παράδειγμα από μια αλλαγή της τιμής χρέωσης ανά kw (Price per KW).

Εξαγωγή Στοιχείων σε μορφή Excel



Charge Id	Charge Date	KW	Total Price	Customer Id
293	2001-08-29	965	673	35
165	2004-08-23	1003	568	35
317	2007-01-06	1292	586	35
233	2009-10-29	1285	1689	35
266	2010-05-26	2090	1195	35
15	2012-01-05	2188	810	35
167	2015-07-29	1569	1822	35

Εικόνα 3.5.04

Η εξαγωγή των δεδομένων σε excel (Εικόνα 3.5.04) είναι μια αρκετά χρήσιμη διαδικασία διότι βοηθάει στο να μπορούν τα στοιχεία της εφαρμογής να μπορούν να προβληθούν και σε άλλες συσκευές ή συστήματα ή/και να αποσταλούν μέσω ενός email σε ένα αρχείο excel ώστε οι διαδικασίες και τα αποτελέσματα να παίρνουν σάρκα και οστά σε οποιοδήποτε υπολογιστή θελήσουμε να έχει πρόσβαση σε αυτά τα στοιχεία.

Τροποποίηση

Εικόνα 3.5.05

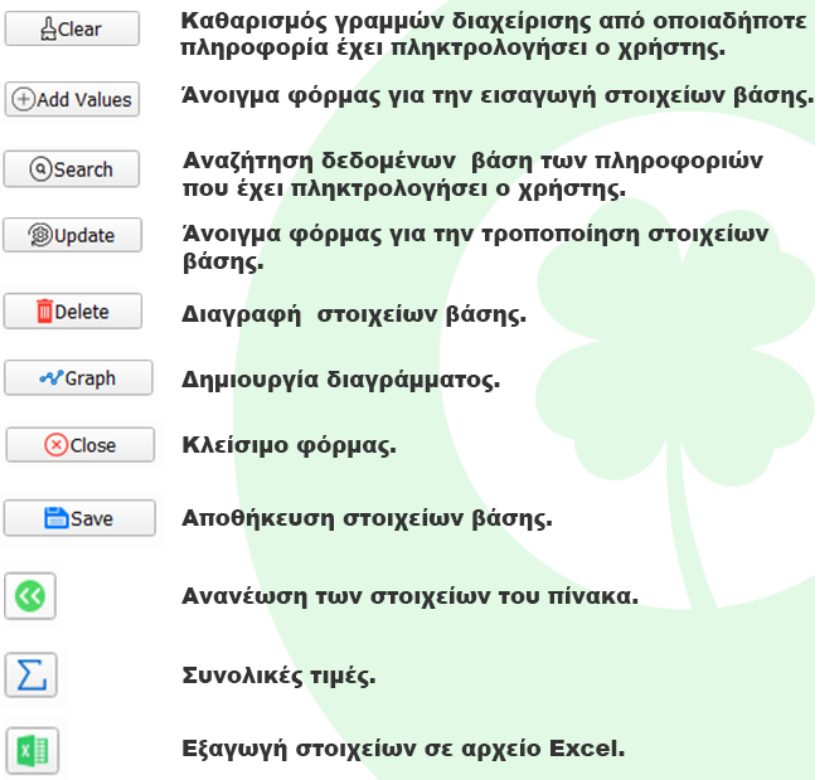
Η τροποποίηση που μπορεί να γίνει είναι μόνο βάση του πεδίου Total Price καθώς μπορεί να προκύψουν λάθη σε πηγές πληροφόρησης που έχει ο χρήσης στην κατοχή του. Κατ' επέκταση εφόσον η διαδικασία χρέωσης είναι αυτοματοποιημένη δίνεται η δυνατότητα να

διορθωθεί το λάθος που πιθανών να προκύψει. Για την εκτέλεση της διαδικασίας τροποποίησης το μόνο που θα απαιτηθεί είναι η επιλογή του κωδικού χρέωσης (Charge Id) για τον οποίο θα γίνει η τροποποίηση. Η επιλογή αυτή θα γίνει μέσω του πίνακα που βρίσκεται στην καρτέλα Charge. Στην συνέχεια μέσα από τη γραμμή εισαγωγής κειμένου (Εικόνα 3.5.05) θα πρέπει να συμπληρωθεί η νέα τιμή και τέλος η οριστικοποίηση των αλλαγών πατώντας το κουμπί αποθήκευσης.

3.6 Κουμπιά (Buttons)

Όλα τα κουμπιά που υπάρχουν στο σύστημα έχουν κάποια σχόλια. Αυτά τα σχόλια αναφέρουν διάφορες οδηγίες σχετικά με την σωστή εκτέλεση της διαδικασίας μέσα το σύστημα. Για να δούμε αυτές τις πληροφορίες ο χρήστης το μόνο που έχει να κάνει είναι μία κατάδειξη, δηλαδή απλά να αφήσει για κάποιο χρονικό διάστημα 1-2 δευτερολέπτων το δείκτη του ποντικού πάνω στο κουμπί όπου θέλει να μάθει κάποιες πληροφορίες σχετικά με την λειτουργία του συγκεκριμένου κουμπιού.

Στην παρακάτω εικόνα (Εικόνα 3.6.01) παρουσιάζεται μία λίστα με τα κουμπιά και τις περιγραφές τους, από την οποία μπορείτε να ενημερωθείτε για τη λειτουργία του κάθε κουμπιού στο σύστημα Clover Software:

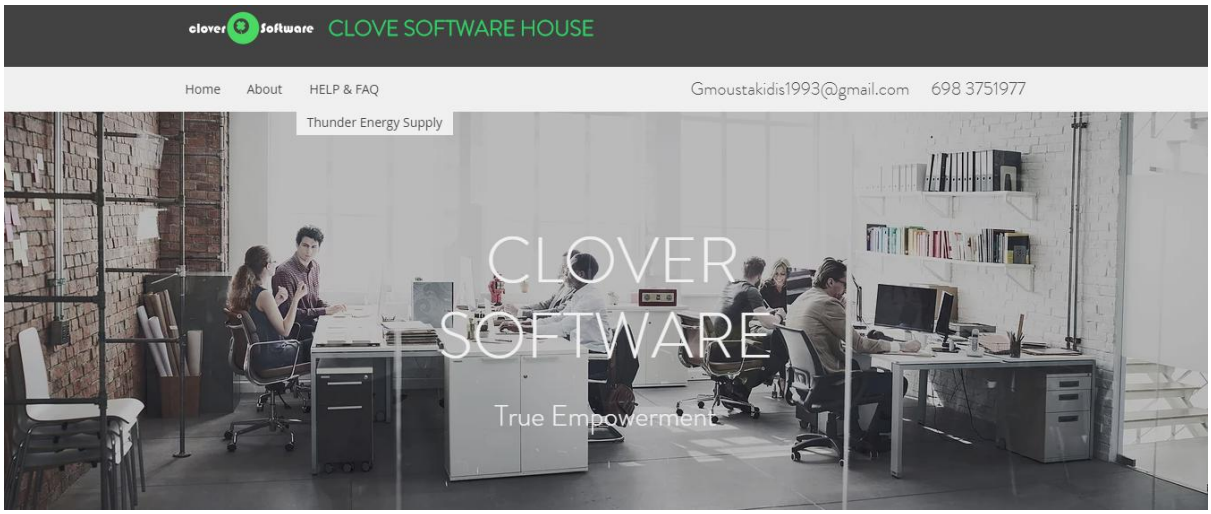


Εικόνα 3.6.01

3.7 Η ιστοσελίδα της εφαρμογής

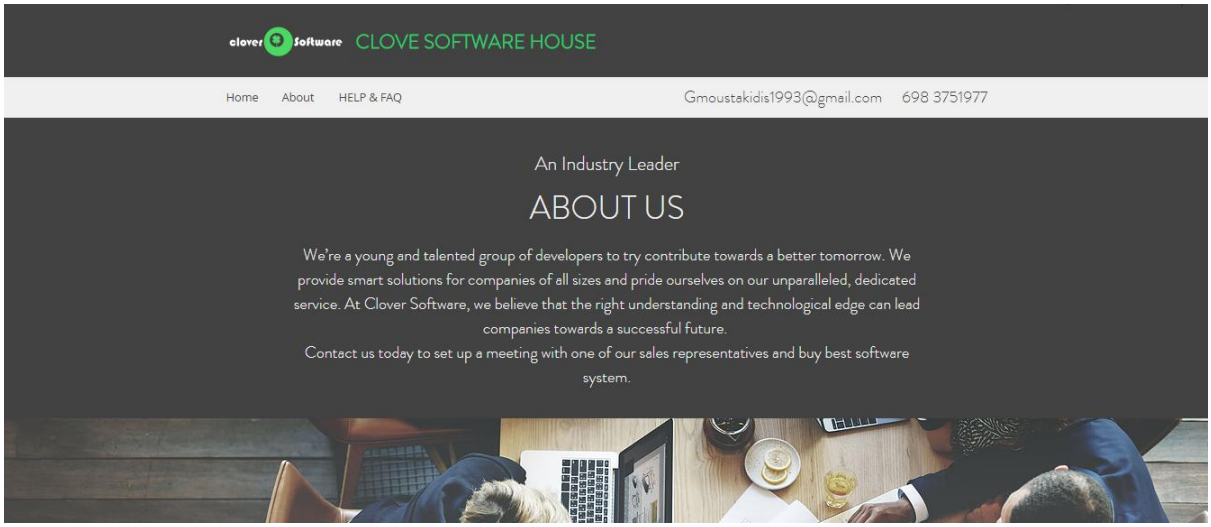
Προκείμενου να δώσουμε ζωή στο όλο σύστημα προσπαθήσαμε να σκεφτούμε πως θα μπορούσε να δικτυωθεί η εταιρία Clover Software. Ο ποιο απλός και λογικός τρόπος για την δικτύωση και εύρεση πελατών είναι η δημιουργία μια ιστοσελίδας όπου θα έχει λίγα λόγια για εκείνη αλλά και τρόπους εύκολης και άμεσης εξυπηρέτησης των πελατών της . Το link της ιστοσελίδας είναι https://gmoustakidis1993.wixsite.com/clover-software όπου μπορείτε να συνδεθείτε σε αυτό είτε σκανάροντας το Qr Code, είτε πατώντας Ctrl+H μέσα στο σύστημα της Clover software. Παράλληλα η ιστοσελίδα αυτή θα λειτουργούσε τόσο ως ένα εργαλείο διαφήμισης και προώθησης της εφαρμογής, αλλά ταυτόχρονα θα λειτουργούσε και σαν κόμβος υποστήριξης και παροχής βοήθειας για τη χρήση της εφαρμογής μέσω

κατάλληλων κειμένων και ενημερωτικών βίντεο για την ορθή και ολοκληρωμένη χρήση της εφαρμογής.

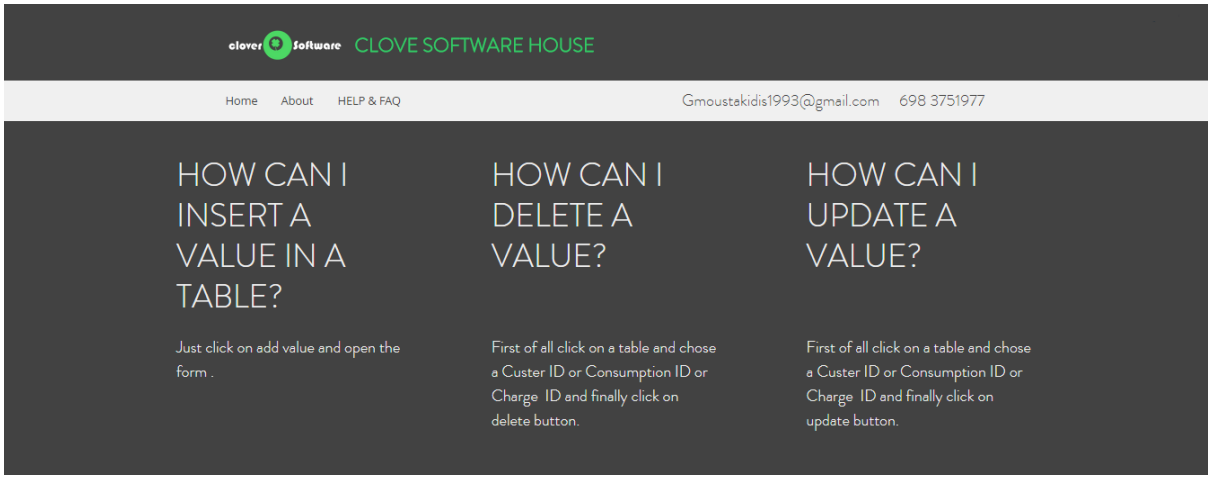


Εικόνα 3.7.1

Στην εικόνα 3.7.1 μπορούμε να παρατηρήσουμε την αρχική σελίδα της εταιρίας Clover Software ενώ στην εικόνα 3.7.2 ο χρήστης μπορεί να ενημερωθεί για το εταιρικό προφίλ της εταιρίας.



Εικόνα 3.7.2



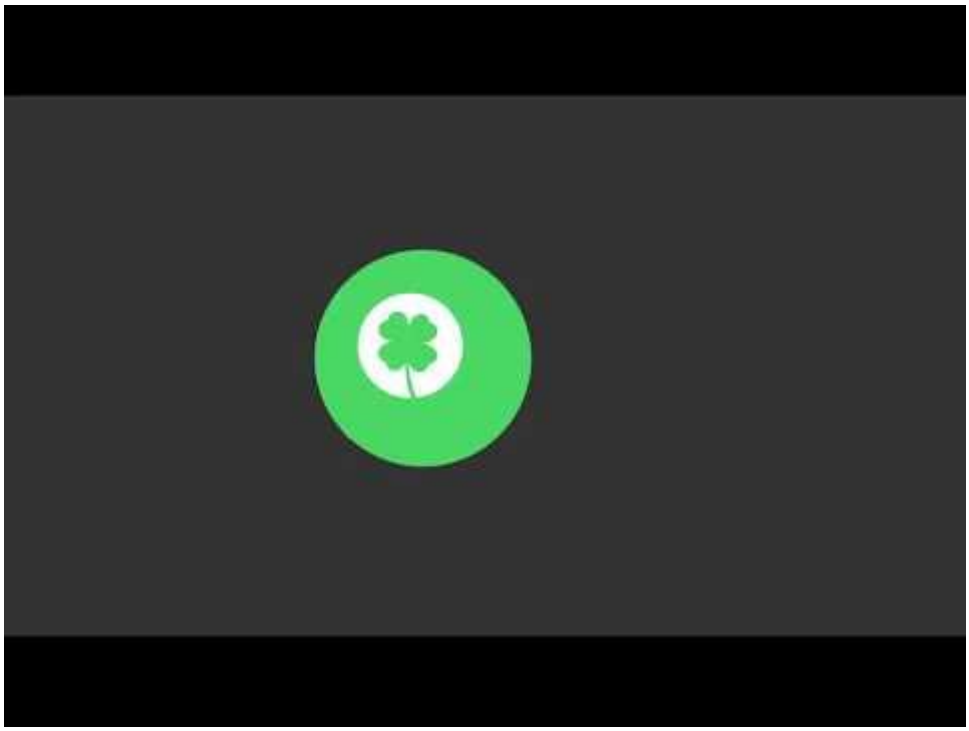
Εικόνα 3.7.3

Στην κατηγορία HELP & FAQ ο χρήστης μπορεί να βρει πληροφορίες για συχνές ερωτήσεις που γίνονται από πελάτες για την επίλυση ενός προβλήματος μέσα στο σύστημα (εικόνα 3.7.3). Επίσης υπάρχουν υποκατηγορίες με της εταιρίες που έχει αναλάβει η Clover Software. Σε κάθε μια εταιρία ο χρήστης μπορεί να βρει βιντεομαθήματα για την εκπαίδευση των εργαζομένων σε συχνές ενέργειες που θα εκτελεί μέσα στο σύστημα. Στην εικόνα 3.7.4 μπορείτε να δείτε το παράδειγμα της Thunder Energy Supply ενώ στην συνέχεια ακολουθούν δύο βιντεομαθήματα που μαθαίνουν στον χρήστη πως να κάνει τροποποίηση και διαγραφή δεδομένων μέσα στο σύστημα της Clover Software.


CLOVE SOFTWARE HOUSE
Home
About
HELP & FAQ
Welcome to Clover
Profits by Years
Software
clover
NEW STARTUP CHANGING THE GAME
TUTORIALS VIDEO
VIDEO
Update an item in clover software.
VIDEO
Delete an item in clover software
CLOVE SOFTWARE HOUSE
Address
Athens, Greece
Contact

Εικόνα 3.7.4

Διαγραφή ενός αντικειμένου μέσα στο σύστημα της Clover Software.



Τροποποίηση ενός αντικειμένου μέσα στο σύστημα της Clover Software.

Κεφάλαιο 4: Λίγα λόγια για τον κώδικα της εφαρμογής

4.1 Χρήση (κλήση) του Γραφικού Περιβάλλοντος QT5

Υπάρχουν αρκετοί τρόποι για να κληθεί και να χρησιμοποιηθεί ένα γραφικό περιβάλλον σε QT5 μέσα από την Python. Στο αρχικό σενάριο επιθυμούσαμε να «τρέχαμε» το γραφικό περιβάλλον TKinter ταυτόχρονα με το υπόλοιπο σύστημα. Η ονομασία και η χρήση των διάφορων γραφικών στοιχείων ελέγχου όμως ήταν πολύ ποιο περιπλοκή. Στην συνέχεια και υστέρα από αρκετή έρευνα ανακαλύψαμε ότι είναι δυνατόν να κληθεί το αρχείο με το γραφικό περιβάλλον σε QT5 και να διαχειριζόμαστε τα διάφορα στοιχεία και τις διευθύνσεις του με πολύ πιο απλό και εύκολο τρόπο.

```
from PyQt5.QtWidgets import QMainWindow, QApplication
from PyQt5 import uic
from PyQt5.QtWidgets import QTableWidget,QTableWidgetItem
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QMessageBox

Ui_MainWindow, QtBaseClass = uic.loadUiType("table.ui")
```

Εικόνα 4.1.01

Έτσι πρώτα καταχωρίσαμε τις βιβλιοθήκες που έχει η έκδοση PyQt 5 με ότι είναι απαραίτητο για την υλοποίηση και παρουσίαση του γραφικού περιβάλλοντος. Στην εικόνα 4.1.01 βλέπουμε κάποιες από τις σημαντικότερες βιβλιοθήκες που καταχωρήθηκαν σε μια από τις πρώτες προσπάθειες δημιουργίας αυτού του συστήματος.

4.2 Πως διαχειριζόμαστε προγραμματιστικά το πάτημα ενός κουμπιού;

Η διαδικασία διαχείρισης ενός συμβάντος όπως το πάτημα ενός κουμπιού είναι απλή. Όπως βλέπουμε στην εικόνα 4.2.01 πρώτα γράφουμε το όνομα του κουμπιού το όποιο σε αυτή την εικόνα έχει την ονομασία btn_area και στη συνέχεια γράφουμε (για το εάν γίνει cklick πάνω σε αυτό το κουμπί) να συνδεθεί και να εκτελεστεί μία συγκεκριμένη διαδικασία. Σε αυτή την περίπτωση η διαδικασία αυτή ονομάζεται DisplayDBDataAr (Εικόνα 4.2.02).

```
self.ui.btn_area.clicked.connect(self.DisplayDBDataAr)
```

Εικόνα 4.2.01

```
def DisplayDBDataAr(self):
  self.A_timi = self.ui.el_id.text()
  self.A_timi_2 = self.ui.el_name.text()
  self.Area_insertdata()
def Area_insertdata(self):
  lista = []
  lista_2 = []
  try:
    if self.A_timi=='':
      pass
    else:
      self.a_id = 'areaId'
      lista.append(self.a_id)
      lista_2.append(int(self.A_timi))
    if self.A_timi_2=='':
      pass
    else:
      self.a_name = 'areaName'
      lista.append(self.a_name)
      lista_2.append(self.A_timi_2)
    Area_leen  = (len(lista))
    if lista[0] =='areaName' :
      item = lista_2[0]
      new_item = item+'%'
      c.execute("select * from Area where areaName like '%s' ORDER BY areaId"% new_item)
      idata = c.fetchall()
```

Εικόνα 4.2.02

4.3 Πώς γίνεται η διασύνδεση με τη βάση δεδομένων;

Για να συνδεθούμε με τη βάση δεδομένων SQLite3 με το υπόλοιπο σύστημα αρχικά θα χρειαστεί να καλέσουμε τη βιβλιοθήκη SQLite της python.

```
import sqlite3
```

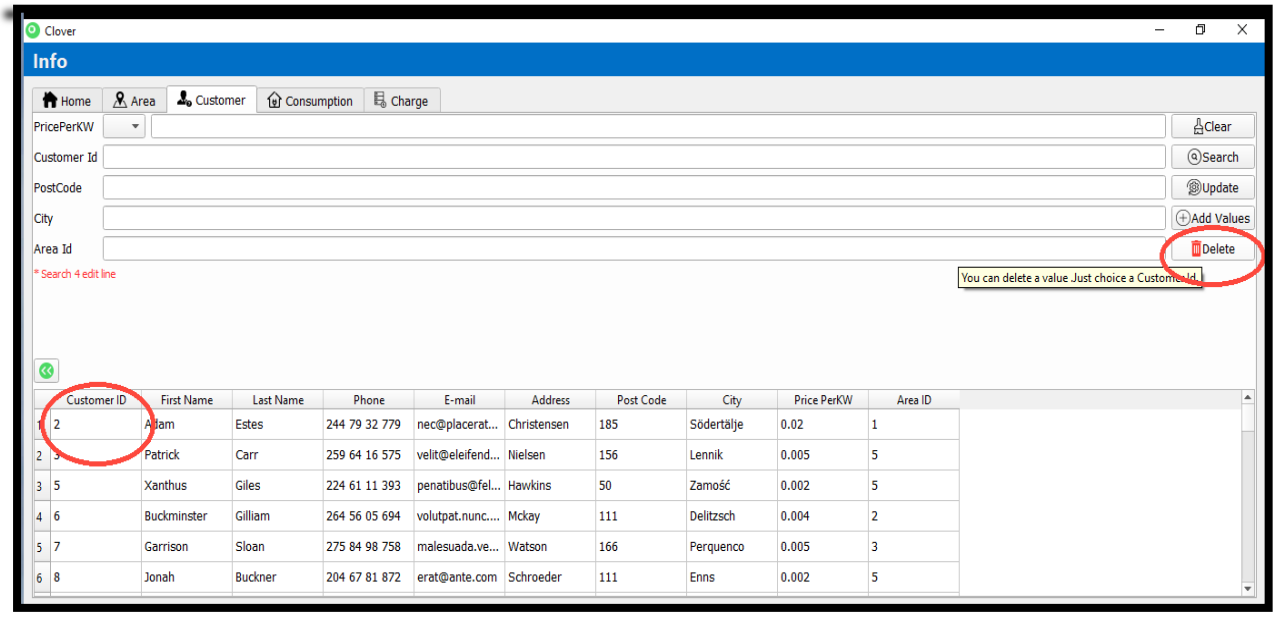
Στη συνέχεια καλούμε το αρχείο της βάσης δεδομένων μας (electricity.db). Τέλος ορίζουμε ένα αντικείμενο για το δρομέα (cursor C) που θα εκτελεί τις αναζητήσεις και τις διάφορεςεντολές σε SQL από το σύστημα στη βάση δεδομένων (Εικόνα 4.3.01).

```
db = sqlite3.connect('electricity.db')
c = db.cursor()
```

Εικόνα 4.3.01

4.3.1 Πως εκτελούμε ένα ερώτημα στη βάση δεδομένων;

Ας υποθέσουμε ότι προσπαθούμε να διαγράψουμε ένα στοιχείο από την καρτέλα του πίνακα Customer.



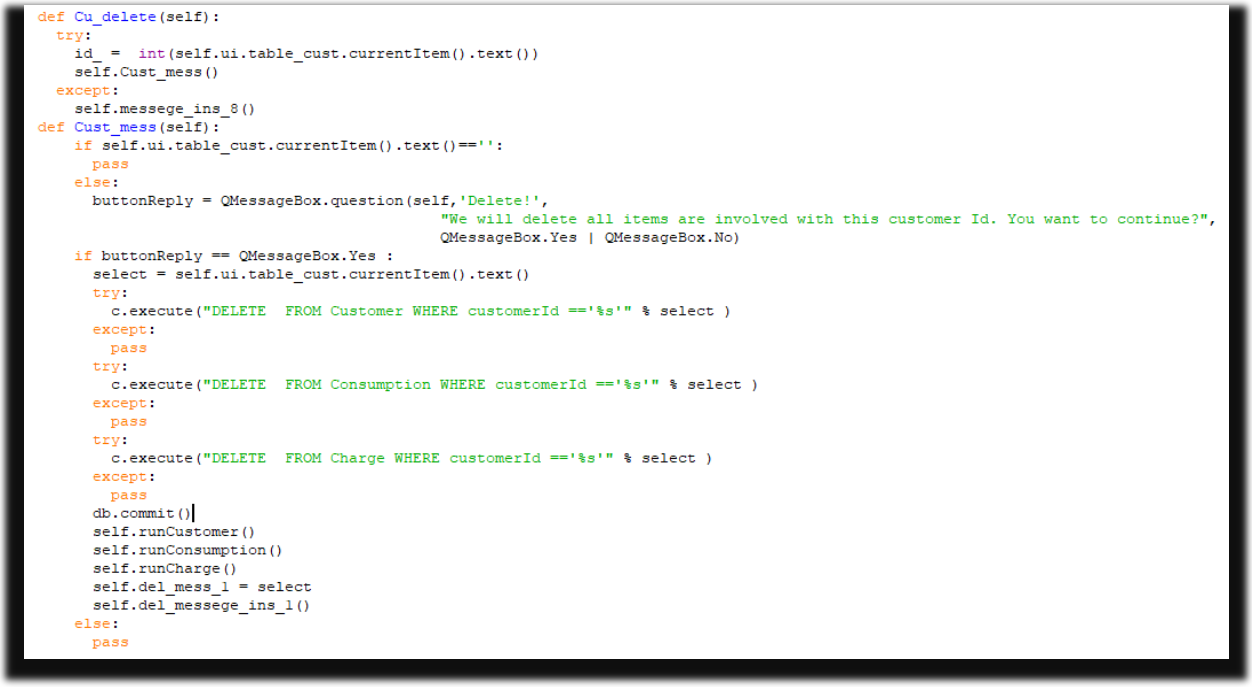
Μέσα από γραφικό περιβάλλον του συστήματος αυτό που έχει να κάνει ο χρήστης είναι να επιλέξει ένα συγκεκριμένο Customer Id μέσα από τον πίνακα. Τέλος πατώντας το κουμπί διαγραφής τότε αυτόματα η πληροφορία αυτή θα διαγραφεί από όλους τους πίνακες που εμπεριέχουν αυτή την πληροφορία. Τί γίνεται όμως εσωτερικά στον κώδικα; Πως υλοποιείται προγραμματιστικά αυτή η λειτουργία;

Όπως βλέπουμε στην εικόνα 4.3.02 όταν πατηθεί το κουμπί διαγράφης θα τότε θα εκτελεστεί η συνάρτηση Cu_delete.

```
self.ui.Cust_del_btn.clicked.connect(self.Cu_delete)
```

Εικόνα 4.3.02

Στην υλοποίηση της συνάρτησης βλέπουμε ότι χρησιμοποιείται το στοιχείο (Customer Id) μέσα από τον πίνακα και αυτό αποδίδεται στην μεταβλητή *id_*(Εικόνα 4.3.03). Σε περίπτωση λάθους θα εμφανιστεί ένα προειδοποιητικό μήνυμα (self.message_ins_8, Εικόνα 4.3.03). Ενώ σε περίπτωση επιτυχίας της διαδικασίας (δηλαδή ότι έχουμε επιλέξει έναν ακέραιο αριθμό) θα εκτελεστεί η ακόλουθη συνάρτηση *Cust_mess*.



```
def Cu_delete(self):
  try:
    id_ =  int(self.ui.table_cust.currentItem().text())
    self.Cust_mess()
  except:
    self.messege_ins_8()
def Cust_mess(self):
    if self.ui.table_cust.currentItem().text()=='':
      pass
    else:
      buttonReply = QMessageBox.question(self,'Delete!',
                                         "We will delete all items are involved with this customer Id. You want to continue?",
                                         QMessageBox.Yes | QMessageBox.No)
    if buttonReply == QMessageBox.Yes :
      select = self.ui.table_cust.currentItem().text()
      try:
        c.execute("DELETE  FROM Customer WHERE customerId =='%s'" % select )
      except:
        pass
      try:
        c.execute("DELETE  FROM Consumption WHERE customerId =='%s'" % select )
      except:
        pass
      try:
        c.execute("DELETE  FROM Charge WHERE customerId =='%s'" % select )
      except:
        pass
      db.commit()
      self.runCustomer()
      self.runConsumption()
      self.runCharge()
      self.del_mess_1 = select
      self.del_messege_ins_1()
    else:
      pass
```

Εικόνα 4.3.03

Στην συνέχεια θα παρουσιαστεί στο χρήστη ένα προειδοποιητικό μήνυμα. Το μήνυμα αυτό θα αναφέρει ότι θα διαγράφουν όλες οι αντίστοιχες εγγραφές μέσα στη βάση δεδομένων που σχετίζονται με το συγκεκριμένο πελάτη (Customer). Έπειτα και με τη ρητή συγκατάθεση του χρήστη, οι πληροφορίες αυτές θα διαγράφουν και από τους τρεις πίνακες (Customer, Consumption, Charge). Τέλος θα γίνει ανανέωση των πινάκων στο γραφικό περιβάλλον του συστήματος ενώ θα εμφανιστεί και ένα ενημερωτικό μήνυμα που θα αναφέρει τον πελάτη (Customer) που διαγράφηκε.

4.4 Πώς παρουσιάζονται τα στοιχεία στον πίνακα;

Η διαδικασία αυτή εκτελείτε σε κάθε tap του πίνακα και αποτελεί μια αυτόματη διαδικασία μέσα στο σύστημα καθώς ο κώδικας τρέχει στην αρχή της εκτέλεση του. Τα στοιχεία μέσα στον κώδικα προσαρμόζονται σύμφωνα με τις ανάγκες του κάθε tap. Σκοπός είναι ο χρήστης να έχει άμεση πρόσβαση στον πίνακα χωρείς να χρειάζεται να εκτελεί κάποια επιπλέον διαδικασία μέσα στο γραφικό περιβάλλον του συστήματος.

Προκειμένου να εισαχθούν τα στοιχεία μέσα στο πίνακα θα χρησιμοποιηθούν οι ακόλουθες γραμμές κώδικα (Εικόνα 4.4.01).

```
def runArea(self):
    invoicesSql = "select * from Area"
    c.execute(invoicesSql)
    idata = c.fetchall()

    rows = len(idata)
    columns = len(idata[0])
    self.ui.table.setRowCount(rows)
    self.ui.table.setColumnCount(columns)
```

Εικόνα 4.4.01

Για να επιλέγουν όλα τα στοιχεία από τη βάση δεδομένων, βρίσκουμε το πλήθος τον στηλών (Columns) και των γραμμών (Rows) μέσα στο σύστημα. Έπειτα δημιουργείται το αντίστοιχο πλήθος γραμμών και στηλών μέσα στο πίνακα του γραφικού περιβάλλοντος.

```
self.ui.table.setHorizontalHeaderLabels(['Area ID', 'Area Name' ])
for i, row in enumerate(idata):
  for j, col in enumerate(row):
    item = QTableWidgetItem(str(col))
    self.ui.table.setItem(i, j, item)
```

Εικόνα 4.4.02

Αφού δοθεί μία ονομασία για την κάθε στήλη του πίνακα, στην συνέχεια τα στοιχεία αυτά τοποθετούνται ένα-ένα μέσα στον πίνακα (Εικόνα 4.4.02).

Κεφάλαιο 5: Συμπεράσματα & Επεκτάσεις

5.1 Συμπεράσματα

Σαν ένα γενικότερο συμπέρασμα με την ενασχόληση με αυτή τη συγκεκριμένη πτυχιακή εργασία είναι πως ο πλήρης κύκλος ζωής ενός έργου πληροφορικής, δηλαδή η ανάλυση, ο σχεδιασμός, η υλοποίηση και η υποστήριξη μιας εφαρμογής λογισμικού είναι κάτι δημιουργικό και σου αφήνει ένα αίσθημα ικανοποίησης για την κατάκτησης της γνώσης και επίλυσης κάτι σύνθετου μέσω ενός ηλεκτρονικού υπολογιστή. Η ενασχόληση με ένα σύνθετο έργο πληροφορικής σε όλες της φάσης διαχείρισης του έργου ήταν από μόνης μία διαδικασία τριβής με διαφορετικά επιστημονικά αντικείμενα και δόθηκε η ευκαιρία, ίσως για πρώτη φορά, να δούμε πως αυτές οι διαφορετικές επιστημονικές περιοχές, η κάθε μία από μόνη της και όλες μαζί συνεκτικά μπορούν να λειτουργήσουν στο να δημιουργηθεί μία ολοκληρωμένη λύση εφαρμογής λογισμικού.

Έτσι, διαπιστώθηκε πως η ανάλυση και η σχεδίαση πληροφοριακών συστημάτων μας βοήθησε στο να εκμαιεύσουμε τις βασικές λειτουργίες της εφαρμογή και η επιλογή των βασικών οντοτήτων και η σχεδίαση του σχεσιακού σχήματος της βάσης δεδομένων έδωσε τη δυνατότητα να ανακαλύψουμε και να σχεδιάσουμε καλύτερα τα δεδομένα σας μέσα σε μία βάση δεδομένων. Ακόμη χρησιμοποιήσαμε τεχνικές του προγραμματιστικού παραδείγματος του αντικειμενοστραφούς προγραμματισμού ώστε να καταφέρουμε να συνδεθούμε με τη βάση δεδομένων που σχεδιάσαμε καθώς και να συγγράψουμε και να εκτελέσουμε απλά και σύνθετα ερωτήματα στη γλώσσα SQL για να αναζητήσουμε και να υπολογίσουμε στοιχεία από τα δεδομένα που φιλοξενεί η βάση των δεδομένων μας. Είχαμε τη δυνατότητα να δούμε στην πράξη τη χρήση του αντικειμενοστραφούς προγραμματισμού μέσω της σχεδίασης του γραφικού περιβάλλοντος διεπαφής της εφαρμογής με τη χρήση του συστήματος QT5 καθώς και να δούμε στην πράξη τη διασύνδεση και το χειρισμό των συμβάντων μέσα στον κώδικα σε γλώσσα python. Τέλος είχαμε την ευκαιρία να σχεδιάσουμε και να αναπτύξουμε έναν όμορφο και λειτουργικό ιστοχώρο (website) που να διαφημίζει και να υποστηρίζει την εφαρμογή λογισμικού που αναπτύχθηκε.

5.2 Μελλοντικές Επεκτάσεις

Αυτή η εφαρμογή λογισμικού που αναπτύχθηκε στα πλαίσια της πτυχιακής εργασίας και για αυτό το λόγο ο διαθέσιμος χρόνος για την υλοποίησή της ήταν σχετικά περιορισμένος,

παρόλα αυτά η εφαρμογή που σχεδιάστηκε και υλοποιήθηκε είναι μία πλήρως λειτουργική εφαρμογή για τις λειτουργίες που επιλέξαμε να υλοποιήσουμε. Με αυτό το σκεπτικό θα μπορούσε αυτή εφαρμογή να επεκταθεί και σε μία επόμενη ή/και επόμενες εκδόσεις να σχεδιαστεί και γραφεί ο κατάλληλος κώδικας ώστε να υποστηρίζονται μία σειρά από λειτουργίες όπως:

- Επέκταση της βάσης δεδομένων και δημιουργία ανάλογων στοιχείων καρτελών της φόρμας ώστε να υποστηρίζονται πληρωμές (τμηματικές/έναντι και εκκαθαριστικές) για τις χρεώσεις της κατανάλωσης της ηλεκτρικής ενέργειας.
- Η δυνατότητα δημιουργίας ενημερωτικών εγγράφων-λογαριασμών των χρεώσεων και η αυτοματοποιημένη αποστολή τους μέσω ηλεκτρονικού ταχυδρομείου (email) στους πελάτες.
- Αυτοματοποιημένη διαδικασία δημιουργίας αντιγράφου ασφαλείας και επαναφοράς αντιγράφου (backup/restore) της βάσης δεδομένων.
- Επανασχεδιασμός του κώδικα της σύνδεσης με τη βάση δεδομένων και η χρήση τεχνολογιών ORM (Object-relational mapping) ώστε να γίνεται χρήση αντικειμένων για τους αντίστοιχους πίνακες της βάσης δεδομένων, καθώς και η ενδεχόμενη χρήση εκφράσεων lambda αντί της χρήσης (πολύπλοκου) κώδικα σε γλώσσα SQL.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Α. ΞΕΝΟΓΛΩΣΣΗ

QT

Riverbank Computing (2016). *Using Qt Designer*. Διαθέσιμο στο: http://pyqt.sourceforge.net/Docs/PyQt4/designer.html [Πρόσβαση: 25 Φεβρουάριου 2018]

Riverbank Computing (2017). *Using Qt Designer*. Διαθέσιμο στο: http://pyqt.sourceforge.net/Docs/PyQt5/designer.html [Πρόσβαση: 25 Φεβρουάριου 2018]

The Qt Company (2018). *Qt Designer Manual*. Διαθέσιμο στο: http://doc.qt.io/qt-5/qtdesigner-manual.html [Πρόσβαση: 25 Φεβρουάριου 2018]

Wikipedia (2016). *XML User Interface*. Διαθέσιμο στο: https://en.wikipedia.org/wiki/XML_User_Interface [Πρόσβαση: 25 Φεβρουάριου 2018]

Wikipedia (2018). *Qt Creator*. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Qt_Creator [Πρόσβαση: 25 Φεβρουάριου 2018]

Wikipedia (2018). *Qt: software*. Διαθέσιμο στο: https://en.wikipedia.org/wiki/Qt_(software) [Πρόσβαση: 25 Φεβρουάριου 2018]

Β. ΕΛΛΗΝΙΚΗ

Αντικειμενοστεφης Προγραμματισμός

Βακαλή, Α., Γιαννόπουλος, Η., Ιωαννίδης, Ν., Κοιλιάς, Χ., Μάλαμας, Κ., Μανωλόπουλος, Ι. και Πολίτης, Π. (2010) *Ανάπτυξη Εφαρμογώ σε: Προγραμματιστικό Περιβάλλον*. Αθήνα: ΕΠΥ. Διαθέσιμο στο: http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C101/36/198,1055/ [Πρόσβαση: 25 Φεβρουάριου 2018]

Βικιπαίδια (2016). *Προγραμματιστικό παράδειγμα.* Διαθέσιμο στο: https://el.wikipedia.org/wiki/%CE%A0%CF%81%CE%BF%CE%B3%CF%81%CE%B1%CE%BC%CE%BC%CE%B1%CF%84%CE%B9%CF%83%CF%84%CE%B9%CE%BA%CF%8C_%CF%80%CE%B1%CF%81%CE%AC%CE%B4%CE%B5%CE%B9%CE%B3%CE%BC%CE%B1 [Πρόσβαση: 25 Φεβρουάριου 2018]

Βικιπαίδια (2016). *Αντικειμενοστρεφής προγραμματισμός.* Διαθέσιμο στο: http://hermes.di.uoa.gr/exe_activities/algorithmoi/_11.html [Πρόσβαση: 25 Φεβρουάριου 2018]

Python Tutorial (2012/ Revision c82074b6775b+). Αντικειμενοστραφής Προγραμματισμός: Object-Oriented Programming. Διαθέσιμό στο: http://python-tutorial-greek.readthedocs.io/en/latest/oop_general.html#object-oriented-programming [Πρόσβαση: 25 Φεβρουάριου 2018]

Python

Αγγελιδάκης, Ν. Α. (2015) *Εισαγωγή στον προγραμματισμό με την Python*. Διαθέσιμο στο: http://aggelid.mysch.gr/pythonbook/INTRODUCTION_TO_COMPUTER_PROGRAMMING_WITH_PYTHON.pdf [Πρόσβαση: 25 Φεβρουάριου 2018]

Downey, A. (2014) *Think Python: How to think Like a Computer Scientist*. Διαθέσιμο στο: http://www.teilar.gr/dbData/ProfAnn/profann-700d1320.pdf [Πρόσβαση: 25 Φεβρουάριου 2018]

Cyber Python. *Byte of Python: Ελληνική Έκδοση*. Διαθέσιμο στο: https://cyberpython.github.io/byte-of-python/introduction.html [Πρόσβαση: 25 Φεβρουάριου 2018]

QT

Κιορπέ, Π. (2013) *MPSMaker: Ανάπτυξη διεπαφής για την επεξεργασία μετραπτογραμμάτων και σύνδεση τους με γνωστούς λύτες*. Διαθέσιμο στο: https://dspace.lib.uom.gr/bitstream/2159/16154/6/KiorpesPeriklisMsc2013.pdf [Πρόσβαση: 25 Φεβρουάριου 2018]

SQL

Βικιπαίδια (2017). *SQL*. Διαθέσιμο στο: https://el.wikipedia.org/wiki/SQL [Πρόσβαση: 25 Φεβρουάριου 2018]

Ip.gr. *Τι είναι SQL: Structured Query Language*. Διαθέσιμο στο: https://www.ip.gr/el/dictionary/144-SQL___Structured_Query_Language [Πρόσβαση: 25 Φεβρουάριου 2018]

Μπαγκέρη, Ε. *Σχεσιακές Βάσεις Δεδομένων: Η γλώσσα SQL και Αποθηκευμένες Διαδικασίες*. Διαθέσιμο στο: http://nefeli.lib.teicrete.gr/browse/stef/epp/2010/MpagkeriEvangelia/attached-document-1271831186-641028-6701/Mpaggeri2010.pdf [Πρόσβαση: 25 Φεβρουάριου 2018]

SQLite

Βικιπαίδια (2017). *SQLite*. Διαθέσιμο στο: https://el.wikipedia.org/wiki/SQLite [Πρόσβαση: 25 Φεβρουάριου 2018]

Λουκόπουλος, Α., και Θεοδωρίδης, Ε. (2015) *Ανάπτυξη Εφαρμογών σε: Προγραμματιστικό Περιβάλλον*. Αθήνα: ΣΕΑΒ. Διαθέσιμο στο:

http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C101/36/198,1055/ [Πρόσβαση: 25 Φεβρουάριου 2018]

Γεωργίου, Μ. (2011) *Μοντέλα Βάσεων Δεδομένων*. Διαθέσιμο στο: https://www.slideshare.net/marygeorg/2-14787550?next_slideshow=1 [Πρόσβαση: 25 Φεβρουάριου 2018]

Microsoft (2018). *Βασικές πληροφορίες για τις βάσεις δεδομένων*. Διαθέσιμο στο: https://support.office.com/el-gr/article/%CE%92%CE%B1%CF%83%CE%B9%CE%BA%CE%AD%CF%82-%CF%80%CE%BB%CE%B7%CF%81%CE%BF%CF%86%CE%BF%CF%81%CE%AF%CE%B5%CF%82-%CE%B3%CE%B9%CE%B1-%CF%84%CE%B9%CF%82-%CE%B2%CE%AC%CF%83%CE%B5%CE%B9%CF%82-%CE%B4%CE%B5%CE%B4%CE%BF%CE%BC%CE%AD%CE%BD%CF%89%CE%BD-a849ac16-07c7-4a31-9948-3c8c94a7c204 [Πρόσβαση: 25 Φεβρουάριου 2018]

Ξένος, Μ., και Χριστοδουλάκης, Δ. (2000) *Βάσεις Δεδομένων*. Πάτρα: TYPORAMA. Διαθέσιμο στο: http://www.jimkava.com/wp-content/uploads/2017/08/xristodoulakis.pdf [Πρόσβαση: 27 Φεβρουάριου 2018]

ΠΑΡΑΡΤΗΜΑ Α

Ο κώδικας σε γλώσσα python της εφαρμογής λογισμικού clover software

```
import sqlite3
import sys
from PyQt5.QtWidgets import QMainWindow, QApplication
from PyQt5 import uic
from PyQt5.QtWidgets import QTableWidget,QTableWidgetItem
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QMessageBox
import webbrowser
import sqlite3
import datetime
from datetime import date, datetime
import matplotlib.pyplot as plt
##import numpy as np
##import matplotlib.dates as mdates
import numpy as np
from dateutil import parser
import matplotlib.patches as mpatches
from matplotlib import style
style.use('ggplot')
date =date.today()
from datetime import date, datetime
import matplotlib.pyplot as plt
import xlsxwriter
from xlrd import open_workbook
from datetime import datetime
from dateutil.parser import parse
import pandas as pd
import xlrd
import xlsxwriter
import datetime, xlrd
Ui_MainWindow, QtBaseClass = uic.loadUiType("table.ui")
db = sqlite3.connect('electricity.db')
c = db.cursor()
class MyApp(QMainWindow):
  def __init__(self):
    super(MyApp, self).__init__()
    self.ui = Ui_MainWindow()
    self.ui.setupUi(self)

    self.showMaximized()
    self.area_comboBox()

    self.ui.actionExit.triggered.connect(self.exit_Action)
    self.ui.actionHelp.triggered.connect(self.link)
```

```
self.ui.actionFullScreen.triggered.connect(self.FullScreen)
self.ui.actionUnique_items.triggered.connect(self.about_unq)
self.ui.actionSystem.triggered.connect(self.about)
self.ui.actionShearch.triggered.connect(self.about_search)
self.ui.cha_dwl.clicked.connect(self.Action_export)
self.ui.label_54.mouseReleaseEvent=self.showGraph
self.ui.btn_area.clicked.connect(self.DisplayDBDataAr)
self.ui.btn_cust.clicked.connect(self.DisplayDBDataCust)
self.ui.btn_con.clicked.connect(self.DisplayDBDataCon)
self.ui.btn_cha.clicked.connect(self.DisplayDBDataCha)
self.ui.btn_sum.clicked.connect(self.Total)
self.ui.btn_h_gr_2.clicked.connect(self.graph_home_1)
self.ui.btn_h_gr.clicked.connect(self.graph_home_2)
self.ui.widget.hide()
self.ui.widget_2.hide()
self.ui.widget_3.hide()
self.ui.widget_4.hide()
self.ui.widget_5.hide()
self.ui.widget_6.hide()
self.ui.widget_7.hide()
self.ui.widget_8.hide()
self.ui.btn_graph.clicked.connect(self.graph)
self.ui.btn_graph_2.clicked.connect(self.graph_2)
self.ui.back_1.clicked.connect(self.runArea)
self.ui.back_2.clicked.connect(self.runCustomer)
self.ui.back_3.clicked.connect(self.runConsumption)
self.ui.back_4.clicked.connect(self.runCharge)
self.ui.Cust_del_btn.clicked.connect(self.Cu_delete)
self.ui.Con_del_btn.clicked.connect(self.Co_delete)
self.ui.Cha_del_btn.clicked.connect(self.Cha_delete)
self.ui.co_btn_up.clicked.connect(self.Con_update)
self.ui.a_btn_up.clicked.connect(self.area)
self.ui.cha_btn_up.clicked.connect(self.Cha_update)
self.ui.cu_btn_up.clicked.connect(self.Cust_update)
self.ui.A_sent_btn.clicked.connect(self.ar_ins)
self.ui.Cust_sent_btn.clicked.connect(self.insert)
self.ui.Con_sent_btn.clicked.connect(self.Con_insert)
self.ui.Cha_sent_btn.clicked.connect(self.Cha_insert)
self.ui.A_add_btn.setToolTip('Add item to Area table.')

self.ui.Cust_del_btn.setToolTip('Delete a value .Just choice a Customer Id.')
self.ui.Cu_add_btn.setToolTip('Add item to Customer table.')

self.ui.Con_del_btn.setToolTip('Delete a value .Just choice a Consumption Id.')
self.ui.Con_add_btn.setToolTip('Add item to Consumption table.')

self.ui.Cha_del_btn.setToolTip('Delete a value .Just choice a Charge Id.')
```

```
    self.ui.Cha_add_btn.setToolTip('Add item to Charge table.')

    self.ui.up_ar_btn.setToolTip('First step for value update, edit the editline and write the new value.Next step choose an integer value from  AreaId.')
    self.ui.cust_btn_update.setToolTip('First step for value update, edit the editline and write the new value.Next step choose an integer value from  CustomerId.')
    self.ui.up_btn_con.setToolTip('First step for value update, edit the editline and write the new value.Next step choose an integer value from ConsumptionId.')
    self.ui.up_btn_cha.setToolTip('First step for value update, edit the editline and write the new value.Next step choose an integer value from  ChargeId.')

    self.ui.back_1.setToolTip('Refresh an Area Table.')
    self.ui.back_2.setToolTip('Refresh a Customer Table.')
    self.ui.back_3.setToolTip('Refresh a Consumption Table.')
    self.ui.back_4.setToolTip('Refresh a Charge Table.')

    self.ui.btn_graph.setToolTip('Choice a customer id and click graph button and will show you the result of Consumption Table.')
    self.ui.btn_graph_2.setToolTip('Choice a customer id and click graph button and will show you the result of Charge Table.')


    self.ui.btn_h_gr.setToolTip('Write a customer id and click to graph button to show a result of consumption.')
    self.ui.btn_h_gr_2.setToolTip('Write the Year and click graph button to show a result of the Year.')

    self.ui.label_54.setToolTip('Click and show more')

    self.ui.radioButton_5.setToolTip('Will  take  the  last  Consumption  Id  from  selected customer')

    self.ui.btn_sum.setToolTip('Display total values')
    self.ui.cha_dwl.setToolTip('For a customer export excel file')
    #auto run btable
    self.runArea()
    self.runCustomer()
    self.runConsumption()
    self.runCharge()
  def export(self):
    try:
      select_id = self.ui.table_cha.currentItem().text()
      select_id = int(select_id)
      tp = type(select_id)
      if tp == int:
        c.execute("SELECT * FROM Charge WHERE customerId == %s "% select_id)
        lista = []
```

```
        count = -1

        self.sentID = select_id
        workbook = xlsxwriter.Workbook("charge_customer_{0}.xlsx".format(select_id))
        worksheet1 = workbook.add_worksheet()
        for i in c.fetchall():

            lista.append(i)
        for q,i in enumerate(lista):

            chaId = i[0]
            chaDate = i[1]
            KW = float(i[2])
            totalPrice = float(i[3])
            CustId = i[4]
            cnt = q+1
            worksheet1.write(0,0,"Charge Id")
            worksheet1.write(0,1,"Charge Date")
            worksheet1.write(0,2,"KW")
            worksheet1.write(0,3,"Total Price")
            worksheet1.write(0,4,"Customer Id")

            worksheet1.write(cnt, 0,chaId)
            worksheet1.write(cnt, 1,chaDate)
            worksheet1.write(cnt, 2,float(KW))
            worksheet1.write(cnt, 3,float(totalPrice))
            worksheet1.write(cnt, 4,CustId)
        workbook.close()
        self.messege_ins_14()
    except:
      self.messege_ins_13()




  def Total(self):
    try:
      select_id = self.ui.table_cha.currentItem().text()

      c.execute("""SELECT SUM(KW) FROM Charge WHERE customerId =
%s"""%select_id)
      item = c.fetchone()
      timi = str(item[0])
      self.ui.label_60.setText(timi)
```

```
      c.execute("""SELECT SUM(totalPrice) FROM Charge WHERE customerId =
%s"""%select_id)
      item = c.fetchone()
      timi = str(item[0])
      self.ui.label_62.setText(timi)
    except:
      pass



  def FullScreen(self):
    self.showFullScreen()

  def showGraph(self,event):
    plt.figure(figsize=(12,6),num='Consumption Graph')
    dates = []
    values = []
    for i in range(2000,2019):
        select_id = i
        c.execute("""SELECT SUM(totalKW) FROM Consumption WHERE
SUBSTR(endDate,1,4)="%.2i" """ % int(select_id))
        row = c.fetchone()
        values.append(row)
        dates.append(i)

    plt.plot(dates,values,'o:',color='#0078d7')
    plt.title(" Profits by Years")
    plt.xlabel('Dates')
    plt.ylabel('Total KW')
    ##plt.xticks(rotation=35)
    plt.show()
  def graph_home_1(self):
    try:
      select_id_1 = self.ui.lineEdit_27.text()
      select_id_2 = self.ui.lineEdit_28.text()
      select_id_3 = self.ui.lineEdit_32.text()

      home_list = []

      if select_id_1 == '':
        pass
      else:
        home_list.append(select_id_1)

      if select_id_2 == '':
        pass
      else:
```

```
        home_list.append(select_id_2)

      if select_id_3 == '':
        pass
      else:
        home_list.append(select_id_3)


      home_leen  = (len(home_list))
      if home_leen == 1:
        select_id = home_list[0]
        c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId == %s
"% select_id)
        data = c.fetchall()
        dates = []
        values = []
        for row in data:
            dates.append(parser.parse(row[0]))
            values.append(row[1])
            dates.sort()
        name_cust_1 = ("Customer Id : %s"%select_id)

        plt.plot(dates,values,'o:',color='#ff3333')
        red_patch = mpatches.Patch(color='#ff3333', label=name_cust_1)
        plt.legend(handles=[red_patch])

        plt.title("Consumption for Customer: %s "% select_id)
        plt.xlabel('Dates')
        plt.ylabel('Total KW')
        plt.xticks(rotation=35)
        plt.show()

      elif home_leen == 2:
        select_id = home_list[0]
        select_id_2 = home_list[1]
        c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId == %s
"% select_id)
        data = c.fetchall()
        dates = []
        values = []
        for row in data:
            dates.append(parser.parse(row[0]))
            values.append(row[1])
            dates.sort()
        c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId == %s
"% select_id_2)
        data_2 = c.fetchall()
```

```
        dates_2 = []
        values_2 = []
        for row in data_2:
            dates_2.append(parser.parse(row[0]))
            values_2.append(row[1])
            dates_2.sort()
        name_cust_1 = ("Customer Id : %s "%select_id)
        name_cust_2 = ("Customer Id : %s "%select_id_2)


        plt.plot(dates,values,'-bs',color='#ff3333')
        red_patch = mpatches.Patch(color='#ff3333', label=name_cust_1)
        plt.plot(dates_2,values_2,'o:',color='#0078d7')
        blue_patch = mpatches.Patch(color='#0078d7', label=name_cust_2)

        plt.legend(handles=[red_patch,blue_patch])

        plt.title("Consumption for Customers: %s - %s"% (select_id,select_id_2))
        plt.xlabel('Dates')
        plt.ylabel('Total KW')
        plt.xticks(rotation=35)
        plt.show()

      elif home_leen == 3:

        select_id = home_list[0]
        select_id_2 = home_list[1]
        select_id_3 = home_list[2]

        c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId == %s
"% select_id)
        data = c.fetchall()
        dates = []
        values = []
        for row in data:
            dates.append(parser.parse(row[0]))
            values.append(row[1])
            dates.sort()
        c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId == %s
"% select_id_2)
        data_2 = c.fetchall()
        dates_2 = []
        values_2 = []
        for row in data_2:
            dates_2.append(parser.parse(row[0]))
            values_2.append(row[1])
            dates_2.sort()
```

```
        c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId == %s
"% select_id_3)
        data_3 = c.fetchall()
        dates_3 = []
        values_3 = []
        for row in data_3:
            dates_3.append(parser.parse(row[0]))
            values_3.append(row[1])
            dates_3.sort()



        name_cust_1 = ("Customer Id : %s "%select_id)
        name_cust_2 = ("Customer Id : %s "%select_id_2)
        name_cust_3 = ("Customer Id : %s "%select_id_3)

        plt.plot(dates,values,'o:',color='#ff3333')
        red_patch = mpatches.Patch(color='#ff3333', label=name_cust_1)
        plt.plot(dates_2,values_2,'-bs',color='#0078d7')
        blue_patch = mpatches.Patch(color='#0078d7', label=name_cust_2)
        plt.plot(dates_3,values_3,'--g^',color='#33cc66')
        green_patch = mpatches.Patch(color='#33cc66', label=name_cust_3)

        plt.legend(handles=[red_patch,blue_patch,green_patch])

        plt.title("Consumption for Customers : %s - %s - %s "%
(select_id,select_id_2,select_id_3))
        plt.xlabel('Dates')
        plt.ylabel('Total KW')
        plt.xticks(rotation=35)
        plt.show()
      else:
        self.messege_ins_8()

    except:
      self.messege_ins_8()



  def graph_home_2(self):
    try:
      select_id = self.ui.lineEdit_33.text()
      plt.figure(figsize=(12,6),num='Consumption Graph')
      c.execute("""SELECT endDate,totalKW FROM Consumption WHERE
SUBSTR(endDate,1,4)="%.2i" """ % int(select_id))
```

```
    ##c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId ==
%s "% select_id)
    data = c.fetchall()
    dates = []
    values = []
    for row in data:
       dates.append(parser.parse(row[0]))
       values.append(row[1])
       dates.sort()
    plt.plot_date(dates,values,'o:',color='#0078d7')
    plt.title("Result in Year: %s "% select_id)
    plt.xlabel('Dates')
    plt.ylabel('Total KW')
    plt.xticks(rotation=35)
    plt.show()
  except:
    self.messege_ins_8()




 def graph(self):
  try:
    plt.figure(figsize=(12,6),num='Consumption Graph')
    select_id = self.ui.table_con.currentItem().text()
    c.execute("SELECT endDate,totalKW FROM Consumption WHERE customerId == %s
"% select_id)
    data = c.fetchall()
    dates = []
    values = []
    for row in data:
       dates.append(parser.parse(row[0]))
       values.append(row[1])
       dates.sort()
    plt.plot_date(dates,values,'o:',color='#0078d7')
    plt.title("Consumption in CustomerId: %s "% select_id)
    plt.xlabel('Dates')
    plt.ylabel('Total KW')
    plt.xticks(rotation=35)
    plt.show()
  except:
    pass

 def graph_2(self):
  try:
    plt.figure(figsize=(12,6),num='Charge Graph')
    select_id = self.ui.table_cha.currentItem().text()
```

```
    c.execute("SELECT chargeDate,KW FROM Charge WHERE customerId == %s "% select_id)
    data = c.fetchall()
    dates = []
    values = []
    for row in data:
        dates.append(parser.parse(row[0]))
        values.append(row[1])
        dates.sort()
    plt.plot_date(dates,values,'o:',color='#0078d7')
    plt.title("Charge in CustomerId: %s "% select_id)
    plt.xlabel('Dates')
    plt.ylabel(' KW')
    plt.xticks(rotation=35)
    plt.show()
  except:
    pass




 def area_comboBox(self):
  lista_a = []
  c.execute("SELECT areaName FROM Area ")
  for row in c.fetchall():
   lista_a.append(row[0])
  self.ui.comboBox.addItems(lista_a)




 def DisplayHelloWord(self):
  self.ui.outputTextBox.setText("Area")
 def runArea(self):
    invoicesSql = "select * from Area"
    c.execute(invoicesSql)
    idata = c.fetchall()

    rows = len(idata)
    columns = len(idata[0])

    self.ui.table.setRowCount(rows)
    self.ui.table.setColumnCount(columns)

    self.ui.table.setHorizontalHeaderLabels(['Area ID', 'Area Name' ])
    for i, row in enumerate(idata):
```

```
        for j, col in enumerate(row):
          item = QTableWidgetItem(str(col))
          self.ui.table.setItem(i, j, item)
  def DisplayDBDataAr(self):
    self.A_timi = self.ui.el_id.text()
    self.A_timi_2 = self.ui.el_name.text()
    self.Area_insertdata()
  def Area_insertdata(self):

    lista = []
    lista_2 = []
    try:
      if self.A_timi==":
        pass
      else:
        self.a_id = 'areaId'
        lista.append(self.a_id)
        lista_2.append(int(self.A_timi))



      if self.A_timi_2==":
        pass
      else:
        self.a_name = 'areaName'
        lista.append(self.a_name)
        lista_2.append(self.A_timi_2)

      Area_leen  = (len(lista))
      if lista[0] =='areaName' :


        item = lista_2[0]
        new_item = item+'%'
        c.execute("select * from Area where areaName like '%s' ORDER BY areaId"% new_item)

        idata = c.fetchall()

        rows = len(idata)
        columns = len(idata[0])

        self.ui.table.setRowCount(rows)
        self.ui.table.setColumnCount(columns)

        self.ui.table.setHorizontalHeaderLabels(['Area ID', 'Area Name'])
        for i, row in enumerate(idata):
```

```
        for j, col in enumerate(row):
          item = QTableWidgetItem(str(col))
          self.ui.table.setItem(i, j, item)


      elif Area_leen == 1 :
        data_1 = lista[0:1]
        data_2 = lista_2[0:1]
        for item_1 in data_1:
          a_1 = item_1
        for item_2 in data_2:
          a_2 = str(item_2)


        invoicesSql = (" SELECT * FROM Area WHERE %s = '%s'"% (a_1,a_2) )
        c.execute(invoicesSql)
        idata = c.fetchall()

        rows = len(idata)
        columns = len(idata[0])

        self.ui.table.setRowCount(rows)
        self.ui.table.setColumnCount(columns)

        self.ui.table.setHorizontalHeaderLabels(['Area ID', 'Area Name'])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table.setItem(i, j, item)
      if Area_leen == 2 :
        data_1 = lista[0:1]
        data_2 = lista_2[0:1]
        data_3 = lista[1:]
        data_4 = lista_2[1:]
        for item_1 in data_1:
          a_1 = item_1
        for item_2 in data_2:
          a_2 = item_2
        for item_3 in data_3:
          a_3 = item_3
        for item_4 in data_4:
          a_4 = item_4+'%'

        invoicesSql = ("""SELECT * FROM Area WHERE areaName LIKE '%s' AND %s =
'%s' ORDER BY areaId    """%( a_4, a_1 , a_2))
        c.execute(invoicesSql)
        idata = c.fetchall()
```

```
        rows = len(idata)
        columns = len(idata[0])

        self.ui.table.setRowCount(rows)
        self.ui.table.setColumnCount(columns)

        self.ui.table.setHorizontalHeaderLabels(['Area ID', 'Area Name'])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table.setItem(i, j, item)



    except:
      self.not_exist()

  def ar_ins(self):
    try:
      timi = self.ui.lineEdit_20.text()
      while timi == '':
        self.messege_ins()
        break
      else:
        c.execute("INSERT INTO Area(areaName) VALUES (?) ", (timi,) )
        db.commit()
        self.messege_ins_6()
        pl_id = c.lastrowid

        self.runArea()

        self.ui.comboBox.addItem("%s"% timi)
        self.ui.comboBox.update()
        self.ui.comboBox.currentIndexChanged[str]

    except:
      pass

  def area(self):
    try:

      a_t_up = str(self.ui.lineEdit_10.text())
      a_select = int(self.ui.table.currentItem().text())
      c.execute("UPDATE Area SET areaName = '%s' WHERE areaId = %s "%(a_t_up,a_select))
```

```
      db.commit()
      self.runArea()
      self.ui.comboBox.addItem("%s"% a_t_up)
      self.messege_ins_12()
    except:
      pass


  def runCustomer(self):
      invoicesSql = "select * from Customer"
      c.execute(invoicesSql)
      idata = c.fetchall()

      rows = len(idata)
      columns = len(idata[0])

      self.ui.table_cust.setRowCount(rows)
      self.ui.table_cust.setColumnCount(columns)

      self.ui.table_cust.setHorizontalHeaderLabels(['Customer ID', 'First Name','Last
Name','Phone','E-mail','Address', 'Post Code','City','Price PerKW','Area ID' ])
      for i, row in enumerate(idata):
        for j, col in enumerate(row):
          item = QTableWidgetItem(str(col))
          self.ui.table_cust.setItem(i, j, item)
  def DisplayDBDataCust(self):
      self.Cu_timi_1 = self.ui.lineEdit.text()
      self.Cu_timi_2 = self.ui.lineEdit_5.text()
      self.Cu_timi_3 = self.ui.lineEdit_6.text()
      self.Cu_timi_4 = self.ui.lineEdit_7.text()
      self.Cu_timi_5 = self.ui.lineEdit_8.text()
      self.Cu_timi_6 = self.ui.comboBox_2.currentText()
      self.Cust_insertdata()
  def Cust_insertdata(self):
 print(self.Cu_timi_1,self.Cu_timi_2,self.Cu_timi_3,self.Cu_timi_4,self.Cu_timi_5)

    lista = []
    lista_2 = []

    try:

      if self.Cu_timi_4 == '':
        pass
      else:
        self.cu_ppKW = 'pricePerKW'
        lista.append(self.cu_ppKW)
```

```
  lista_2.append(float(self.Cu_timi_4))

if self.Cu_timi_1 == '':
  pass
else:
  self.cu_id = 'customerId'
  lista.append(self.cu_id)
  lista_2.append(int(self.Cu_timi_1))

if self.Cu_timi_2 == '':
  pass
else:
  self.cu_pc = 'postcode'
  lista.append(self.cu_pc)
  lista_2.append(str(self.Cu_timi_2))

if self.Cu_timi_3 == '':
  pass
else:
  self.cu_cty = 'city'
  lista.append(self.cu_cty)
  lista_2.append(str(self.Cu_timi_3))



if self.Cu_timi_5 == '':
  pass
else:
  self.cu_aId = 'AreaId'
  lista.append(self.cu_aId)
  lista_2.append(int(self.Cu_timi_5))

Cust_leen  = (len(lista))
if Cust_leen == 1 :
  data_1 = lista[0:1]
  data_2 = lista_2[0:1]
  for item_1 in data_1:
    a_1 = item_1
  for item_2 in data_2:
    a_2 = str(item_2)
  if self.Cu_timi_6 == '>=':
    invoicesSql = ("SELECT * FROM Customer WHERE  %s >= '%s'"% (a_1,a_2) )
  elif self.Cu_timi_6 == '<=':
    invoicesSql = ("SELECT * FROM Customer WHERE  %s <= '%s'"% (a_1,a_2) )
  elif self.Cu_timi_6 == '==':
    invoicesSql = ("SELECT * FROM Customer WHERE  %s == '%s'"% (a_1,a_2) )
  else:
```

```
        a_3 = a_2+'%'
        invoicesSql = ("SELECT * FROM Customer WHERE %s LIKE '%s' ORDER BY %s
"% (a_1,a_3,a_1) )


       c.execute(invoicesSql)
       idata = c.fetchall()

       rows = len(idata)
       columns = len(idata[0])

       self.ui.table_cust.setRowCount(rows)
       self.ui.table_cust.setColumnCount(columns)
       self.ui.table_cust.setHorizontalHeaderLabels(['Customer ID', 'First Name','Last
Name','Phone','E-mail','Address', 'Post Code','City','Price PerKW','Area ID'])
       for i, row in enumerate(idata):
         for j, col in enumerate(row):
           item = QTableWidgetItem(str(col))
           self.ui.table_cust.setItem(i, j, item)

      elif Cust_leen == 2 :

       data_1 = lista[0:1]
       data_2 = lista_2[0:1]
       data_3 = lista[1:2]
       data_4 = lista_2[1:2]
       for item_1 in data_1:
         a_1 = item_1
       for item_2 in data_2:
         a_2 = item_2
       for item_3 in data_3:
         a_3 = item_3
       for item_4 in data_4:
         a_4 = item_4
       if self.Cu_timi_6 == '>=':
         if a_1 == 'pricePerKW':
           invoicesSql = ("SELECT * FROM Customer WHERE  %s >= '%s' AND %s =
'%s'"%( a_1, a_2 , a_3, a_4))
         else:
           invoicesSql = ("SELECT * FROM Customer WHERE  %s = '%s' AND  %s >=
'%s'"%( a_1, a_2 , a_3, a_4))

       elif self.Cu_timi_6 == '<=':
         if a_1 == 'pricePerKW':
           invoicesSql = ("SELECT * FROM Customer WHERE  %s <= '%s' AND %s =
'%s'"%( a_1, a_2 , a_3, a_4))
         else:
```

```
        invoicesSql = ("SELECT * FROM Customer WHERE   %s = '%s' AND  %s <= '%s'"%( a_1, a_2 , a_3, a_4))

      elif self.Cu_timi_6 == '==':
        if a_1 == 'pricePerKW':
          invoicesSql = ("SELECT * FROM Customer WHERE  %s == '%s' AND %s = '%s'"%( a_1, a_2 , a_3, a_4))
        else:
          invoicesSql = ("SELECT * FROM Customer WHERE   %s = '%s' AND  %s == '%s'"%( a_1, a_2 , a_3, a_4))
      else:
        invoicesSql = ("SELECT * FROM Customer WHERE   %s = '%s' AND  %s = '%s' "%( a_1, a_2 , a_3, a_4))
      c.execute(invoicesSql)
      idata = c.fetchall()
      rows = len(idata)
      columns = len(idata[0])

      self.ui.table_cust.setRowCount(rows)
      self.ui.table_cust.setColumnCount(columns)
      self.ui.table_cust.setHorizontalHeaderLabels(['Customer ID', 'First Name','Last Name','Phone','E-mail','Address', 'Post Code','City','Price PerKW','Area ID'])
      for i, row in enumerate(idata):
        for j, col in enumerate(row):
          item = QTableWidgetItem(str(col))
          self.ui.table_cust.setItem(i, j, item)


    elif Cust_leen == 3 :
      data_1 = lista[0:1]
      data_2 = lista_2[0:1]
      data_3 = lista[1:2]
      data_4 = lista_2[1:2]
      data_5 = lista[2:3]
      data_6 = lista_2[2:3]
      for item_1 in data_1:
        a_1 = item_1
      for item_2 in data_2:
        a_2 = item_2
      for item_3 in data_3:
        a_3 = item_3
      for item_4 in data_4:
        a_4 = item_4
      for item_5 in data_5:
        a_5 = item_5
      for item_6 in data_6:
        a_6 = item_6
```

```
##        print( a_1, a_2 , a_3, a_4, a_5, a_6)
       if self.Cu_timi_6 == '>=' and a_1 == 'pricePerKW':
          invoicesSql = ("SELECT * FROM Customer WHERE  %s >= %s AND %s = '%s'
AND %s = '%s' "%( a_1, float(a_2 ), a_3, a_4, a_5, a_6))

       elif self.Cu_timi_6 == '<=' and a_1 == 'pricePerKW':
          invoicesSql = ("SELECT * FROM Customer WHERE  %s <= '%s' AND %s = '%s'
AND %s = '%s'"%( a_1,float(a_2 ), a_3, a_4, a_5, a_6))

       elif self.Cu_timi_6 == '==' and a_1 == 'pricePerKW':
          invoicesSql = ("SELECT * FROM Customer WHERE  %s == '%s' AND %s = '%s'
AND %s = '%s'"%( a_1, float(a_2 ) , a_3, a_4, a_5, a_6))
       else:
         invoicesSql = ("SELECT * FROM Customer WHERE   %s = '%s' AND  %s = '%s'
AND %s = '%s'"%( a_1, a_2 , a_3, a_4, a_5, a_6))
       c.execute(invoicesSql)
       idata = c.fetchall()
##        print(idata)
       rows = len(idata)
       columns = len(idata[0])

       self.ui.table_cust.setRowCount(rows)
       self.ui.table_cust.setColumnCount(columns)
       self.ui.table_cust.setHorizontalHeaderLabels(['Customer ID', 'First Name','Last
Name','Phone','E-mail','Address', 'Post Code','City','Price PerKW','Area ID'])
       for i, row in enumerate(idata):
         for j, col in enumerate(row):
           item = QTableWidgetItem(str(col))
           self.ui.table_cust.setItem(i, j, item)




     elif Cust_leen == 4 :

       data_1 = lista[0:1]
       data_2 = lista_2[0:1]
       data_3 = lista[1:2]
       data_4 = lista_2[1:2]
       data_5 = lista[2:3]
       data_6 = lista_2[2:3]
       data_7 = lista[3:4]
       data_8 = lista_2[3:4]
       for item_1 in data_1:
         a_1 = item_1
```

```
        for item_2 in data_2:
          a_2 = item_2
        for item_3 in data_3:
          a_3 = item_3
        for item_4 in data_4:
          a_4 = item_4
        for item_5 in data_5:
          a_5 = item_5
        for item_6 in data_6:
          a_6 = item_6
        for item_7 in data_7:
          a_7 = item_7
        for item_8 in data_8:
          a_8 = item_8
        if self.Cu_timi_6 == '>=' and a_1 == 'pricePerKW':
          invoicesSql = ("SELECT * FROM Customer WHERE  %s >= %s AND %s = '%s'
AND %s = '%s' AND %s = '%s' "%( a_1, a_2 , a_3, a_4, a_5, a_6, a_7, a_8) )

        elif self.Cu_timi_6 == '<=' and a_1 == 'pricePerKW':
          invoicesSql = ("SELECT * FROM Customer WHERE  %s <= %s AND %s = '%s'
AND %s = '%s' AND %s = '%s' "%( a_1, a_2 , a_3, a_4, a_5, a_6, a_7, a_8) )

        elif self.Cu_timi_6 == '==' and a_1 == 'pricePerKW':
          invoicesSql = ("SELECT * FROM Customer WHERE  %s == %s AND %s = '%s'
AND %s = '%s' AND %s = '%s' "%( a_1, a_2 , a_3, a_4, a_5, a_6, a_7, a_8) )

        else:
          invoicesSql = ("SELECT * FROM Customer WHERE   %s = '%s' AND  %s = '%s'
AND %s = '%s' AND %s = '%s'"%( a_1, a_2 , a_3, a_4, a_5, a_6, a_7, a_8))
        c.execute(invoicesSql)
        idata = c.fetchall()
        rows = len(idata)
        columns = len(idata[0])

        self.ui.table_cust.setRowCount(rows)
        self.ui.table_cust.setColumnCount(columns)
        self.ui.table_cust.setHorizontalHeaderLabels(['Customer ID', 'First Name','Last
Name','Phone','E-mail','Address', 'Post Code','City','Price PerKW','Area ID'])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table_cust.setItem(i, j, item)


      else:
       self.runCustomer()
```

```
    except:
      self.not_exist()
  def Cu_delete(self):
    try:
      id_ =  int(self.ui.table_cust.currentItem().text())
      self.Cust_mess()
    except:
      self.messege_ins_8()
  def Cust_mess(self):
      if self.ui.table_cust.currentItem().text()=='':
        pass
      else:
        buttonReply = QMessageBox.question(self,'Delete!',
                                           "We will delete all items are involved with this customer Id. You
want to continue?",
                                           QMessageBox.Yes | QMessageBox.No)
      if buttonReply == QMessageBox.Yes :
        select = self.ui.table_cust.currentItem().text()
        try:
          c.execute("DELETE  FROM Customer WHERE customerId =='%s'" % select )
        except:
          pass
        try:
          c.execute("DELETE  FROM Consumption WHERE customerId =='%s'" % select )
        except:
          pass
        try:
          c.execute("DELETE  FROM Charge WHERE customerId =='%s'" % select )
        except:
          pass
        db.commit()
        self.runCustomer()
        self.runConsumption()
        self.runCharge()
        self.del_mess_1 = select
        self.del_messege_ins_1()
      else:
        pass

  def Cust_update(self):

    try:
      cust_t_up = float(self.ui.lineEdit_4.text())
      cust_select = int(self.ui.table_cust.currentItem().text())
      c.execute("UPDATE Customer SET pricePerKW = '%s' WHERE customerId = %s
"%(cust_t_up,cust_select))
```

```
      db.commit()
      self.runCustomer()
      self.messege_ins_12()
    except:
      pass

  def insert(self):
      try:
        timi_0 = self.ui.lineEdit_19.text()
        timi_1 = self.ui.lineEdit_21.text()
        timi_7 = self.ui.lineEdit_11.text()
        timi_8 = self.ui.lineEdit_16.text()

        timi_2 = self.ui.lineEdit_22.text()
        timi_3 = self.ui.lineEdit_23.text()
        timi_4 = self.ui.lineEdit_24.text()
        self.timi_5 = self.ui.lineEdit_25.text()
        timi_6 = self.ui.comboBox.currentText()
        while timi_0 == " or timi_1=="or timi_2=="or timi_3=="or timi_4=="or
self.timi_5=="or timi_6==" or timi_7==" or timi_8==":
          self.messege_ins()
          break
        else:
          c.execute("SELECT  areaId FROM Area WHERE areaName = '%s' "% timi_6)
          ID = c.fetchone()
          for i in ID:
            t_6 = i
          self.trans()
          self.times = ((timi_0,timi_1,timi_7,timi_8,timi_2,timi_3,timi_4,self.t_5,t_6))
          self.inst()

      except:
        pass
  def trans(self):
    try:
      self.t_5 = float(self.timi_5)
    except:
      self.messege_ins_2()
      self.messege_ins_5()
  def inst(self):

    c.execute("INSERT                    INTO                    Customer
(firstName,lastName,phone,email,address,postcode,city,pricePerKW,areaId)VALUES(?,?,?,?
,?,?,?,?,?)",self.times)
    db.commit()
```

```
    ##                invoicesSql_2 = "select * from Consumption where totalKW >=
{0}".format(timi_2)
    ##       c.execute(invoicesSql_2)
   c.execute("SELECT * FROM Customer ")
   idata_2 = c.fetchall()
##    print(idata_2)
   self.runCustomer()
   self.messege_ins_6()

  def runConsumption(self):
     invoicesSql = "select * from Consumption"
     c.execute(invoicesSql)
     idata = c.fetchall()

     rows = len(idata)
     columns = len(idata[0])

     self.ui.table_con.setRowCount(rows)
     self.ui.table_con.setColumnCount(columns)

     self.ui.table_con.setHorizontalHeaderLabels(['Consumption ID ', 'Start Date','End
Date','Total KW','Customer ID' ])
     for i, row in enumerate(idata):
       for j, col in enumerate(row):
         item = QTableWidgetItem(str(col))
         self.ui.table_con.setItem(i, j, item)
  def DisplayDBDataCon(self):

   date_3 = self.ui.dateEdit_3.date()
   date_4 = self.ui.dateEdit_4.date()
   self.cons_timi_1 = date_3.toPyDate()
   self.cons_timi_2 = date_4.toPyDate()

   self.cons_timi_3 = self.ui.comboBox_4.currentText()
   self.cons_timi_4 = self.ui.lineEdit_12.text()

   self.cons_timi_5 = self.ui.lineEdit_9.text()
   self.cons_timi_6 = self.ui.lineEdit_13.text()



   self.Cons_insertdata()

  def Cons_insertdata(self):
   if self.ui.radioButton.isChecked() and self.ui.radioButton_2.isChecked():
     self.messege_ins_11()
```

```
else:

  item = self.cons_timi_4
  lista = []
  lista_2 = []

  try:

    if self.ui.checkBox_2.isChecked():
      self.stdate = 'startDate'
      self.endate = 'endDate'
      lista.append(self.stdate)
      lista.append(self.endate)
      lista_2.append(str(self.cons_timi_1))
      lista_2.append(str(self.cons_timi_2))

    else:
      pass

    if   self.cons_timi_4 == '' and self.cons_timi_3 == '':
      pass
    elif self.cons_timi_4 == '' and self.cons_timi_3 == '>=':
      pass
    elif self.cons_timi_4 == '' and self.cons_timi_3 == '<=':
      pass
    elif self.cons_timi_4 == '' and self.cons_timi_3 == '==':
      pass
    elif self.cons_timi_4 == item and self.cons_timi_3 == '' :
      pass
    else:
      self.cons_tkw = 'totalKW'
      lista.append(self.cons_tkw)
      lista_2.append(float(self.cons_timi_4))

    if self.cons_timi_5 == '':
      pass
    else:
      self.cons_id = 'consumeId'
      lista.append(self.cons_id)
      lista_2.append(int(self.cons_timi_5))


    if self.cons_timi_6 == '':
      pass
    else:
      self.cons_cId = 'customerId'
      lista.append(self.cons_cId)
```

```
        lista_2.append(int(self.cons_timi_6))




      Cons_leen  = (len(lista))
      if Cons_leen == 1 :
        data_1 = lista[0:1]
        data_2 = lista_2[0:1]

        for item_1 in data_1:
          a_1 = item_1
        for item_2 in data_2:
          a_2 = item_2
        if self.cons_timi_3 == '>=' and a_1 ==  'totalKW':
          invoicesSql = ("SELECT * FROM Consumption WHERE  %s >= '%s' "%
(a_1,float(a_2) ))

        elif self.cons_timi_3 == '<=' and a_1 ==  'totalKW':
          invoicesSql = ("SELECT * FROM Consumption WHERE  %s <= '%s'"%
(a_1,float(a_2) ))
        elif self.cons_timi_3 == '==' and a_1 ==  'totalKW':
           invoicesSql = ("SELECT * FROM Consumption WHERE  %s == '%s'"%
(a_1,float(a_2) ))
        else:
           invoicesSql = (" SELECT * FROM Consumption WHERE %s = '%s'"% (a_1,a_2) )

        c.execute(invoicesSql)
        idata = c.fetchall()

        rows = len(idata)
        columns = len(idata[0])

        self.ui.table_con.setRowCount(rows)
        self.ui.table_con.setColumnCount(columns)

        self.ui.table_con.setHorizontalHeaderLabels(['Consumption ID ', 'Start Date','End
Date','Total KW','Customer ID' ])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table_con.setItem(i, j, item)
      elif Cons_leen == 2 :

        data_1 = lista[0:1]
```

```
data_2 = lista_2[0:1]
data_3 = lista[1:2]
data_4 = lista_2[1:2]
for item_1 in data_1:
  a_1 = item_1
for item_2 in data_2:
  a_2 = item_2
for item_3 in data_3:
  a_3 = item_3
for item_4 in data_4:
  a_4 = item_4
if a_1 =='startDate' :

  data_1 = lista[0:1]
  data_2 = lista[1:2]

  data_3 = lista_2[0:1]
  data_4 = lista_2[1:2]


  for item_1 in data_1:
    a_1 = item_1
  for item_2 in data_2:
    a_2 = item_2


  for item_3 in data_3:
    a_3 = item_3
    lookForMonth = a_3[5:7]



  for item_4 in data_4:
    a_4 = item_4
  if lookForMonth == '8':
      lookForMonth_2 = 1

  elif lookForMonth == '9':
      lookForMonth_2 = 2

  elif lookForMonth == '10':
      lookForMonth_2 = 3

  elif lookForMonth=='11':
      lookForMonth_2 = 4

  elif lookForMonth=='12':
```

```
            lookForMonth_2 = 5
          else:
            item = int(lookForMonth)
            lookForMonth_2 = item+6



          if self.ui.radioButton.isChecked():
            invoicesSql = ("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND endDate <= '%s'
"""% (int(lookForMonth),int(lookForMonth_2),a_4))
          elif self.ui.radioButton_2.isChecked():
            invoicesSql = ("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND endDate <= '%s'
"""% (int(lookForMonth),int(lookForMonth),a_4))
          else:
            invoicesSql = ("SELECT * FROM Consumption WHERE  startDate  BETWEEN
'%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s'  "% (a_3,a_4,a_3,a_4))

        elif a_1 ==  'totalKW' and self.cons_timi_3 == '>=':
            invoicesSql = ("SELECT * FROM Consumption WHERE  %s >= '%s' AND  %s =
'%s' "% (a_1,float(a_2),a_3,a_4))
        elif a_1 ==  'totalKW' and self.cons_timi_3 == '<=':
            invoicesSql = ("SELECT * FROM Consumption WHERE  %s <= '%s' AND  %s =
'%s' "% (a_1,float(a_2),a_3,a_4))
        elif a_1 ==  'totalKW' and self.cons_timi_3 == '==':
            invoicesSql = ("SELECT * FROM Consumption WHERE  %s == '%s' AND  %s =
'%s' "% (a_1,float(a_2),a_3,a_4))

        else:
          invoicesSql = (" SELECT * FROM Consumption WHERE %s = '%s' AND %s = '%s'
"% (a_1,a_2,a_3,a_4) )

        c.execute(invoicesSql)
        idata = c.fetchall()

        rows = len(idata)
        columns = len(idata[0])

        self.ui.table_con.setRowCount(rows)
        self.ui.table_con.setColumnCount(columns)

        self.ui.table_con.setHorizontalHeaderLabels(['Consumption  ID  ', 'Start  Date','End
Date','Total KW','Customer ID' ])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
```

```
            self.ui.table_con.setItem(i, j, item)
        elif Cons_leen == 3 :
          data_1 = lista[0:1]
          data_2 = lista_2[0:1]
          data_3 = lista[1:2]
          data_4 = lista_2[1:2]
          data_5 = lista[2:3]
          data_6 = lista_2[2:3]
          for item_1 in data_1:
            a_1 = item_1
          for item_2 in data_2:
            a_2 = item_2
          for item_3 in data_3:
            a_3 = item_3
          for item_4 in data_4:
            a_4 = item_4
          for item_5 in data_5:
            a_5 = item_5
          for item_6 in data_6:
            a_6 = item_6
          if a_1 ==  'totalKW' and self.cons_timi_3 == '>=':
            invoicesSql = ("SELECT * FROM Consumption WHERE  %s >= '%s' AND  %s =
'%s' AND  %s = '%s'"% (a_1,float(a_2),a_3,a_4,a_5, a_6))
          elif a_1 ==  'totalKW' and self.cons_timi_3 == '<=':
            invoicesSql = ("SELECT * FROM Consumption WHERE  %s <= '%s' AND  %s =
'%s' AND  %s = '%s'"% (a_1,float(a_2),a_3,a_4,a_5, a_6))
          elif a_1 == 'totalKW' and self.cons_timi_3 == '==':
            invoicesSql = ("SELECT * FROM Consumption WHERE  %s == '%s' AND  %s =
'%s' AND  %s = '%s' "% (a_1,float(a_2),a_3,a_4,a_5, a_6))

          elif a_1 =='startDate' :

            data_1 = lista[0:1]
            data_2 = lista[1:2]
            data_3 = lista[2:3]
            data_4 = lista_2[0:1]
            data_5 = lista_2[1:2]
            data_6 = lista_2[2:3]
            for item_1 in data_1:
              a_1 = item_1
            for item_2 in data_2:
              a_2 = item_2
            for item_3 in data_3:
              a_3 = item_3
            for item_4 in data_4:
              a_4 = item_4
              lookForMonth = a_4[5:7]
```

```
        for item_5 in data_5:
          a_5 = item_5
        for item_6 in data_6:
          a_6 = item_6
        if lookForMonth == '8':
            lookForMonth_2 = 1

        elif lookForMonth == '9':
            lookForMonth_2 = 2

        elif lookForMonth == '10':
            lookForMonth_2 = 3

        elif lookForMonth=='11':
            lookForMonth_2 = 4

        elif lookForMonth=='12':
            lookForMonth_2 = 5
        else:
            item = int(lookForMonth)
            lookForMonth_2 = item+6


        if self.ui.radioButton.isChecked():

          if a_3 ==  'totalKW' and self.cons_timi_3 == '>=':
            invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s >= '%s' AND
endDate <= '%s' """% (int(lookForMonth),int(lookForMonth_2),a_3,float(a_6),a_5))
          elif a_3 ==  'totalKW' and self.cons_timi_3 == '<=':
            invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s <= '%s' AND
endDate <= '%s' """% (int(lookForMonth),int(lookForMonth_2),a_3,float(a_6),a_5))
          elif a_3 == 'totalKW' and self.cons_timi_3 == '==':
            invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s == '%s' AND
endDate <= '%s' """% (int(lookForMonth),int(lookForMonth_2),a_3,float(a_6),a_5))
          else:
            invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s = '%s' AND
endDate <= '%s'   """% (int(lookForMonth),int(lookForMonth_2),a_3,a_6,a_5))



        elif self.ui.radioButton_2.isChecked():
          if a_3 ==  'totalKW' and self.cons_timi_3 == '>=':
```

```
            invoicesSql =("""SELECT * FROM Consumption WHERE SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s >= '%s' AND endDate <= '%s' """% (int(lookForMonth),int(lookForMonth),a_3,float(a_6),a_5))
          elif a_3 ==  'totalKW' and self.cons_timi_3 == '<=':
            invoicesSql =("""SELECT * FROM Consumption WHERE SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s <= '%s' AND endDate <= '%s' """% (int(lookForMonth),int(lookForMonth),a_3,float(a_6),a_5))
          elif a_3 == 'totalKW' and self.cons_timi_3 == '==':
            invoicesSql =("""SELECT * FROM Consumption WHERE SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s == '%s' AND endDate <= '%s' """% (int(lookForMonth),int(lookForMonth),a_3,float(a_6),a_5))
          else:
            invoicesSql =("""SELECT * FROM Consumption WHERE SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s = '%s' AND endDate <= '%s' """% (int(lookForMonth),int(lookForMonth),a_3,a_6,a_5))




        else:
          if a_3 ==  'totalKW' and self.cons_timi_3 == '>=':
            invoicesSql =("SELECT * FROM Consumption WHERE  startDate BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s >= '%s' "% (a_4,a_5,a_4,a_5,a_3,float(a_6)))
          elif a_3 ==  'totalKW' and self.cons_timi_3 == '<=':
            invoicesSql =("SELECT * FROM Consumption WHERE  startDate BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s <= '%s' "% (a_4,a_5,a_4,a_5,a_3,float(a_6)))
          elif a_3 == 'totalKW' and self.cons_timi_3 == '==':
            invoicesSql =("SELECT * FROM Consumption WHERE  startDate BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s == '%s' "% (a_4,a_5,a_4,a_5,a_3,float(a_6)))
          else:
           invoicesSql =("SELECT * FROM Consumption WHERE  startDate BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s = '%s' "% (a_4,a_5,a_4,a_5,a_3,a_6))

        c.execute(invoicesSql)
        idata = c.fetchall()

        rows = len(idata)
        columns = len(idata[0])

        self.ui.table_con.setRowCount(rows)
        self.ui.table_con.setColumnCount(columns)
```

```
        self.ui.table_con.setHorizontalHeaderLabels(['Consumption ID ', 'Start Date','End Date','Total KW','Customer ID' ])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table_con.setItem(i, j, item)

       elif Cons_leen == 4 :
        data_1 = lista[0:1]
        data_2 = lista[1:2]
        data_3 = lista[2:3]
        data_4 = lista[3:4]

        data_5 = lista_2[0:1]
        data_6 = lista_2[1:2]
        data_7 = lista_2[2:3]
        data_8 = lista_2[3:4]

 (data_1,data_2,data_3,data_4,data_5,data_6,data_7,data_8)

        for item_1 in data_1:
          a_1 = item_1
        for item_2 in data_2:
          a_2 = item_2
        for item_3 in data_3:
          a_3 = item_3
        for item_4 in data_4:
          a_4 = item_4

        for item_5 in data_5:
          a_5 = item_5
          lookForMonth = a_5[5:7]
        for item_6 in data_6:
          a_6 = item_6
        for item_7 in data_7:
          a_7 = item_7
        for item_8 in data_8:
          a_8 = item_8
        if lookForMonth == '8':
            lookForMonth_2 = 1

        elif lookForMonth == '9':
            lookForMonth_2 = 2

        elif lookForMonth == '10':
            lookForMonth_2 = 3
```

```
        elif lookForMonth=='11':
            lookForMonth_2 = 4

        elif lookForMonth=='12':
            lookForMonth_2 = 5
        else:
            item = int(lookForMonth)
            lookForMonth_2 = item+6

        if self.ui.radioButton.isChecked():

          if a_3 ==  'totalKW' and self.cons_timi_3 == '>=':
              invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s >= '%s' AND
%s = '%s' AND endDate <= '%s' """%
(int(lookForMonth),int(lookForMonth_2),a_3,float(a_7),a_4,a_8,a_6))
          elif a_3 ==  'totalKW' and self.cons_timi_3 == '<=':
              invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s <= '%s' AND
%s = '%s' AND endDate <= '%s' """%
(int(lookForMonth),int(lookForMonth_2),a_3,float(a_7),a_4,a_8,a_6))
          elif a_3 == 'totalKW' and self.cons_timi_3 == '==':
              invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s == '%s' AND
%s = '%s' AND endDate <= '%s' """%
(int(lookForMonth),int(lookForMonth_2),a_3,float(a_7),a_4,a_8,a_6))
          else:
   ##           print('ok')
              invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s = '%s' AND
%s = '%s' AND endDate <= '%s' """%
(int(lookForMonth),int(lookForMonth_2),a_3,a_7,a_4,a_8,a_6))



        elif self.ui.radioButton_2.isChecked():
          if a_3 ==  'totalKW' and self.cons_timi_3 == '>=':
              invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s >= '%s' AND
%s = '%s' AND endDate <= '%s' """%
(int(lookForMonth),int(lookForMonth),a_3,float(a_7),a_4,a_8,a_6))
          elif a_3 ==  'totalKW' and self.cons_timi_3 == '<=':
              invoicesSql =("""SELECT * FROM Consumption WHERE
SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s <= '%s' AND
%s = '%s' AND endDate <= '%s' """%
(int(lookForMonth),int(lookForMonth),a_3,float(a_7),a_4,a_8,a_6))
          elif a_3 == 'totalKW' and self.cons_timi_3 == '==':
```

```
            invoicesSql =("""SELECT * FROM Consumption WHERE SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s == '%s' AND %s = '%s' AND endDate <= '%s' """% (int(lookForMonth),int(lookForMonth),a_3,float(a_7),a_4,a_8,a_6))
          else:

            invoicesSql =("""SELECT * FROM Consumption WHERE SUBSTR(startDate,6,2)="%.2i" AND SUBSTR(endDate,6,2)="%.2i" AND %s = '%s' AND %s = '%s' AND endDate <= '%s' """% (int(lookForMonth),int(lookForMonth),a_3,a_7,a_4,a_8,a_6))




        else:
          if a_3 ==  'totalKW' and self.cons_timi_3 == '>=':
            invoicesSql =("SELECT * FROM Consumption WHERE  startDate  BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s >= '%s' AND %s = '%s' "% (a_5,a_6,a_5,a_6,a_3,float(a_7),a_4,a_8))
          elif a_3 ==  'totalKW' and self.cons_timi_3 == '<=':
            invoicesSql =("SELECT * FROM Consumption WHERE  startDate  BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s <= '%s' AND %s = '%s' "% (a_5,a_6,a_5,a_6,a_3,float(a_7),a_4,a_8))
          elif a_3 == 'totalKW' and self.cons_timi_3 == '==':
            invoicesSql =("SELECT * FROM Consumption WHERE  startDate  BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s == '%s' AND %s = '%s' "% (a_5,a_6,a_5,a_6,a_3,float(a_7),a_4,a_8))
          else:
            invoicesSql =("SELECT * FROM Consumption WHERE  startDate  BETWEEN '%s'  AND '%s'  AND  endDate BETWEEN '%s' AND '%s' AND %s = '%s' AND %s = '%s' "% (a_5,a_6,a_5,a_6,a_3,a_7,a_4,a_8))




        c.execute(invoicesSql)
        idata = c.fetchall()
```

```
        rows = len(idata)
        columns = len(idata[0])

        self.ui.table_con.setRowCount(rows)
        self.ui.table_con.setColumnCount(columns)

        self.ui.table_con.setHorizontalHeaderLabels(['Consumption ID ', 'Start Date','End Date','Total KW','Customer ID' ])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table_con.setItem(i, j, item)




       else:
         pass
     except:

       self.not_exist()
  def Co_delete(self):

    try:
      id_ = int( self.ui.table_con.currentItem().text())
      con_tp = type(id_)
      if (con_tp == int) == True:

       buttonReply = QMessageBox.question(self,'The document has been modified.', "Do you want to save your changes?.", QMessageBox.Cancel ,QMessageBox.Save)
       if buttonReply == QMessageBox.Save:
         co_select = self.ui.table_con.currentItem().text()

         c.execute("DELETE  FROM Consumption WHERE consumeId ='%s'" % co_select )
         db.commit()


         self.del_mess_2 = co_select
         self.del_messege_ins_2()
         self.runConsumption()
       else:
         pass
      else:
       self.messege_ins_8()
    except:
      pass
  def Con_update(self):
```

```
    try:
      co_t_up = float(self.ui.lineEdit_3.text())
      co_select = int(self.ui.table_con.currentItem().text())
##      print(co_t_up,co_select)
      c.execute("UPDATE Consumption SET totalKW = '%s' WHERE consumeId = %s "%(co_t_up,co_select))

      db.commit()
      self.runConsumption()
      self.messege_ins_12()
    except:
      pass
  def Con_insert(self):
      try:
        date_5 = self.ui.dateEdit_5.date()
        date_6 = self.ui.dateEdit_6.date()
        self.co_timi_1 = date_5.toPyDate()
        self.co_timi_2 = date_6.toPyDate()
        self.co_timi_3 = self.ui.lineEdit_26.text()
        self.co_timi_4 = self.ui.lineEdit_31.text()

        while  self.co_timi_1==''or self.co_timi_2==''or self.co_timi_3==' or self.co_timi_4==' :
          self.messege_ins()
          break
        else:
          self.con_check()
          self.co_times = ((self.co_timi_1,self.co_timi_2,self.kw_data,self.id_data))
          self.co_inst()

      except:
        pass

  def con_check(self):
      self.kw_data = float(self.co_timi_3)
      self.id_data = int(self.co_timi_4)
    except:
      self.messege_ins_4()
      self.messege_ins_5()
  def co_inst(self):
    c.execute("INSERT INTO Consumption (startDate,endDate,totalKW,customerId)VALUES(?,?,?,?)",self.co_times)
    db.commit()
    self.runConsumption()
    self.messege_ins_6()
  def runCharge(self):
```

```
    invoicesSql = "select * from Charge"
    c.execute(invoicesSql)
    idata = c.fetchall()

    rows = len(idata)
    columns = len(idata[0])

    self.ui.table_cha.setRowCount(rows)
    self.ui.table_cha.setColumnCount(columns)

    self.ui.table_cha.setHorizontalHeaderLabels(['Charge ID', 'Charge Date','KW','Total
Price','Customer ID' ])
    for i, row in enumerate(idata):
      for j, col in enumerate(row):
        item = QTableWidgetItem(str(col))
        self.ui.table_cha.setItem(i, j, item)

  def DisplayDBDataCha(self):

    self.charge_timi_1 = self.ui.comboBox_5.currentText()
    self.charge_timi_2 = self.ui.comboBox_6.currentText()

    self.charge_timi_3 = self.ui.lineEdit_17.text()
    self.charge_timi_4 = self.ui.lineEdit_18.text()

    date_1 = self.ui.dateEdit.date()
    date_2 = self.ui.dateEdit_2.date()
    self.charge_timi_5   = date_1.toPyDate()
    self.charge_timi_5_2 = date_2.toPyDate()

    self.charge_timi_6 = self.ui.lineEdit_15.text()
    self.charge_timi_7 = self.ui.lineEdit_14.text()

    self.Cha_insertdata()

  def Cha_insertdata(self):
print(self.charge_timi_3,self.charge_timi_4,self.charge_timi_5,self.charge_timi_5_2,self.char
ge_timi_6,self.charge_timi_7)
   lista = []
   lista_2 = []
   try:
     if self.charge_timi_3 == '':
       pass
     else:
       self.cha_kw = 'KW'
       lista.append(self.cha_kw)
       lista_2.append(float(self.charge_timi_3))
```

```
if self.charge_timi_4 == '':
  pass
else:
  self.cha_tol = 'totalPrice'
  lista.append(self.cha_tol)
  lista_2.append(float(self.charge_timi_4))


if self.ui.checkBox.isChecked():
  self.cha_chday = 'chargeDate'
  lista.append(self.cha_chday )
  lista_2.append(str(self.charge_timi_5))
  lista_2.append(str(self.charge_timi_5_2))
else:
  pass



if self.charge_timi_6 == '':
  pass
else:
  self.cha_id = 'chargeId'
  lista.append(self.cha_id )
  lista_2.append(int(self.charge_timi_6))

if self.charge_timi_7 == '':
  pass
else:
  self.cha_cu_id = 'customerId'
  lista.append(self.cha_cu_id)
  lista_2.append(int(self.charge_timi_7))

Cha_leen  = (len(lista))

if Cha_leen == 1 :

  data_1 = lista[0:1]
  data_2 = lista_2[0:1]
  for item_1 in data_1:
    a_1 = item_1
  for item_2 in data_2:
    a_2 = item_2
  if self.charge_timi_1 == '>=':
    invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' "% (a_1,float(a_2) ))

  elif self.charge_timi_1 == '<=':
```

```
        invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' "% (a_1,float(a_2) ))
      elif self.charge_timi_1 == '==':
        invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' "% (a_1,float(a_2) ))

      elif self.charge_timi_2 == '>=':
        invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' "% (a_1,float(a_2) ))

      elif self.charge_timi_2 == '<=':
        invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' "% (a_1,float(a_2) ))
      elif self.charge_timi_2 == '==':
        invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' "% (a_1,float(a_2) ))


      elif a_1 == 'chargeDate':


        data_1 = lista_2[0:1]
        data_2 = lista_2[1:2]

        for item_1 in data_1:
          a_1 = item_1
        for item_2 in data_2:
          a_2 = item_2

        invoicesSql = (""" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' """% (a_1,a_2) )



      else:




      c.execute(invoicesSql)
      idata = c.fetchall()

      rows = len(idata)
      columns = len(idata[0])

      self.ui.table_cha.setRowCount(rows)
      self.ui.table_cha.setColumnCount(columns)

      self.ui.table_cha.setHorizontalHeaderLabels(['Charge ID', 'Charge Date','KW','Total
Price','Customer ID' ])
      for i, row in enumerate(idata):
```

```
        for j, col in enumerate(row):
          item = QTableWidgetItem(str(col))
          self.ui.table_cha.setItem(i, j, item)

      if Cha_leen == 2 :
        data_1 = lista[0:1]
        data_2 = lista_2[0:1]
        data_3 = lista[1:2]
        data_4 = lista_2[1:2]

        for item_1 in data_1:
          a_1 = item_1
        for item_2 in data_2:
          a_2 = item_2
        for item_3 in data_3:
          a_3 = item_3
        for item_4 in data_4:
          a_4 = item_4

        if a_1 == 'chargeDate' or a_3 == 'chargeDate':
          data_1 = lista[0:1]
          data_2 = lista_2[0:1]
          data_3 = lista_2[1:2]

          data_4 = lista[1:2]
          data_5 = lista_2[2:3]

          for item_1 in data_1:
            a_1 = item_1#----------------------lista
          for item_2 in data_2:
            a_2 = item_2
          for item_3 in data_3:
            a_3 = item_3
          for item_4 in data_4:
            a_4 = item_4
          for item_5 in data_5:
             a_5 = item_5
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND
'%s' AND %s = '%s' "% (a_2,a_3,a_4,a_5))
          if a_1 == 'KW' and self.charge_timi_1 == '>=' and  a_4 == 'chargeDate':
            invoicesSql = ("""SELECT * FROM Charge WHERE %s >= '%s' AND chargeDate
BETWEEN '%s' AND '%s'  """ % (a_1,float(a_2),a_3,a_5))

          elif a_1 == 'KW' and self.charge_timi_1 == '<=' and  a_4 == 'chargeDate':
            invoicesSql = ("""SELECT * FROM Charge WHERE %s <= '%s' AND chargeDate
BETWEEN '%s' AND '%s'  """ % (a_1,float(a_2),a_3,a_5))
```

```
        elif a_1 == 'KW' and self.charge_timi_1 == '==' and  a_4 == 'chargeDate':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s == '%s' AND chargeDate
BETWEEN '%s' AND '%s'  """ % (a_1,float(a_2),a_3,a_5))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '>=' and  a_4 == 'chargeDate':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s >= '%s' AND chargeDate
BETWEEN '%s' AND '%s'  """ % (a_1,float(a_2),a_3,a_5))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '<=' and  a_4 == 'chargeDate':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s <= '%s' AND chargeDate
BETWEEN '%s' AND '%s'  """ % (a_1,float(a_2),a_3,a_5))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '==' and  a_4 == 'chargeDate':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s == '%s' AND chargeDate
BETWEEN '%s' AND '%s'  """ % (a_1,float(a_2),a_3,a_5))

       else:

        if a_1 == 'KW' and a_3 == 'totalPrice':
          if  self.charge_timi_1 == '>=' and self.charge_timi_2 == '>=':
            invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s >= '%s'
"% (a_1,float(a_2),a_3,float(a_4) ))

          elif self.charge_timi_1 == '<=' and self.charge_timi_2 == '<=':
            invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s <= '%s'
"% (a_1,float(a_2),a_3,float(a_4) ))

          elif self.charge_timi_1 == '>=' and self.charge_timi_2 == '<=':
            invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s <= '%s'
"% (a_1,float(a_2),a_3,float(a_4) ))
          elif self.charge_timi_1 == '<=' and self.charge_timi_2 == '>=':
            invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s >= '%s'
"% (a_1,float(a_2),a_3,float(a_4) ))

          elif self.charge_timi_1 == '>=' and self.charge_timi_2 == '==':
            invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s == '%s'
"% (a_1,float(a_2),a_3,float(a_4) ))

          elif self.charge_timi_1 == '==' and self.charge_timi_2 == '>=':
            invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s >= '%s'
"% (a_1,float(a_2),a_3,float(a_4) ))

          elif self.charge_timi_1 == '<=' and self.charge_timi_2 == '==' :
            invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s == '%s'
"% (a_1,float(a_2),a_3,float(a_4) ))

          elif self.charge_timi_1 == '==' and self.charge_timi_2 == '<=' :
```

```
            invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s <= '%s' "% (a_1,float(a_2),a_3,float(a_4) ))
          else:
            invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s == '%s' "% (a_1,float(a_2),a_3,float(a_4) ))


        elif a_1 == 'KW' and self.charge_timi_1 == '>=':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s >= '%s' AND %s = '%s' """ % (a_1,float(a_2),a_3,int(a_4)  ))

        elif a_1 == 'KW' and self.charge_timi_1 == '<=':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s <= '%s' AND %s = '%s' """ % (a_1,float(a_2),a_3,int(a_4)  ))

        elif a_1 == 'KW' and self.charge_timi_1 == '==':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s == '%s' AND %s = '%s' """ % (a_1,float(a_2),a_3,int(a_4)  ))


        elif a_1 == 'totalPrice' and self.charge_timi_2 == '>=':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s >= '%s' AND %s = '%s' """ % (a_1,float(a_2),a_3,int(a_4)  ))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '<=':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s <= '%s' AND %s = '%s' """ % (a_1,float(a_2),a_3,int(a_4)  ))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '==':
          invoicesSql = ("""SELECT * FROM Charge WHERE %s == '%s' AND %s = '%s' """ % (a_1,float(a_2),a_3,int(a_4)  ))




        else:
          invoicesSql = (" SELECT * FROM Charge WHERE  %s = '%s' AND %s = '%s' "% (a_1,a_2,a_3,int(a_4)))




      c.execute(invoicesSql)
      idata = c.fetchall()

      rows = len(idata)
```

```
        columns = len(idata[0])
        self.ui.table_cha.setRowCount(rows)
        self.ui.table_cha.setColumnCount(columns)

        self.ui.table_cha.setHorizontalHeaderLabels(['Charge ID', 'Charge Date','KW','Total Price','Customer ID'])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table_cha.setItem(i, j, item)
      if Cha_leen == 3 :
        data_1 = lista[0:1]
        data_2 = lista_2[0:1]
        data_3 = lista[1:2]
        data_4 = lista_2[1:2]
        data_5 = lista[2:3]
        data_6 = lista_2[2:3]

        for item_1 in data_1:
          a_1 = item_1
        for item_2 in data_2:
          a_2 = item_2
        for item_3 in data_3:
          a_3 = item_3
        for item_4 in data_4:
          a_4 = item_4
        for item_5 in data_5:
          a_5 = item_5
        for item_6 in data_6:
          a_6 = item_6

        if a_1 == 'chargeDate' or a_3 == 'chargeDate' or a_5 == 'chargeDate':
          data_1 = lista[0:1]
          data_2 = lista_2[0:1]

          data_3 = lista[1:2]
          data_4 = lista_2[1:2]

          data_5 = lista[2:3]
          data_6 = lista_2[2:3]
          data_7 = lista_2[3:4]
          for item_1 in data_1:
            a_1 = item_1
          for item_2 in data_2:
            a_2 = item_2
          for item_3 in data_3:
            a_3 = item_3
```

```
        for item_4 in data_4:
          a_4 = item_4
        for item_5 in data_5:
          a_5 = item_5
        for item_6 in data_6:
          a_6 = item_6
        for item_7 in data_7:
          a_7 = item_7

        invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND
'%s' AND %s = '%s' AND %s = '%s'"% (a_2,a_4,a_3,a_6,a_5,a_7))

        if a_1 == 'KW' and self.charge_timi_1 == '>=' and a_3 == 'totalPrice' and
self.charge_timi_2 == '>=' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' AND %s >= '%s' AND %s >= '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))

        elif a_1 == 'KW' and self.charge_timi_1 == '<=' and a_3 == 'totalPrice' and
self.charge_timi_2 == '<=' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' AND %s <= '%s' AND %s <= '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))

        elif a_1 == 'KW' and self.charge_timi_1 == '<=' and a_3 == 'totalPrice' and
self.charge_timi_2 == '>=' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' AND %s <= '%s' AND %s >= '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))

        elif a_1 == 'KW' and self.charge_timi_1 == '>=' and a_3 == 'totalPrice' and
self.charge_timi_2 == '<=' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' AND %s >= '%s' AND %s <= '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))

        elif a_1 == 'KW' and self.charge_timi_1 == '==' and a_3 == 'totalPrice' and
self.charge_timi_2 == '<=' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' AND %s == '%s' AND %s <= '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))

        elif a_1 == 'KW' and self.charge_timi_1 == '==' and a_3 == 'totalPrice' and
self.charge_timi_2 == '>=' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' AND %s == '%s' AND %s >= '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))

        elif a_1 == 'KW' and self.charge_timi_1 == '<=' and a_3 == 'totalPrice' and
self.charge_timi_2 == '==' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s'
AND '%s' AND %s <= '%s' AND %s == '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))
```

```
        elif a_1 == 'KW' and self.charge_timi_1 == '>=' and a_3 == 'totalPrice' and self.charge_timi_2 == '==' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s >= '%s' AND %s == '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))

        elif a_1 == 'KW' and self.charge_timi_1 == '==' and a_3 == 'totalPrice' and self.charge_timi_2 == '==' and  a_5 == 'chargeDate':
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s == '%s' AND %s == '%s'"% (a_6,a_7,a_1,a_2,a_3,a_4))



        elif a_1 == 'KW' and self.charge_timi_1 == '>=' and a_3 == 'chargeDate' :
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s >= '%s' AND %s = '%s'"% (a_4,a_6,a_1,a_2,a_5,a_7))

        elif a_1 == 'KW' and self.charge_timi_1 == '<=' and a_3 == 'chargeDate' :
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s <= '%s' AND %s = '%s'"% (a_4,a_6,a_1,a_2,a_5,a_7))

        elif a_1 == 'KW' and self.charge_timi_1 == '==' and a_3 == 'chargeDate' :
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s == '%s' AND %s = '%s'"% (a_4,a_6,a_1,a_2,a_5,a_7))



        elif a_1 == 'totalPrice' and self.charge_timi_2 == '>=' and a_3 == 'chargeDate' :
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s >= '%s' AND %s = '%s'"% (a_4,a_6,a_1,a_2,a_5,a_7))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '<=' and a_3 == 'chargeDate' :
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s <= '%s' AND %s = '%s'"% (a_4,a_6,a_1,a_2,a_5,a_7))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '==' and a_3 == 'chargeDate' :
          invoicesSql =(" SELECT * FROM Charge WHERE chargeDate BETWEEN '%s' AND '%s' AND %s == '%s' AND %s = '%s'"% (a_4,a_6,a_1,a_2,a_5,a_7))
        else:
          pass
      else:

        if a_1 == 'KW' and a_3 == 'totalPrice':
          if  self.charge_timi_1 == '>=' and self.charge_timi_2 == '>=':
            invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s >= '%s' AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))

          elif self.charge_timi_1 == '<=' and self.charge_timi_2 == '<=':
```

```
          invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s <= '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))

        elif self.charge_timi_1 == '>=' and self.charge_timi_2 == '<=':
          invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s <= '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))
        elif self.charge_timi_1 == '<=' and self.charge_timi_2 == '>=':
          invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s >= '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))

        elif self.charge_timi_1 == '>=' and self.charge_timi_2 == '==':
          invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s == '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))

        elif self.charge_timi_1 == '==' and self.charge_timi_2 == '>=':
          invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s >= '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))

        elif self.charge_timi_1 == '<=' and self.charge_timi_2 == '==' :
          invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s == '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))

        elif self.charge_timi_1 == '==' and self.charge_timi_2 == '<=' :
          invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s <= '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))
        else:
          invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s == '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,float(a_4)  ,a_5,int(a_6)))

       elif a_1 == 'KW' and self.charge_timi_1 == '>=':
         invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s = '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,a_4 ,a_5,a_6))

       elif a_1 == 'KW' and self.charge_timi_1 == '<=':
         invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s = '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,a_4 ,a_5,a_6))

       elif a_1 == 'KW' and self.charge_timi_1 == '==':
         invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s = '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,a_4,a_5,a_6 ))



       elif a_1 == 'totalPrice' and self.charge_timi_2 == '>=':
         invoicesSql = ("SELECT * FROM Charge WHERE  %s >= '%s' AND %s = '%s'
AND %s = '%s'"% (a_1,float(a_2),a_3,a_4 ,a_5,a_6))
```

```
        elif a_1 == 'totalPrice' and self.charge_timi_2 == '<=':
          invoicesSql = ("SELECT * FROM Charge WHERE  %s <= '%s' AND %s = '%s' AND %s = '%s'"% (a_1,float(a_2),a_3,a_4 ,a_5,a_6))

        elif a_1 == 'totalPrice' and self.charge_timi_2 == '==':
          invoicesSql = ("SELECT * FROM Charge WHERE  %s == '%s' AND %s = '%s' AND %s = '%s'"% (a_1,float(a_2),a_3,a_4, a_5,a_6))

        else:

          invoicesSql = (""""SELECT * FROM Charge WHERE SUBSTR(chargeDate,6,2)="%.2i" AND %s = '%s' AND %s = '%s'""" % (int(a_2),a_3,int(a_4),a_5,int(a_6)))
        c.execute(invoicesSql)
        idata = c.fetchall()

        rows = len(idata)
        columns = len(idata[0])

        self.ui.table_cha.setRowCount(rows)
        self.ui.table_cha.setColumnCount(columns)

        self.ui.table_cha.setHorizontalHeaderLabels(['Charge ID', 'Charge Date','KW','Total Price','Customer ID'])
        for i, row in enumerate(idata):
          for j, col in enumerate(row):
            item = QTableWidgetItem(str(col))
            self.ui.table_cha.setItem(i, j, item)




    except:
      self.not_exist()

  def Cha_insert(self):
      try:
        self.cha_timi_0 = self.ui.lineEdit_35.text()

        if self.cha_timi_0 == '' :
          self.messege_ins()
        elif type(self.cha_timi_0) == 'str':
          self.messege_ins_8()
        else:
          self.cha_inst()
```

```
      except:
        self.messege_ins_5()

  def cha_inst(self):
    if self.ui.radioButton_5.isChecked():

      c.execute("SELECT max(consumeId) FROM Consumption WHERE customerId == %s "%(self.cha_timi_0) )
      for item in c.fetchall():
          self.ch_id = item[0]

      c.execute("SELECT endDate,totalKW FROM Consumption WHERE consumeId == %s "%(self.ch_id))
      for item in c.fetchall():
          self.ch_date = item[0]
          self.ch_kw = item[1]
      c.execute("SELECT pricePerKW FROM Customer WHERE customerId == %s "%(self.cha_timi_0) )
      for i in c.fetchall():
        cc_3 = i[0]
      t_2 = float(cc_3)
      self.cha_t_2 = float(self.ch_kw*t_2)
      self.cha_times = ((self.ch_date,self.ch_kw,self.cha_t_2,self.cha_timi_0))
##    print(self.cha_times)




    elif self.ui.radioButton_3.isChecked():

      c.execute("""SELECT SUM(totalKW) FROM Consumption WHERE endDate BETWEEN datetime('now', '-31 day') AND datetime('now', '-1 day') AND customerId == %s """%(self.cha_timi_0))
      for item in c.fetchall():
          self.ch_kw = item[0]

      c.execute("SELECT max(consumeId) FROM Consumption WHERE customerId == %s "%(self.cha_timi_0) )
      for item in c.fetchall():
          self.ch_id = item[0]

      c.execute("SELECT endDate  FROM Consumption WHERE consumeId == %s "%(self.ch_id))
      for item in c.fetchall():
```

```
        self.ch_date = item[0]


    c.execute("SELECT pricePerKW FROM Customer WHERE customerId == %s "%(self.cha_timi_0) )
    for i in c.fetchall():
      cc_3 = i[0]
    t_2 = float(cc_3)
    self.cha_t_2 = float(self.ch_kw*t_2)

    self.cha_times = ((self.ch_date, self.ch_kw, self.cha_t_2, self.cha_timi_0))
   elif self.ui.radioButton_4.isChecked():
    lista = []
    c.execute("""SELECT SUM(totalKW),consumeId FROM Consumption WHERE endDate BETWEEN datetime('now', '-3 month') AND datetime('now', '-1 day') AND customerId == %s """%(self.cha_timi_0))
    for item in c.fetchall():
       self.ch_kw = item[0]
    c.execute("SELECT max(consumeId) FROM Consumption WHERE customerId == %s "%(self.cha_timi_0) )
    for item in c.fetchall():
       self.ch_id = item[0]

    c.execute("SELECT endDate  FROM Consumption WHERE consumeId == %s "%(self.ch_id))
    for item in c.fetchall():
       self.ch_date = item[0]



    c.execute("SELECT pricePerKW FROM Customer WHERE customerId == %s "%(self.cha_timi_0) )
    for i in c.fetchall():
      cc_3 = i[0]
    t_2 = float(cc_3)
    self.cha_t_2 = float(self.ch_kw*t_2)


    self.cha_times = ((self.ch_date,self.ch_kw,self.cha_t_2,self.cha_timi_0))
   else:

    self.messege_ins_5()
```

```
    c.execute("INSERT OR IGNORE INTO Charge (chargeDate,KW,totalPrice,customerId)VALUES(?,?,?,?)",self.cha_times)
    db.commit()
    self.runCharge()
    self.messege_ins_6()

  def Cha_delete(self):
    try:
      id_ = int( self.ui.table_cha.currentItem().text())
      cha_tp = type(id_)
      if (cha_tp == int) == True:
        if self.ui.table_cha.currentItem().text()=='':
          pass
        else:
          buttonReply = QMessageBox.question(self,
                'The document has been modified.', "Do you want to save your changes?.", QMessageBox.Cancel ,QMessageBox.Save)
          if buttonReply == QMessageBox.Save:
            cha_select = self.ui.table_cha.currentItem().text()

            c.execute("DELETE  FROM Charge WHERE chargeId =='%s'" % cha_select )
            db.commit()
            self.runCustomer()
            self.runConsumption()
            self.runCharge()

            self.del_mess_3 = cha_select
            self.del_messege_ins_3()
          else:
            pass

      else:
        self.messege_ins_8()
    except:
      pass
  def Cha_update(self):

    try:
      cha_t_up = float(self.ui.lineEdit_2.text())
      cha_select = int(self.ui.table_cha.currentItem().text())
      c.execute("UPDATE Charge SET totalPrice = '%s' WHERE chargeId = %s "%(cha_t_up,cha_select))
```

```
      db.commit()
      self.runCharge()
      self.messege_ins_12()
    except:
      pass
  def messege_ins(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "Maybe edit line was empty. Check it and try again. ", QMessageBox.Ok)

  def messege_ins_2(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "This value (Price Per KW) we need to be a number. ", QMessageBox.Ok)

  def messege_ins_3(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "This values (KW,ToTal Price and Customer ID) we need to be a number. ",
QMessageBox.Ok)

  def messege_ins_4(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "This values (TotalKW and Customer ID )we need to be a number. ",
QMessageBox.Ok)

  def messege_ins_5(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "Your add was failed. Try again. ", QMessageBox.Ok)

  def messege_ins_6(self):
    QMessageBox.about(self, "Insert Value",
      "<p> Your add it was successful.</p>")

QMessageBox.Ok)

  def messege_ins_8(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "Failed.\nTry to insert an integer number. ", QMessageBox.Ok)

  def messege_ins_9(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "Failed.\nTry to insert a str item. ", QMessageBox.Ok)
  def messege_ins_11(self):
    buttonReply = QMessageBox.critical(self,
          'Error', "Please check in one radio button. ", QMessageBox.Ok)

  def messege_ins_12(self):
    QMessageBox.about(self, "Update",
                "<p>Update it was Successful.</p>")
```

```
  def messege_ins_13(self):
    buttonReply = QMessageBox.critical(self,
            'Fail', "Fail export excel  file.\n Try to insert integer number", QMessageBox.Ok)

  def messege_ins_14(self):

    QMessageBox.about(self, "Export",
                      "<p>The export for Customer {0} it was
Successful.</p>".format(self.sentID))



  def del_messege_ins_1(self):

    QMessageBox.about(self, "Delete",
                      "<p>The delete rows for CustomerId = {0} it was
Successful.</p>".format(self.del_mess_1))


  def del_messege_ins_2(self):

    QMessageBox.about(self, "Delete",
                      "<p>The delete row for ConsumptionId = {0} it was
Successful.</p>".format(self.del_mess_2))

  def del_messege_ins_3(self):

    QMessageBox.about(self, "Delete",
                      "<p>The delete row for Charge = {0} it was
Successful.</p>".format(self.del_mess_3))



  def Action_export(self):
     buttonReply = QMessageBox.question(self,'Export',
                                        "If you export excel file with same Id will be replaced the old
file.",
                                        QMessageBox.Yes | QMessageBox.No)
     if buttonReply == QMessageBox.Yes :
       self.export()
     else:
       pass


  def exit_Action(self):
     buttonReply = QMessageBox.question(self,'Exit!',
```

```
                                "Did you really want to leave?",
                                QMessageBox.Yes | QMessageBox.No)
    if buttonReply == QMessageBox.Yes :
      sys.exit(app.exec_())
    else:
      pass

  def link(self):
##    webbrowser.open('https://gmoustakidis1993.wixsite.com/thandersys')
    webbrowser.open('https://gmoustakidis1993.wixsite.com/clover-software')
  def about(self):

    QMessageBox.about(self, "About CRM",
                      "<p>   Customer relationship management <b>(CRM)</b>"
                      " is a term that refers to practices, strategies and "
                      "technologies that companies use to manage and analyze"
                      "customer interactions and data throughout the customer"
                      "lifecycle, with the goal of improving business "
                      "relationships with customers, assisting in customer"
                      "retention and driving sales growth. CRM systems are "
                      "designed to compile information on customers across "
                      "different channels -- or points of contact between the"
                      "customer and the company -- which could include the "
                      "company's website, telephone, live chat, direct mail, marketing"
                      "materials and social media. CRM systems can also give "
                      "customer-facing staff detailed information on "
                      "customers' personal information, purchase history, buying"
                      "preferences and concerns.</p>")


  def about_unq(self):
    QMessageBox.about(self, "Unique",
                      "If the insert row you set it's same with someone else row .The system will be not accept this row.\nUnique Combination:\n\nArea (areaName)\nCustomer (lastName, phone)\nConsumption (startDate, endDate, totalKW, customerId)\nCharge (chargeDate, KW).")
  def about_search(self):
    QMessageBox.about(self, "Search",
                      "If you search a date, click on Searching Date button to show you all values between this date.If you also click on radio buttons.\nThe system will be show you all values <=  endDate and six monts later or one year later from the date you set.")

  def not_exist(self):
    buttonReply = QMessageBox.critical(self,
    'Not Exist', "This searching item is not exist.", QMessageBox.Ok)
```

```
if __name__ == "__main__":
  app = QApplication(sys.argv)
  app.setStyle('Fusion')
  window = MyApp()
  window.show()
  sys.exit(app.exec_())
```

ΠΑΡΑΡΤΗΜΑ Β

Ο κώδικας QT5 για τη γραφική διεπαφή της εφαρμογής λογισμικού clover software

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'table.ui'
#
# Created by: PyQt5 UI code generator 5.8.1
#
# WARNING! All changes made in this file will be lost!

from PyQt5 import QtCore, QtGui, QtWidgets

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(1364, 818)
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap("icons/logo.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        Form.setWindowIcon(icon)
        Form.setStyleSheet("")
        self.centralwidget = QtWidgets.QWidget(Form)
        self.centralwidget.setObjectName("centralwidget")
        self.gridLayout_2 = QtWidgets.QGridLayout(self.centralwidget)
        self.gridLayout_2.setObjectName("gridLayout_2")
        self.tab = QtWidgets.QTabWidget(self.centralwidget)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.tab.setFont(font)
        self.tab.setObjectName("tab")
        self.tab_4 = QtWidgets.QWidget()
        self.tab_4.setObjectName("tab_4")
        self.gridLayout_15 = QtWidgets.QGridLayout(self.tab_4)
        self.gridLayout_15.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_15.setObjectName("gridLayout_15")
        self.horizontalLayout_31 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_31.setObjectName("horizontalLayout_31")
        self.label_51 = QtWidgets.QLabel(self.tab_4)
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_51.setFont(font)
        self.label_51.setStyleSheet("color:#006fc6")
        self.label_51.setObjectName("label_51")
        self.horizontalLayout_31.addWidget(self.label_51)
```

```
        spacerItem = QtWidgets.QSpacerItem(0, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_31.addItem(spacerItem)
        self.label_47 = QtWidgets.QLabel(self.tab_4)
        font = QtGui.QFont()
        font.setPointSize(24)
        font.setBold(True)
        font.setWeight(75)
        self.label_47.setFont(font)
        self.label_47.setStyleSheet("color:#0078d7")
        self.label_47.setAlignment(QtCore.Qt.AlignCenter)
        self.label_47.setObjectName("label_47")
        self.horizontalLayout_31.addWidget(self.label_47)
        spacerItem1 = QtWidgets.QSpacerItem(550, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_31.addItem(spacerItem1)
        self.gridLayout_15.addLayout(self.horizontalLayout_31, 0, 0, 1, 2)
        self.widget_9 = QtWidgets.QWidget(self.tab_4)
        self.widget_9.setObjectName("widget_9")
        self.gridLayout_13 = QtWidgets.QGridLayout(self.widget_9)
        self.gridLayout_13.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_13.setObjectName("gridLayout_13")
        self.horizontalLayout_43 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_43.setObjectName("horizontalLayout_43")
        self.verticalLayout_21 = QtWidgets.QVBoxLayout()
        self.verticalLayout_21.setObjectName("verticalLayout_21")
        self.horizontalLayout_6 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_6.setObjectName("horizontalLayout_6")
        self.label_4 = QtWidgets.QLabel(self.widget_9)
        self.label_4.setObjectName("label_4")
        self.horizontalLayout_6.addWidget(self.label_4)
        self.lineEdit_27 = QtWidgets.QLineEdit(self.widget_9)
        self.lineEdit_27.setObjectName("lineEdit_27")
        self.horizontalLayout_6.addWidget(self.lineEdit_27)
        self.verticalLayout_21.addLayout(self.horizontalLayout_6)
        self.horizontalLayout_39 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_39.setObjectName("horizontalLayout_39")
        self.label_31 = QtWidgets.QLabel(self.widget_9)
        self.label_31.setObjectName("label_31")
        self.horizontalLayout_39.addWidget(self.label_31)
        self.lineEdit_28 = QtWidgets.QLineEdit(self.widget_9)
        self.lineEdit_28.setObjectName("lineEdit_28")
        self.horizontalLayout_39.addWidget(self.lineEdit_28)
        self.verticalLayout_21.addLayout(self.horizontalLayout_39)
        self.horizontalLayout_40 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_40.setObjectName("horizontalLayout_40")
        self.label_32 = QtWidgets.QLabel(self.widget_9)
```

```
        self.label_32.setObjectName("label_32")
        self.horizontalLayout_40.addWidget(self.label_32)
        self.lineEdit_32 = QtWidgets.QLineEdit(self.widget_9)
        self.lineEdit_32.setObjectName("lineEdit_32")
        self.horizontalLayout_40.addWidget(self.lineEdit_32)
        self.verticalLayout_21.addLayout(self.horizontalLayout_40)
        self.horizontalLayout_43.addLayout(self.verticalLayout_21)
        self.verticalLayout_22 = QtWidgets.QVBoxLayout()
        self.verticalLayout_22.setObjectName("verticalLayout_22")
        spacerItem2 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_22.addItem(spacerItem2)
        self.home_cl = QtWidgets.QPushButton(self.widget_9)
        icon1 = QtGui.QIcon()
        icon1.addPixmap(QtGui.QPixmap("icons/clear.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.home_cl.setIcon(icon1)
        self.home_cl.setObjectName("home_cl")
        self.verticalLayout_22.addWidget(self.home_cl)
        self.btn_h_gr_2 = QtWidgets.QPushButton(self.widget_9)
        icon2 = QtGui.QIcon()
        icon2.addPixmap(QtGui.QPixmap("icons/graph.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_h_gr_2.setIcon(icon2)
        self.btn_h_gr_2.setObjectName("btn_h_gr_2")
        self.verticalLayout_22.addWidget(self.btn_h_gr_2)
        self.horizontalLayout_43.addLayout(self.verticalLayout_22)
        self.gridLayout_13.addLayout(self.horizontalLayout_43, 1, 0, 1, 1)
        self.label_52 = QtWidgets.QLabel(self.widget_9)
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_52.setFont(font)
        self.label_52.setStyleSheet("color:#006fc6")
        self.label_52.setObjectName("label_52")
        self.gridLayout_13.addWidget(self.label_52, 3, 0, 1, 1)
        self.label_48 = QtWidgets.QLabel(self.widget_9)
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_48.setFont(font)
        self.label_48.setObjectName("label_48")
        self.gridLayout_13.addWidget(self.label_48, 4, 0, 1, 1)
        self.label_46 = QtWidgets.QLabel(self.widget_9)
        font = QtGui.QFont()
```

```
        font.setPointSize(11)
        font.setBold(False)
        font.setWeight(50)
        self.label_46.setFont(font)
        self.label_46.setObjectName("label_46")
        self.gridLayout_13.addWidget(self.label_46, 0, 0, 1, 1)
        self.horizontalLayout_30 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_30.setObjectName("horizontalLayout_30")
        self.label_34 = QtWidgets.QLabel(self.widget_9)
        self.label_34.setObjectName("label_34")
        self.horizontalLayout_30.addWidget(self.label_34)
        self.lineEdit_33 = QtWidgets.QLineEdit(self.widget_9)
        self.lineEdit_33.setObjectName("lineEdit_33")
        self.horizontalLayout_30.addWidget(self.lineEdit_33)
        self.btn_h_gr = QtWidgets.QPushButton(self.widget_9)
        self.btn_h_gr.setIcon(icon2)
        self.btn_h_gr.setObjectName("btn_h_gr")
        self.horizontalLayout_30.addWidget(self.btn_h_gr)
        self.gridLayout_13.addLayout(self.horizontalLayout_30, 5, 0, 1, 1)
        self.label_54 = QtWidgets.QLabel(self.widget_9)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Maximum,
QtWidgets.QSizePolicy.Maximum)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.label_54.sizePolicy().hasHeightForWidth())
        self.label_54.setSizePolicy(sizePolicy)
        self.label_54.setMinimumSize(QtCore.QSize(0, 0))
        self.label_54.setMaximumSize(QtCore.QSize(662, 16777215))
        self.label_54.setText("")
        self.label_54.setPixmap(QtGui.QPixmap("icons/graph_home.png"))
        self.label_54.setObjectName("label_54")
        self.gridLayout_13.addWidget(self.label_54, 6, 0, 1, 1)
        spacerItem3 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.gridLayout_13.addItem(spacerItem3, 2, 0, 1, 1)
        self.gridLayout_15.addWidget(self.widget_9, 1, 0, 1, 1)
        self.widget_10 = QtWidgets.QWidget(self.tab_4)
        self.widget_10.setObjectName("widget_10")
        self.gridLayout_14 = QtWidgets.QGridLayout(self.widget_10)
        self.gridLayout_14.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_14.setObjectName("gridLayout_14")
        self.horizontalLayout_19 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_19.setObjectName("horizontalLayout_19")
        self.label_45 = QtWidgets.QLabel(self.widget_10)
        font = QtGui.QFont()
        font.setPointSize(12)
        self.label_45.setFont(font)
```

```
        self.label_45.setObjectName("label_45")
        self.horizontalLayout_19.addWidget(self.label_45)
        self.label_50 = QtWidgets.QLabel(self.widget_10)
        font = QtGui.QFont()
        font.setPointSize(12)
        font.setBold(True)
        font.setWeight(75)
        self.label_50.setFont(font)
        self.label_50.setStyleSheet("color:#4cd964")

self.label_50.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.AlignT
op)
        self.label_50.setObjectName("label_50")
        self.horizontalLayout_19.addWidget(self.label_50)
        self.gridLayout_14.addLayout(self.horizontalLayout_19, 1, 0, 2, 2)
        self.label_49 = QtWidgets.QLabel(self.widget_10)
        self.label_49.setMaximumSize(QtCore.QSize(108, 103))
        self.label_49.setText("")
        self.label_49.setPixmap(QtGui.QPixmap("icons/QRcode.png"))
        self.label_49.setScaledContents(True)
        self.label_49.setObjectName("label_49")
        self.gridLayout_14.addWidget(self.label_49, 2, 1, 2, 1)
        self.label_53 = QtWidgets.QLabel(self.widget_10)
        font = QtGui.QFont()
        font.setPointSize(12)
        font.setBold(True)
        font.setWeight(75)
        self.label_53.setFont(font)
        self.label_53.setStyleSheet("color:#006fc6")
        self.label_53.setObjectName("label_53")
        self.gridLayout_14.addWidget(self.label_53, 3, 0, 1, 1)
        self.calendarWidget = QtWidgets.QCalendarWidget(self.widget_10)
        self.calendarWidget.setLocale(QtCore.QLocale(QtCore.QLocale.English,
QtCore.QLocale.UnitedStates))
        self.calendarWidget.setGridVisible(True)
        self.calendarWidget.setSelectionMode(QtWidgets.QCalendarWidget.SingleSelection)
        self.calendarWidget.setNavigationBarVisible(True)
        self.calendarWidget.setDateEditEnabled(True)
        self.calendarWidget.setObjectName("calendarWidget")
        self.gridLayout_14.addWidget(self.calendarWidget, 0, 0, 1, 2)
        self.gridLayout_15.addWidget(self.widget_10, 1, 1, 1, 1)
        icon3 = QtGui.QIcon()
        icon3.addPixmap(QtGui.QPixmap("icons/home.png"),                QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.tab.addTab(self.tab_4, icon3, "")
        self.tab_area = QtWidgets.QWidget()
        self.tab_area.setObjectName("tab_area")
```

```
        self.gridLayout_11 = QtWidgets.QGridLayout(self.tab_area)
        self.gridLayout_11.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_11.setObjectName("gridLayout_11")
        self.horizontalLayout_27 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_27.setObjectName("horizontalLayout_27")
        self.verticalLayout_30 = QtWidgets.QVBoxLayout()
        self.verticalLayout_30.setContentsMargins(-1, -1, -1, 0)
        self.verticalLayout_30.setObjectName("verticalLayout_30")
        self.verticalLayout_7 = QtWidgets.QVBoxLayout()
        self.verticalLayout_7.setObjectName("verticalLayout_7")
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout")
        self.areaId = QtWidgets.QLabel(self.tab_area)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.areaId.setFont(font)
        self.areaId.setObjectName("areaId")
        self.horizontalLayout.addWidget(self.areaId)
        self.el_id = QtWidgets.QLineEdit(self.tab_area)
        self.el_id.setObjectName("el_id")
        self.horizontalLayout.addWidget(self.el_id)
        self.verticalLayout_7.addLayout(self.horizontalLayout)
        self.horizontalLayout_22 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_22.setObjectName("horizontalLayout_22")
        self.areaName = QtWidgets.QLabel(self.tab_area)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.areaName.setFont(font)
        self.areaName.setObjectName("areaName")
        self.horizontalLayout_22.addWidget(self.areaName)
        self.el_name = QtWidgets.QLineEdit(self.tab_area)
        self.el_name.setObjectName("el_name")
        self.horizontalLayout_22.addWidget(self.el_name)
        self.verticalLayout_7.addLayout(self.horizontalLayout_22)
        self.verticalLayout_30.addLayout(self.verticalLayout_7)
        self.horizontalLayout_4 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_4.setObjectName("horizontalLayout_4")
        self.label_2 = QtWidgets.QLabel(self.tab_area)
        self.label_2.setStyleSheet("color :#ff3333")
        self.label_2.setObjectName("label_2")
        self.horizontalLayout_4.addWidget(self.label_2)
        spacerItem4 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_4.addItem(spacerItem4)
        self.verticalLayout_30.addLayout(self.horizontalLayout_4)
        spacerItem5 = QtWidgets.QSpacerItem(20, 0, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
```

```
        self.verticalLayout_30.addItem(spacerItem5)
        self.horizontalLayout_27.addLayout(self.verticalLayout_30)
        self.verticalLayout_13 = QtWidgets.QVBoxLayout()
        self.verticalLayout_13.setObjectName("verticalLayout_13")
        self.cl_4 = QtWidgets.QPushButton(self.tab_area)
        self.cl_4.setIcon(icon1)
        self.cl_4.setObjectName("cl_4")
        self.verticalLayout_13.addWidget(self.cl_4)
        self.btn_area = QtWidgets.QPushButton(self.tab_area)
        icon4 = QtGui.QIcon()
        icon4.addPixmap(QtGui.QPixmap("icons/search.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_area.setIcon(icon4)
        self.btn_area.setObjectName("btn_area")
        self.verticalLayout_13.addWidget(self.btn_area)
        self.A_add_btn = QtWidgets.QPushButton(self.tab_area)
        icon5 = QtGui.QIcon()
        icon5.addPixmap(QtGui.QPixmap("icons/add.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.A_add_btn.setIcon(icon5)
        self.A_add_btn.setObjectName("A_add_btn")
        self.verticalLayout_13.addWidget(self.A_add_btn)
        self.up_ar_btn = QtWidgets.QPushButton(self.tab_area)
        icon6 = QtGui.QIcon()
        icon6.addPixmap(QtGui.QPixmap("icons/update.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.up_ar_btn.setIcon(icon6)
        self.up_ar_btn.setObjectName("up_ar_btn")
        self.verticalLayout_13.addWidget(self.up_ar_btn)
        spacerItem6 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_13.addItem(spacerItem6)
        self.horizontalLayout_27.addLayout(self.verticalLayout_13)
        self.gridLayout_11.addLayout(self.horizontalLayout_27, 0, 0, 1, 1)
        self.horizontalLayout_20 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_20.setObjectName("horizontalLayout_20")
        self.verticalLayout_24 = QtWidgets.QVBoxLayout()
        self.verticalLayout_24.setObjectName("verticalLayout_24")
        self.widget_7 = QtWidgets.QWidget(self.tab_area)
        self.widget_7.setObjectName("widget_7")
        self.gridLayout_3 = QtWidgets.QGridLayout(self.widget_7)
        self.gridLayout_3.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_3.setObjectName("gridLayout_3")
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")
        self.label_39 = QtWidgets.QLabel(self.widget_7)
        self.label_39.setObjectName("label_39")
```

```
        self.horizontalLayout_2.addWidget(self.label_39)
        self.lineEdit_10 = QtWidgets.QLineEdit(self.widget_7)
        self.lineEdit_10.setObjectName("lineEdit_10")
        self.horizontalLayout_2.addWidget(self.lineEdit_10)
        self.a_btn_up = QtWidgets.QPushButton(self.widget_7)
        self.a_btn_up.setMinimumSize(QtCore.QSize(120, 0))
        icon7 = QtGui.QIcon()
        icon7.addPixmap(QtGui.QPixmap("icons/save.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.a_btn_up.setIcon(icon7)
        self.a_btn_up.setObjectName("a_btn_up")
        self.horizontalLayout_2.addWidget(self.a_btn_up)
        self.pushButton = QtWidgets.QPushButton(self.widget_7)
        self.pushButton.setMinimumSize(QtCore.QSize(120, 0))
        icon8 = QtGui.QIcon()
        icon8.addPixmap(QtGui.QPixmap("icons/close.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.pushButton.setIcon(icon8)
        self.pushButton.setObjectName("pushButton")
        self.horizontalLayout_2.addWidget(self.pushButton)
        self.gridLayout_3.addLayout(self.horizontalLayout_2, 0, 0, 1, 1)
        self.verticalLayout_24.addWidget(self.widget_7)
        self.horizontalLayout_33 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_33.setObjectName("horizontalLayout_33")
        self.back_1 = QtWidgets.QPushButton(self.tab_area)
        self.back_1.setMinimumSize(QtCore.QSize(28, 24))
        self.back_1.setMaximumSize(QtCore.QSize(28, 24))
        self.back_1.setText("")
        icon9 = QtGui.QIcon()
        icon9.addPixmap(QtGui.QPixmap("icons/back.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.back_1.setIcon(icon9)
        self.back_1.setObjectName("back_1")
        self.horizontalLayout_33.addWidget(self.back_1)
        spacerItem7 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_33.addItem(spacerItem7)
        self.verticalLayout_24.addLayout(self.horizontalLayout_33)
        self.table = QtWidgets.QTableWidget(self.tab_area)
        self.table.setVerticalScrollBarPolicy(QtCore.Qt.ScrollBarAlwaysOn)
        self.table.setSizeAdjustPolicy(QtWidgets.QAbstractScrollArea.AdjustIgnored)
        self.table.setEditTriggers(QtWidgets.QAbstractItemView.NoEditTriggers)
        self.table.setObjectName("table")
        self.table.setColumnCount(0)
        self.table.setRowCount(0)
        self.verticalLayout_24.addWidget(self.table)
        self.horizontalLayout_20.addLayout(self.verticalLayout_24)
```

```
        self.widget = QtWidgets.QWidget(self.tab_area)
        self.widget.setEnabled(True)
        self.widget.setObjectName("widget")
        self.gridLayout = QtWidgets.QGridLayout(self.widget)
        self.gridLayout.setContentsMargins(0, 0, 0, 0)
        self.gridLayout.setObjectName("gridLayout")
        self.horizontalLayout_8 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_8.setObjectName("horizontalLayout_8")
        self.A_sent_btn = QtWidgets.QPushButton(self.widget)
        self.A_sent_btn.setIcon(icon7)
        self.A_sent_btn.setObjectName("A_sent_btn")
        self.horizontalLayout_8.addWidget(self.A_sent_btn)
        self.A_cl_btn = QtWidgets.QPushButton(self.widget)
        self.A_cl_btn.setIcon(icon8)
        self.A_cl_btn.setObjectName("A_cl_btn")
        self.horizontalLayout_8.addWidget(self.A_cl_btn)
        self.gridLayout.addLayout(self.horizontalLayout_8, 2, 0, 1, 1)
        spacerItem8 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.gridLayout.addItem(spacerItem8, 3, 0, 1, 1)
        self.verticalLayout_12 = QtWidgets.QVBoxLayout()
        self.verticalLayout_12.setObjectName("verticalLayout_12")
        self.horizontalLayout_9 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_9.setObjectName("horizontalLayout_9")
        self.label_21 = QtWidgets.QLabel(self.widget)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_21.setFont(font)
        self.label_21.setObjectName("label_21")
        self.horizontalLayout_9.addWidget(self.label_21)
        self.lineEdit_20 = QtWidgets.QLineEdit(self.widget)
        self.lineEdit_20.setObjectName("lineEdit_20")
        self.horizontalLayout_9.addWidget(self.lineEdit_20)
        self.verticalLayout_12.addLayout(self.horizontalLayout_9)
        self.gridLayout.addLayout(self.verticalLayout_12, 1, 0, 1, 1)
        self.label_58 = QtWidgets.QLabel(self.widget)
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_58.setFont(font)
        self.label_58.setObjectName("label_58")
        self.gridLayout.addWidget(self.label_58, 0, 0, 1, 1)
        self.horizontalLayout_20.addWidget(self.widget)
        self.gridLayout_11.addLayout(self.horizontalLayout_20, 1, 0, 1, 1)
        icon10 = QtGui.QIcon()
```

```
        icon10.addPixmap(QtGui.QPixmap("icons/area.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.tab.addTab(self.tab_area, icon10, "")
        self.tab_customer = QtWidgets.QWidget()
        self.tab_customer.setObjectName("tab_customer")
        self.gridLayout_12 = QtWidgets.QGridLayout(self.tab_customer)
        self.gridLayout_12.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_12.setObjectName("gridLayout_12")
        self.verticalLayout_26 = QtWidgets.QVBoxLayout()
        self.verticalLayout_26.setObjectName("verticalLayout_26")
        self.horizontalLayout_24 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_24.setObjectName("horizontalLayout_24")
        self.verticalLayout_3 = QtWidgets.QVBoxLayout()
        self.verticalLayout_3.setObjectName("verticalLayout_3")
        self.label_7 = QtWidgets.QLabel(self.tab_customer)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_7.setFont(font)
        self.label_7.setObjectName("label_7")
        self.verticalLayout_3.addWidget(self.label_7)
        self.label = QtWidgets.QLabel(self.tab_customer)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label.setFont(font)
        self.label.setObjectName("label")
        self.verticalLayout_3.addWidget(self.label)
        self.label_5 = QtWidgets.QLabel(self.tab_customer)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_5.setFont(font)
        self.label_5.setObjectName("label_5")
        self.verticalLayout_3.addWidget(self.label_5)
        self.label_6 = QtWidgets.QLabel(self.tab_customer)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_6.setFont(font)
        self.label_6.setObjectName("label_6")
        self.verticalLayout_3.addWidget(self.label_6)
        self.label_8 = QtWidgets.QLabel(self.tab_customer)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_8.setFont(font)
        self.label_8.setObjectName("label_8")
        self.verticalLayout_3.addWidget(self.label_8)
        self.horizontalLayout_24.addLayout(self.verticalLayout_3)
        self.verticalLayout_5 = QtWidgets.QVBoxLayout()
        self.verticalLayout_5.setObjectName("verticalLayout_5")
```

```
        self.horizontalLayout_18 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_18.setObjectName("horizontalLayout_18")
        self.comboBox_2 = QtWidgets.QComboBox(self.tab_customer)
        self.comboBox_2.setObjectName("comboBox_2")
        self.comboBox_2.addItem("")
        self.comboBox_2.setItemText(0, "")
        self.comboBox_2.addItem("")
        self.comboBox_2.addItem("")
        self.comboBox_2.addItem("")
        self.horizontalLayout_18.addWidget(self.comboBox_2)
        self.lineEdit_7 = QtWidgets.QLineEdit(self.tab_customer)
        self.lineEdit_7.setObjectName("lineEdit_7")
        self.horizontalLayout_18.addWidget(self.lineEdit_7)
        self.verticalLayout_5.addLayout(self.horizontalLayout_18)
        self.lineEdit = QtWidgets.QLineEdit(self.tab_customer)
        self.lineEdit.setObjectName("lineEdit")
        self.verticalLayout_5.addWidget(self.lineEdit)
        self.lineEdit_5 = QtWidgets.QLineEdit(self.tab_customer)
        self.lineEdit_5.setObjectName("lineEdit_5")
        self.verticalLayout_5.addWidget(self.lineEdit_5)
        self.lineEdit_6 = QtWidgets.QLineEdit(self.tab_customer)
        self.lineEdit_6.setObjectName("lineEdit_6")
        self.verticalLayout_5.addWidget(self.lineEdit_6)
        self.lineEdit_8 = QtWidgets.QLineEdit(self.tab_customer)
        self.lineEdit_8.setObjectName("lineEdit_8")
        self.verticalLayout_5.addWidget(self.lineEdit_8)
        self.horizontalLayout_24.addLayout(self.verticalLayout_5)
        self.verticalLayout_26.addLayout(self.horizontalLayout_24)
        self.label_11 = QtWidgets.QLabel(self.tab_customer)
        font = QtGui.QFont()
        font.setPointSize(8)
        self.label_11.setFont(font)
        self.label_11.setStyleSheet("color :#ff3333")
        self.label_11.setObjectName("label_11")
        self.verticalLayout_26.addWidget(self.label_11)
        spacerItem9 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_26.addItem(spacerItem9)
        self.gridLayout_12.addLayout(self.verticalLayout_26, 0, 0, 1, 1)
        self.verticalLayout_6 = QtWidgets.QVBoxLayout()
        self.verticalLayout_6.setObjectName("verticalLayout_6")
        self.cl_3 = QtWidgets.QPushButton(self.tab_customer)
        self.cl_3.setIcon(icon1)
        self.cl_3.setObjectName("cl_3")
        self.verticalLayout_6.addWidget(self.cl_3)
        self.btn_cust = QtWidgets.QPushButton(self.tab_customer)
        self.btn_cust.setIcon(icon4)
```

```
        self.btn_cust.setObjectName("btn_cust")
        self.verticalLayout_6.addWidget(self.btn_cust)
        self.cust_btn_update = QtWidgets.QPushButton(self.tab_customer)
        self.cust_btn_update.setIcon(icon6)
        self.cust_btn_update.setObjectName("cust_btn_update")
        self.verticalLayout_6.addWidget(self.cust_btn_update)
        self.Cu_add_btn = QtWidgets.QPushButton(self.tab_customer)
        self.Cu_add_btn.setIcon(icon5)
        self.Cu_add_btn.setObjectName("Cu_add_btn")
        self.verticalLayout_6.addWidget(self.Cu_add_btn)
        self.Cust_del_btn = QtWidgets.QPushButton(self.tab_customer)
        icon11 = QtGui.QIcon()
        icon11.addPixmap(QtGui.QPixmap("icons/delete.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.Cust_del_btn.setIcon(icon11)
        self.Cust_del_btn.setObjectName("Cust_del_btn")
        self.verticalLayout_6.addWidget(self.Cust_del_btn)
        spacerItem10 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_6.addItem(spacerItem10)
        self.gridLayout_12.addLayout(self.verticalLayout_6, 0, 1, 1, 1)
        self.horizontalLayout_12 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_12.setObjectName("horizontalLayout_12")
        self.verticalLayout_16 = QtWidgets.QVBoxLayout()
        self.verticalLayout_16.setObjectName("verticalLayout_16")
        self.horizontalLayout_23 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_23.setObjectName("horizontalLayout_23")
        self.verticalLayout_16.addLayout(self.horizontalLayout_23)
        self.widget_8 = QtWidgets.QWidget(self.tab_customer)
        self.widget_8.setObjectName("widget_8")
        self.gridLayout_8 = QtWidgets.QGridLayout(self.widget_8)
        self.gridLayout_8.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_8.setObjectName("gridLayout_8")
        self.horizontalLayout_25 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_25.setObjectName("horizontalLayout_25")
        self.label_43 = QtWidgets.QLabel(self.widget_8)
        self.label_43.setObjectName("label_43")
        self.horizontalLayout_25.addWidget(self.label_43)
        self.lineEdit_4 = QtWidgets.QLineEdit(self.widget_8)
        self.lineEdit_4.setObjectName("lineEdit_4")
        self.horizontalLayout_25.addWidget(self.lineEdit_4)
        self.cu_btn_up = QtWidgets.QPushButton(self.widget_8)
        self.cu_btn_up.setMinimumSize(QtCore.QSize(120, 0))
        self.cu_btn_up.setIcon(icon7)
        self.cu_btn_up.setObjectName("cu_btn_up")
        self.horizontalLayout_25.addWidget(self.cu_btn_up)
        self.cust_btn_close = QtWidgets.QPushButton(self.widget_8)
```

```
        self.cust_btn_close.setMinimumSize(QtCore.QSize(120, 0))
        self.cust_btn_close.setIcon(icon8)
        self.cust_btn_close.setObjectName("cust_btn_close")
        self.horizontalLayout_25.addWidget(self.cust_btn_close)
        self.gridLayout_8.addLayout(self.horizontalLayout_25, 0, 0, 1, 1)
        self.verticalLayout_16.addWidget(self.widget_8)
        self.horizontalLayout_34 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_34.setObjectName("horizontalLayout_34")
        self.back_2 = QtWidgets.QPushButton(self.tab_customer)
        self.back_2.setMinimumSize(QtCore.QSize(28, 24))
        self.back_2.setMaximumSize(QtCore.QSize(28, 24))
        self.back_2.setText("")
        self.back_2.setIcon(icon9)
        self.back_2.setObjectName("back_2")
        self.horizontalLayout_34.addWidget(self.back_2)
        spacerItem11 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_34.addItem(spacerItem11)
        self.verticalLayout_16.addLayout(self.horizontalLayout_34)
        self.table_cust = QtWidgets.QTableWidget(self.tab_customer)
        self.table_cust.setEditTriggers(QtWidgets.QAbstractItemView.NoEditTriggers)
        self.table_cust.setObjectName("table_cust")
        self.table_cust.setColumnCount(0)
        self.table_cust.setRowCount(0)
        self.verticalLayout_16.addWidget(self.table_cust)
        self.horizontalLayout_12.addLayout(self.verticalLayout_16)
        self.widget_2 = QtWidgets.QWidget(self.tab_customer)
        self.widget_2.setObjectName("widget_2")
        self.gridLayout_5 = QtWidgets.QGridLayout(self.widget_2)
        self.gridLayout_5.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_5.setObjectName("gridLayout_5")
        spacerItem12 = QtWidgets.QSpacerItem(563, 1, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.gridLayout_5.addItem(spacerItem12, 2, 0, 1, 1)
        self.verticalLayout_19 = QtWidgets.QVBoxLayout()
        self.verticalLayout_19.setObjectName("verticalLayout_19")
        self.horizontalLayout_21 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_21.setObjectName("horizontalLayout_21")
        self.verticalLayout_18 = QtWidgets.QVBoxLayout()
        self.verticalLayout_18.setObjectName("verticalLayout_18")
        self.label_22 = QtWidgets.QLabel(self.widget_2)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_22.setFont(font)
        self.label_22.setObjectName("label_22")
        self.verticalLayout_18.addWidget(self.label_22)
        self.label_23 = QtWidgets.QLabel(self.widget_2)
```

```
font = QtGui.QFont()
font.setPointSize(10)
self.label_23.setFont(font)
self.label_23.setObjectName("label_23")
self.verticalLayout_18.addWidget(self.label_23)
self.label_44 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(10)
self.label_44.setFont(font)
self.label_44.setObjectName("label_44")
self.verticalLayout_18.addWidget(self.label_44)
self.label_35 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(10)
self.label_35.setFont(font)
self.label_35.setObjectName("label_35")
self.verticalLayout_18.addWidget(self.label_35)
self.label_24 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(10)
self.label_24.setFont(font)
self.label_24.setObjectName("label_24")
self.verticalLayout_18.addWidget(self.label_24)
self.label_25 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(10)
self.label_25.setFont(font)
self.label_25.setObjectName("label_25")
self.verticalLayout_18.addWidget(self.label_25)
self.label_26 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(10)
self.label_26.setFont(font)
self.label_26.setObjectName("label_26")
self.verticalLayout_18.addWidget(self.label_26)
self.label_27 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(10)
self.label_27.setFont(font)
self.label_27.setObjectName("label_27")
self.verticalLayout_18.addWidget(self.label_27)
self.horizontalLayout_21.addLayout(self.verticalLayout_18)
self.verticalLayout_15 = QtWidgets.QVBoxLayout()
self.verticalLayout_15.setObjectName("verticalLayout_15")
self.lineEdit_19 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_19.setObjectName("lineEdit_19")
self.verticalLayout_15.addWidget(self.lineEdit_19)
```

```
self.lineEdit_21 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_21.setObjectName("lineEdit_21")
self.verticalLayout_15.addWidget(self.lineEdit_21)
self.lineEdit_11 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_11.setObjectName("lineEdit_11")
self.verticalLayout_15.addWidget(self.lineEdit_11)
self.lineEdit_16 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_16.setObjectName("lineEdit_16")
self.verticalLayout_15.addWidget(self.lineEdit_16)
self.lineEdit_22 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_22.setObjectName("lineEdit_22")
self.verticalLayout_15.addWidget(self.lineEdit_22)
self.lineEdit_23 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_23.setObjectName("lineEdit_23")
self.verticalLayout_15.addWidget(self.lineEdit_23)
self.lineEdit_24 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_24.setObjectName("lineEdit_24")
self.verticalLayout_15.addWidget(self.lineEdit_24)
self.lineEdit_25 = QtWidgets.QLineEdit(self.widget_2)
self.lineEdit_25.setObjectName("lineEdit_25")
self.verticalLayout_15.addWidget(self.lineEdit_25)
self.horizontalLayout_21.addLayout(self.verticalLayout_15)
self.verticalLayout_19.addLayout(self.horizontalLayout_21)
self.horizontalLayout_13 = QtWidgets.QHBoxLayout()
self.horizontalLayout_13.setObjectName("horizontalLayout_13")
self.label_28 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(10)
self.label_28.setFont(font)
self.label_28.setObjectName("label_28")
self.horizontalLayout_13.addWidget(self.label_28)
self.comboBox = QtWidgets.QComboBox(self.widget_2)
self.comboBox.setObjectName("comboBox")
self.horizontalLayout_13.addWidget(self.comboBox)
self.Cust_sent_btn = QtWidgets.QPushButton(self.widget_2)
self.Cust_sent_btn.setIcon(icon7)
self.Cust_sent_btn.setObjectName("Cust_sent_btn")
self.horizontalLayout_13.addWidget(self.Cust_sent_btn)
self.Cust_cl_btn = QtWidgets.QPushButton(self.widget_2)
self.Cust_cl_btn.setIcon(icon8)
self.Cust_cl_btn.setObjectName("Cust_cl_btn")
self.horizontalLayout_13.addWidget(self.Cust_cl_btn)
self.verticalLayout_19.addLayout(self.horizontalLayout_13)
self.gridLayout_5.addLayout(self.verticalLayout_19, 1, 0, 1, 1)
self.label_57 = QtWidgets.QLabel(self.widget_2)
font = QtGui.QFont()
font.setPointSize(11)
```

```
        font.setBold(True)
        font.setWeight(75)
        self.label_57.setFont(font)
        self.label_57.setObjectName("label_57")
        self.gridLayout_5.addWidget(self.label_57, 0, 0, 1, 1)
        self.horizontalLayout_12.addWidget(self.widget_2)
        self.gridLayout_12.addLayout(self.horizontalLayout_12, 1, 0, 1, 2)
        icon12 = QtGui.QIcon()
        icon12.addPixmap(QtGui.QPixmap("icons/Customer.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.tab.addTab(self.tab_customer, icon12, "")
        self.tab_2 = QtWidgets.QWidget()
        self.tab_2.setObjectName("tab_2")
        self.gridLayout_9 = QtWidgets.QGridLayout(self.tab_2)
        self.gridLayout_9.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_9.setObjectName("gridLayout_9")
        self.verticalLayout_29 = QtWidgets.QVBoxLayout()
        self.verticalLayout_29.setObjectName("verticalLayout_29")
        self.horizontalLayout_11 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_11.setObjectName("horizontalLayout_11")
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setContentsMargins(-1, 8, -1, 0)
        self.verticalLayout.setSpacing(0)
        self.verticalLayout.setObjectName("verticalLayout")
        self.label_10 = QtWidgets.QLabel(self.tab_2)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_10.setFont(font)
        self.label_10.setObjectName("label_10")
        self.verticalLayout.addWidget(self.label_10)
        self.label_12 = QtWidgets.QLabel(self.tab_2)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_12.setFont(font)
        self.label_12.setObjectName("label_12")
        self.verticalLayout.addWidget(self.label_12)
        self.label_9 = QtWidgets.QLabel(self.tab_2)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_9.setFont(font)
        self.label_9.setObjectName("label_9")
        self.verticalLayout.addWidget(self.label_9)
        self.label_13 = QtWidgets.QLabel(self.tab_2)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_13.setFont(font)
        self.label_13.setObjectName("label_13")
```

```
        self.verticalLayout.addWidget(self.label_13)
        self.horizontalLayout_11.addLayout(self.verticalLayout)
        self.verticalLayout_27 = QtWidgets.QVBoxLayout()
        self.verticalLayout_27.setObjectName("verticalLayout_27")
        self.horizontalLayout_10 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_10.setObjectName("horizontalLayout_10")
        self.dateEdit_3 = QtWidgets.QDateEdit(self.tab_2)
        self.dateEdit_3.setEnabled(False)
        self.dateEdit_3.setObjectName("dateEdit_3")
        self.horizontalLayout_10.addWidget(self.dateEdit_3)
        self.label_42 = QtWidgets.QLabel(self.tab_2)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_42.setFont(font)
        self.label_42.setObjectName("label_42")
        self.horizontalLayout_10.addWidget(self.label_42)
        self.dateEdit_4 = QtWidgets.QDateEdit(self.tab_2)
        self.dateEdit_4.setEnabled(False)
        self.dateEdit_4.setObjectName("dateEdit_4")
        self.horizontalLayout_10.addWidget(self.dateEdit_4)
        self.checkBox_2 = QtWidgets.QCheckBox(self.tab_2)
        self.checkBox_2.setObjectName("checkBox_2")
        self.horizontalLayout_10.addWidget(self.checkBox_2)
        self.verticalLayout_2 = QtWidgets.QVBoxLayout()
        self.verticalLayout_2.setSpacing(0)
        self.verticalLayout_2.setObjectName("verticalLayout_2")
        self.radioButton = QtWidgets.QRadioButton(self.tab_2)
        self.radioButton.setAutoExclusive(False)
        self.radioButton.setObjectName("radioButton")
        self.verticalLayout_2.addWidget(self.radioButton)
        self.radioButton_2 = QtWidgets.QRadioButton(self.tab_2)
        self.radioButton_2.setAutoExclusive(False)
        self.radioButton_2.setObjectName("radioButton_2")
        self.verticalLayout_2.addWidget(self.radioButton_2)
        self.horizontalLayout_10.addLayout(self.verticalLayout_2)
        spacerItem13 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_10.addItem(spacerItem13)
        self.verticalLayout_27.addLayout(self.horizontalLayout_10)
        self.horizontalLayout_3 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_3.setObjectName("horizontalLayout_3")
        self.comboBox_4 = QtWidgets.QComboBox(self.tab_2)
        self.comboBox_4.setObjectName("comboBox_4")
        self.comboBox_4.addItem("")
        self.comboBox_4.setItemText(0, "")
        self.comboBox_4.addItem("")
        self.comboBox_4.addItem("")
```

```
        self.comboBox_4.addItem("")
        self.horizontalLayout_3.addWidget(self.comboBox_4)
        self.lineEdit_12 = QtWidgets.QLineEdit(self.tab_2)
        self.lineEdit_12.setObjectName("lineEdit_12")
        self.horizontalLayout_3.addWidget(self.lineEdit_12)
        self.verticalLayout_27.addLayout(self.horizontalLayout_3)
        self.lineEdit_9 = QtWidgets.QLineEdit(self.tab_2)
        self.lineEdit_9.setObjectName("lineEdit_9")
        self.verticalLayout_27.addWidget(self.lineEdit_9)
        self.lineEdit_13 = QtWidgets.QLineEdit(self.tab_2)
        self.lineEdit_13.setObjectName("lineEdit_13")
        self.verticalLayout_27.addWidget(self.lineEdit_13)
        self.horizontalLayout_11.addLayout(self.verticalLayout_27)
        self.verticalLayout_29.addLayout(self.horizontalLayout_11)
        self.label_30 = QtWidgets.QLabel(self.tab_2)
        font = QtGui.QFont()
        font.setPointSize(8)
        self.label_30.setFont(font)
        self.label_30.setStyleSheet("color :#ff3333")
        self.label_30.setObjectName("label_30")
        self.verticalLayout_29.addWidget(self.label_30)
        spacerItem14 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_29.addItem(spacerItem14)
        self.gridLayout_9.addLayout(self.verticalLayout_29, 0, 0, 1, 1)
        self.verticalLayout_4 = QtWidgets.QVBoxLayout()
        self.verticalLayout_4.setObjectName("verticalLayout_4")
        self.cl_2 = QtWidgets.QPushButton(self.tab_2)
        self.cl_2.setIcon(icon1)
        self.cl_2.setObjectName("cl_2")
        self.verticalLayout_4.addWidget(self.cl_2)
        self.btn_con = QtWidgets.QPushButton(self.tab_2)
        self.btn_con.setIcon(icon4)
        self.btn_con.setObjectName("btn_con")
        self.verticalLayout_4.addWidget(self.btn_con)
        self.up_btn_con = QtWidgets.QPushButton(self.tab_2)
        self.up_btn_con.setIcon(icon6)
        self.up_btn_con.setObjectName("up_btn_con")
        self.verticalLayout_4.addWidget(self.up_btn_con)
        self.Con_add_btn = QtWidgets.QPushButton(self.tab_2)
        self.Con_add_btn.setIcon(icon5)
        self.Con_add_btn.setObjectName("Con_add_btn")
        self.verticalLayout_4.addWidget(self.Con_add_btn)
        self.Con_del_btn = QtWidgets.QPushButton(self.tab_2)
        self.Con_del_btn.setIcon(icon11)
        self.Con_del_btn.setObjectName("Con_del_btn")
        self.verticalLayout_4.addWidget(self.Con_del_btn)
```

```
        spacerItem15 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_4.addItem(spacerItem15)
        self.gridLayout_9.addLayout(self.verticalLayout_4, 0, 1, 1, 1)
        self.horizontalLayout_32 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_32.setObjectName("horizontalLayout_32")
        self.verticalLayout_25 = QtWidgets.QVBoxLayout()
        self.verticalLayout_25.setObjectName("verticalLayout_25")
        self.widget_5 = QtWidgets.QWidget(self.tab_2)
        self.widget_5.setObjectName("widget_5")
        self.gridLayout_6 = QtWidgets.QGridLayout(self.widget_5)
        self.gridLayout_6.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_6.setObjectName("gridLayout_6")
        self.verticalLayout_14 = QtWidgets.QVBoxLayout()
        self.verticalLayout_14.setObjectName("verticalLayout_14")
        self.horizontalLayout_17 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_17.setObjectName("horizontalLayout_17")
        self.label_38 = QtWidgets.QLabel(self.widget_5)
        self.label_38.setObjectName("label_38")
        self.horizontalLayout_17.addWidget(self.label_38)
        self.lineEdit_3 = QtWidgets.QLineEdit(self.widget_5)
        self.lineEdit_3.setObjectName("lineEdit_3")
        self.horizontalLayout_17.addWidget(self.lineEdit_3)
        self.co_btn_up = QtWidgets.QPushButton(self.widget_5)
        self.co_btn_up.setMinimumSize(QtCore.QSize(100, 0))
        self.co_btn_up.setIcon(icon7)
        self.co_btn_up.setObjectName("co_btn_up")
        self.horizontalLayout_17.addWidget(self.co_btn_up)
        self.con_cl_btn = QtWidgets.QPushButton(self.widget_5)
        self.con_cl_btn.setMinimumSize(QtCore.QSize(100, 0))
        self.con_cl_btn.setIcon(icon8)
        self.con_cl_btn.setObjectName("con_cl_btn")
        self.horizontalLayout_17.addWidget(self.con_cl_btn)
        self.verticalLayout_14.addLayout(self.horizontalLayout_17)
        self.gridLayout_6.addLayout(self.verticalLayout_14, 0, 0, 1, 1)
        self.verticalLayout_25.addWidget(self.widget_5)
        self.horizontalLayout_35 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_35.setObjectName("horizontalLayout_35")
        self.back_3 = QtWidgets.QPushButton(self.tab_2)
        self.back_3.setMinimumSize(QtCore.QSize(28, 24))
        self.back_3.setMaximumSize(QtCore.QSize(28, 24))
        self.back_3.setText("")
        self.back_3.setIcon(icon9)
        self.back_3.setObjectName("back_3")
        self.horizontalLayout_35.addWidget(self.back_3)
        spacerItem16 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
```

```
        self.horizontalLayout_35.addItem(spacerItem16)
        self.btn_graph = QtWidgets.QPushButton(self.tab_2)
        self.btn_graph.setMinimumSize(QtCore.QSize(100, 0))
        self.btn_graph.setIcon(icon2)
        self.btn_graph.setObjectName("btn_graph")
        self.horizontalLayout_35.addWidget(self.btn_graph)
        self.verticalLayout_25.addLayout(self.horizontalLayout_35)
        self.table_con = QtWidgets.QTableWidget(self.tab_2)
        self.table_con.setEditTriggers(QtWidgets.QAbstractItemView.NoEditTriggers)
        self.table_con.setObjectName("table_con")
        self.table_con.setColumnCount(0)
        self.table_con.setRowCount(0)
        self.verticalLayout_25.addWidget(self.table_con)
        self.horizontalLayout_32.addLayout(self.verticalLayout_25)
        self.widget_3 = QtWidgets.QWidget(self.tab_2)
        self.widget_3.setObjectName("widget_3")
        self.gridLayout_24 = QtWidgets.QGridLayout(self.widget_3)
        self.gridLayout_24.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_24.setObjectName("gridLayout_24")
        spacerItem17 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.gridLayout_24.addItem(spacerItem17, 3, 0, 1, 1)
        self.horizontalLayout_14 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_14.setObjectName("horizontalLayout_14")
        self.label_33 = QtWidgets.QLabel(self.widget_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_33.setFont(font)
        self.label_33.setObjectName("label_33")
        self.horizontalLayout_14.addWidget(self.label_33)
        self.lineEdit_31 = QtWidgets.QLineEdit(self.widget_3)
        self.lineEdit_31.setObjectName("lineEdit_31")
        self.horizontalLayout_14.addWidget(self.lineEdit_31)
        self.Con_sent_btn = QtWidgets.QPushButton(self.widget_3)
        self.Con_sent_btn.setIcon(icon7)
        self.Con_sent_btn.setObjectName("Con_sent_btn")
        self.horizontalLayout_14.addWidget(self.Con_sent_btn)
        self.Con_cl_btn = QtWidgets.QPushButton(self.widget_3)
        self.Con_cl_btn.setIcon(icon8)
        self.Con_cl_btn.setObjectName("Con_cl_btn")
        self.horizontalLayout_14.addWidget(self.Con_cl_btn)
        self.gridLayout_24.addLayout(self.horizontalLayout_14, 2, 0, 1, 1)
        self.label_55 = QtWidgets.QLabel(self.widget_3)
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
```

```
        self.label_55.setFont(font)
        self.label_55.setObjectName("label_55")
        self.gridLayout_24.addWidget(self.label_55, 0, 0, 1, 1)
        self.horizontalLayout_15 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_15.setObjectName("horizontalLayout_15")
        self.verticalLayout_17 = QtWidgets.QVBoxLayout()
        self.verticalLayout_17.setObjectName("verticalLayout_17")
        self.label_29 = QtWidgets.QLabel(self.widget_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_29.setFont(font)
        self.label_29.setObjectName("label_29")
        self.verticalLayout_17.addWidget(self.label_29)
        self.label_20 = QtWidgets.QLabel(self.widget_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_20.setFont(font)
        self.label_20.setObjectName("label_20")
        self.verticalLayout_17.addWidget(self.label_20)
        self.label_3 = QtWidgets.QLabel(self.widget_3)
        self.label_3.setObjectName("label_3")
        self.verticalLayout_17.addWidget(self.label_3)
        self.horizontalLayout_15.addLayout(self.verticalLayout_17)
        self.verticalLayout_9 = QtWidgets.QVBoxLayout()
        self.verticalLayout_9.setObjectName("verticalLayout_9")
        self.horizontalLayout_29 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_29.setObjectName("horizontalLayout_29")
        self.dateEdit_5 = QtWidgets.QDateEdit(self.widget_3)
        self.dateEdit_5.setObjectName("dateEdit_5")
        self.horizontalLayout_29.addWidget(self.dateEdit_5)
        spacerItem18 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_29.addItem(spacerItem18)
        self.verticalLayout_9.addLayout(self.horizontalLayout_29)
        self.horizontalLayout_26 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_26.setObjectName("horizontalLayout_26")
        self.dateEdit_6 = QtWidgets.QDateEdit(self.widget_3)
        self.dateEdit_6.setObjectName("dateEdit_6")
        self.horizontalLayout_26.addWidget(self.dateEdit_6)
        spacerItem19 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_26.addItem(spacerItem19)
        self.verticalLayout_9.addLayout(self.horizontalLayout_26)
        self.lineEdit_26 = QtWidgets.QLineEdit(self.widget_3)
        self.lineEdit_26.setObjectName("lineEdit_26")
        self.verticalLayout_9.addWidget(self.lineEdit_26)
        self.horizontalLayout_15.addLayout(self.verticalLayout_9)
```

```
        self.gridLayout_24.addLayout(self.horizontalLayout_15, 1, 0, 1, 1)
        self.horizontalLayout_32.addWidget(self.widget_3)
        self.gridLayout_9.addLayout(self.horizontalLayout_32, 1, 0, 1, 2)
        icon13 = QtGui.QIcon()
        icon13.addPixmap(QtGui.QPixmap("icons/Consumtrion.png"),      QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.tab.addTab(self.tab_2, icon13, "")
        self.tab_3 = QtWidgets.QWidget()
        self.tab_3.setObjectName("tab_3")
        self.gridLayout_10 = QtWidgets.QGridLayout(self.tab_3)
        self.gridLayout_10.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_10.setObjectName("gridLayout_10")
        self.verticalLayout_28 = QtWidgets.QVBoxLayout()
        self.verticalLayout_28.setObjectName("verticalLayout_28")
        self.horizontalLayout_7 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_7.setObjectName("horizontalLayout_7")
        self.verticalLayout_10 = QtWidgets.QVBoxLayout()
        self.verticalLayout_10.setObjectName("verticalLayout_10")
        self.label_17 = QtWidgets.QLabel(self.tab_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_17.setFont(font)
        self.label_17.setObjectName("label_17")
        self.verticalLayout_10.addWidget(self.label_17)
        self.label_18 = QtWidgets.QLabel(self.tab_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_18.setFont(font)
        self.label_18.setObjectName("label_18")
        self.verticalLayout_10.addWidget(self.label_18)
        self.label_16 = QtWidgets.QLabel(self.tab_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_16.setFont(font)
        self.label_16.setObjectName("label_16")
        self.verticalLayout_10.addWidget(self.label_16)
        self.label_15 = QtWidgets.QLabel(self.tab_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_15.setFont(font)
        self.label_15.setObjectName("label_15")
        self.verticalLayout_10.addWidget(self.label_15)
        self.label_19 = QtWidgets.QLabel(self.tab_3)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_19.setFont(font)
        self.label_19.setObjectName("label_19")
```

```
self.verticalLayout_10.addWidget(self.label_19)
self.horizontalLayout_7.addLayout(self.verticalLayout_10)
self.verticalLayout_11 = QtWidgets.QVBoxLayout()
self.verticalLayout_11.setObjectName("verticalLayout_11")
self.horizontalLayout_41 = QtWidgets.QHBoxLayout()
self.horizontalLayout_41.setObjectName("horizontalLayout_41")
self.comboBox_5 = QtWidgets.QComboBox(self.tab_3)
self.comboBox_5.setObjectName("comboBox_5")
self.comboBox_5.addItem("")
self.comboBox_5.setItemText(0, "")
self.comboBox_5.addItem("")
self.comboBox_5.addItem("")
self.comboBox_5.addItem("")
self.horizontalLayout_41.addWidget(self.comboBox_5)
self.lineEdit_17 = QtWidgets.QLineEdit(self.tab_3)
self.lineEdit_17.setObjectName("lineEdit_17")
self.horizontalLayout_41.addWidget(self.lineEdit_17)
self.verticalLayout_11.addLayout(self.horizontalLayout_41)
self.horizontalLayout_37 = QtWidgets.QHBoxLayout()
self.horizontalLayout_37.setObjectName("horizontalLayout_37")
self.comboBox_6 = QtWidgets.QComboBox(self.tab_3)
self.comboBox_6.setObjectName("comboBox_6")
self.comboBox_6.addItem("")
self.comboBox_6.setItemText(0, "")
self.comboBox_6.addItem("")
self.comboBox_6.addItem("")
self.comboBox_6.addItem("")
self.horizontalLayout_37.addWidget(self.comboBox_6)
self.lineEdit_18 = QtWidgets.QLineEdit(self.tab_3)
self.lineEdit_18.setObjectName("lineEdit_18")
self.horizontalLayout_37.addWidget(self.lineEdit_18)
self.verticalLayout_11.addLayout(self.horizontalLayout_37)
self.horizontalLayout_42 = QtWidgets.QHBoxLayout()
self.horizontalLayout_42.setObjectName("horizontalLayout_42")
self.dateEdit = QtWidgets.QDateEdit(self.tab_3)
self.dateEdit.setEnabled(False)
self.dateEdit.setObjectName("dateEdit")
self.horizontalLayout_42.addWidget(self.dateEdit)
self.label_41 = QtWidgets.QLabel(self.tab_3)
self.label_41.setObjectName("label_41")
self.horizontalLayout_42.addWidget(self.label_41)
self.dateEdit_2 = QtWidgets.QDateEdit(self.tab_3)
self.dateEdit_2.setEnabled(False)
self.dateEdit_2.setObjectName("dateEdit_2")
self.horizontalLayout_42.addWidget(self.dateEdit_2)
self.checkBox = QtWidgets.QCheckBox(self.tab_3)
font = QtGui.QFont()
```

```
        font.setPointSize(10)
        self.checkBox.setFont(font)
        self.checkBox.setObjectName("checkBox")
        self.horizontalLayout_42.addWidget(self.checkBox)
        spacerItem20 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_42.addItem(spacerItem20)
        self.verticalLayout_11.addLayout(self.horizontalLayout_42)
        self.lineEdit_15 = QtWidgets.QLineEdit(self.tab_3)
        self.lineEdit_15.setObjectName("lineEdit_15")
        self.verticalLayout_11.addWidget(self.lineEdit_15)
        self.lineEdit_14 = QtWidgets.QLineEdit(self.tab_3)
        self.lineEdit_14.setObjectName("lineEdit_14")
        self.verticalLayout_11.addWidget(self.lineEdit_14)
        self.horizontalLayout_7.addLayout(self.verticalLayout_11)
        self.verticalLayout_28.addLayout(self.horizontalLayout_7)
        self.label_36 = QtWidgets.QLabel(self.tab_3)
        font = QtGui.QFont()
        font.setPointSize(8)
        self.label_36.setFont(font)
        self.label_36.setStyleSheet("color :#ff3333")
        self.label_36.setObjectName("label_36")
        self.verticalLayout_28.addWidget(self.label_36)
        spacerItem21 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_28.addItem(spacerItem21)
        self.gridLayout_10.addLayout(self.verticalLayout_28, 0, 0, 1, 1)
        self.verticalLayout_8 = QtWidgets.QVBoxLayout()
        self.verticalLayout_8.setObjectName("verticalLayout_8")
        self.cl = QtWidgets.QPushButton(self.tab_3)
        self.cl.setIcon(icon1)
        self.cl.setObjectName("cl")
        self.verticalLayout_8.addWidget(self.cl)
        self.btn_cha = QtWidgets.QPushButton(self.tab_3)
        self.btn_cha.setIcon(icon4)
        self.btn_cha.setObjectName("btn_cha")
        self.verticalLayout_8.addWidget(self.btn_cha)
        self.up_btn_cha = QtWidgets.QPushButton(self.tab_3)
        self.up_btn_cha.setIcon(icon6)
        self.up_btn_cha.setObjectName("up_btn_cha")
        self.verticalLayout_8.addWidget(self.up_btn_cha)
        self.Cha_add_btn = QtWidgets.QPushButton(self.tab_3)
        self.Cha_add_btn.setIcon(icon5)
        self.Cha_add_btn.setObjectName("Cha_add_btn")
        self.verticalLayout_8.addWidget(self.Cha_add_btn)
        self.Cha_del_btn = QtWidgets.QPushButton(self.tab_3)
        self.Cha_del_btn.setIcon(icon11)
```

```
        self.Cha_del_btn.setObjectName("Cha_del_btn")
        self.verticalLayout_8.addWidget(self.Cha_del_btn)
        spacerItem22 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.verticalLayout_8.addItem(spacerItem22)
        self.gridLayout_10.addLayout(self.verticalLayout_8, 0, 1, 1, 1)
        self.horizontalLayout_28 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_28.setObjectName("horizontalLayout_28")
        self.verticalLayout_20 = QtWidgets.QVBoxLayout()
        self.verticalLayout_20.setObjectName("verticalLayout_20")
        self.widget_6 = QtWidgets.QWidget(self.tab_3)
        self.widget_6.setObjectName("widget_6")
        self.gridLayout_7 = QtWidgets.QGridLayout(self.widget_6)
        self.gridLayout_7.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_7.setObjectName("gridLayout_7")
        self.horizontalLayout_16 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_16.setObjectName("horizontalLayout_16")
        self.label_37 = QtWidgets.QLabel(self.widget_6)
        self.label_37.setObjectName("label_37")
        self.horizontalLayout_16.addWidget(self.label_37)
        self.lineEdit_2 = QtWidgets.QLineEdit(self.widget_6)
        self.lineEdit_2.setObjectName("lineEdit_2")
        self.horizontalLayout_16.addWidget(self.lineEdit_2)
        self.cha_btn_up = QtWidgets.QPushButton(self.widget_6)
        self.cha_btn_up.setMinimumSize(QtCore.QSize(100, 0))
        self.cha_btn_up.setIcon(icon7)
        self.cha_btn_up.setObjectName("cha_btn_up")
        self.horizontalLayout_16.addWidget(self.cha_btn_up)
        self.cha_cl_btn = QtWidgets.QPushButton(self.widget_6)
        self.cha_cl_btn.setMinimumSize(QtCore.QSize(100, 0))
        self.cha_cl_btn.setIcon(icon8)
        self.cha_cl_btn.setObjectName("cha_cl_btn")
        self.horizontalLayout_16.addWidget(self.cha_cl_btn)
        self.gridLayout_7.addLayout(self.horizontalLayout_16, 0, 0, 1, 1)
        self.verticalLayout_20.addWidget(self.widget_6)
        self.horizontalLayout_36 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_36.setObjectName("horizontalLayout_36")
        self.back_4 = QtWidgets.QPushButton(self.tab_3)
        self.back_4.setMinimumSize(QtCore.QSize(28, 24))
        self.back_4.setMaximumSize(QtCore.QSize(28, 24))
        self.back_4.setText("")
        self.back_4.setIcon(icon9)
        self.back_4.setObjectName("back_4")
        self.horizontalLayout_36.addWidget(self.back_4)
        self.cha_dwl = QtWidgets.QPushButton(self.tab_3)
        self.cha_dwl.setMaximumSize(QtCore.QSize(28, 24))
        icon14 = QtGui.QIcon()
```

```
        icon14.addPixmap(QtGui.QPixmap("icons/exlxs.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.cha_dwl.setIcon(icon14)
        self.cha_dwl.setObjectName("cha_dwl")
        self.horizontalLayout_36.addWidget(self.cha_dwl)
        self.btn_sum = QtWidgets.QPushButton(self.tab_3)
        self.btn_sum.setMaximumSize(QtCore.QSize(28, 24))
        self.btn_sum.setText("")
        icon15 = QtGui.QIcon()
        icon15.addPixmap(QtGui.QPixmap("icons/s.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.btn_sum.setIcon(icon15)
        self.btn_sum.setObjectName("btn_sum")
        self.horizontalLayout_36.addWidget(self.btn_sum)
        self.label_59 = QtWidgets.QLabel(self.tab_3)
        self.label_59.setObjectName("label_59")
        self.horizontalLayout_36.addWidget(self.label_59)
        self.label_60 = QtWidgets.QLabel(self.tab_3)
        self.label_60.setStyleSheet("color :#ff3333")
        self.label_60.setText("")
        self.label_60.setObjectName("label_60")
        self.horizontalLayout_36.addWidget(self.label_60)
        self.label_61 = QtWidgets.QLabel(self.tab_3)
        self.label_61.setObjectName("label_61")
        self.horizontalLayout_36.addWidget(self.label_61)
        self.label_62 = QtWidgets.QLabel(self.tab_3)
        self.label_62.setStyleSheet("color :#ff3333")
        self.label_62.setText("")
        self.label_62.setObjectName("label_62")
        self.horizontalLayout_36.addWidget(self.label_62)
        spacerItem23 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_36.addItem(spacerItem23)
        self.btn_graph_2 = QtWidgets.QPushButton(self.tab_3)
        self.btn_graph_2.setMinimumSize(QtCore.QSize(100, 0))
        self.btn_graph_2.setIcon(icon2)
        self.btn_graph_2.setObjectName("btn_graph_2")
        self.horizontalLayout_36.addWidget(self.btn_graph_2)
        self.verticalLayout_20.addLayout(self.horizontalLayout_36)
        self.table_cha = QtWidgets.QTableWidget(self.tab_3)
        self.table_cha.setEditTriggers(QtWidgets.QAbstractItemView.NoEditTriggers)
        self.table_cha.setObjectName("table_cha")
        self.table_cha.setColumnCount(0)
        self.table_cha.setRowCount(0)
        self.verticalLayout_20.addWidget(self.table_cha)
        self.horizontalLayout_28.addLayout(self.verticalLayout_20)
        self.widget_4 = QtWidgets.QWidget(self.tab_3)
```

```
        self.widget_4.setObjectName("widget_4")
        self.gridLayout_4 = QtWidgets.QGridLayout(self.widget_4)
        self.gridLayout_4.setContentsMargins(0, 0, 0, 0)
        self.gridLayout_4.setObjectName("gridLayout_4")
        self.label_56 = QtWidgets.QLabel(self.widget_4)
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_56.setFont(font)
        self.label_56.setObjectName("label_56")
        self.gridLayout_4.addWidget(self.label_56, 0, 0, 1, 1)
        spacerItem24 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
QtWidgets.QSizePolicy.Expanding)
        self.gridLayout_4.addItem(spacerItem24, 4, 0, 1, 1)
        self.verticalLayout_23 = QtWidgets.QVBoxLayout()
        self.verticalLayout_23.setObjectName("verticalLayout_23")
        self.horizontalLayout_5 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_5.setObjectName("horizontalLayout_5")
        self.label_14 = QtWidgets.QLabel(self.widget_4)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.label_14.setFont(font)
        self.label_14.setObjectName("label_14")
        self.horizontalLayout_5.addWidget(self.label_14)
        self.lineEdit_35 = QtWidgets.QLineEdit(self.widget_4)
        self.lineEdit_35.setObjectName("lineEdit_35")
        self.horizontalLayout_5.addWidget(self.lineEdit_35)
        self.Cha_sent_btn = QtWidgets.QPushButton(self.widget_4)
        self.Cha_sent_btn.setIcon(icon7)
        self.Cha_sent_btn.setObjectName("Cha_sent_btn")
        self.horizontalLayout_5.addWidget(self.Cha_sent_btn)
        self.Cha_cl_btn = QtWidgets.QPushButton(self.widget_4)
        self.Cha_cl_btn.setIcon(icon8)
        self.Cha_cl_btn.setObjectName("Cha_cl_btn")
        self.horizontalLayout_5.addWidget(self.Cha_cl_btn)
        self.verticalLayout_23.addLayout(self.horizontalLayout_5)
        self.gridLayout_4.addLayout(self.verticalLayout_23, 2, 0, 1, 1)
        self.horizontalLayout_38 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_38.setContentsMargins(76, -1, -1, -1)
        self.horizontalLayout_38.setSpacing(6)
        self.horizontalLayout_38.setObjectName("horizontalLayout_38")
        self.radioButton_5 = QtWidgets.QRadioButton(self.widget_4)
        self.radioButton_5.setChecked(True)
        self.radioButton_5.setObjectName("radioButton_5")
        self.horizontalLayout_38.addWidget(self.radioButton_5)
        self.radioButton_3 = QtWidgets.QRadioButton(self.widget_4)
```

```
        self.radioButton_3.setObjectName("radioButton_3")
        self.horizontalLayout_38.addWidget(self.radioButton_3)
        self.radioButton_4 = QtWidgets.QRadioButton(self.widget_4)
        self.radioButton_4.setObjectName("radioButton_4")
        self.horizontalLayout_38.addWidget(self.radioButton_4)
        spacerItem25 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_38.addItem(spacerItem25)
        self.gridLayout_4.addLayout(self.horizontalLayout_38, 3, 0, 1, 1)
        self.horizontalLayout_28.addWidget(self.widget_4)
        self.gridLayout_10.addLayout(self.horizontalLayout_28, 1, 0, 1, 2)
        icon16 = QtGui.QIcon()
        icon16.addPixmap(QtGui.QPixmap("icons/charge.png"), QtGui.QIcon.Normal,
QtGui.QIcon.Off)
        self.tab.addTab(self.tab_3, icon16, "")
        self.gridLayout_2.addWidget(self.tab, 1, 0, 1, 1)
        self.label_40 = QtWidgets.QLabel(self.centralwidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Maximum,
QtWidgets.QSizePolicy.Maximum)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.label_40.sizePolicy().hasHeightForWidth())
        self.label_40.setSizePolicy(sizePolicy)
        self.label_40.setMinimumSize(QtCore.QSize(1000, 0))
        self.label_40.setMaximumSize(QtCore.QSize(16777215, 16777215))
        self.label_40.setText("")
        self.label_40.setTextFormat(QtCore.Qt.RichText)
        self.label_40.setPixmap(QtGui.QPixmap("icons/crm.png"))
        self.label_40.setScaledContents(False)
        self.label_40.setObjectName("label_40")
        self.gridLayout_2.addWidget(self.label_40, 2, 0, 1, 1)
        Form.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(Form)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 1364, 34))
        font = QtGui.QFont()
        font.setPointSize(15)
        font.setBold(True)
        font.setWeight(75)
        self.menubar.setFont(font)
        self.menubar.setStyleSheet("background-color:#006fc6;color:#f0f0f0")
        self.menubar.setDefaultUp(False)
        self.menubar.setNativeMenuBar(True)
        self.menubar.setObjectName("menubar")
        self.menuInfo = QtWidgets.QMenu(self.menubar)
        self.menuInfo.setObjectName("menuInfo")
        self.menuAbout_CRM_sys = QtWidgets.QMenu(self.menuInfo)
        self.menuAbout_CRM_sys.setObjectName("menuAbout_CRM_sys")
```

```
Form.setMenuBar(self.menubar)
self.statusbar = QtWidgets.QStatusBar(Form)
self.statusbar.setObjectName("statusbar")
Form.setStatusBar(self.statusbar)
self.actionHelp = QtWidgets.QAction(Form)
self.actionHelp.setObjectName("actionHelp")
self.actionExit = QtWidgets.QAction(Form)
self.actionExit.setObjectName("actionExit")
self.actionFullScreen = QtWidgets.QAction(Form)
self.actionFullScreen.setObjectName("actionFullScreen")
self.actionUnique_items = QtWidgets.QAction(Form)
self.actionUnique_items.setObjectName("actionUnique_items")
self.actionUpdate = QtWidgets.QAction(Form)
self.actionUpdate.setObjectName("actionUpdate")
self.actionDelete = QtWidgets.QAction(Form)
self.actionDelete.setObjectName("actionDelete")
self.actionSystem = QtWidgets.QAction(Form)
self.actionSystem.setObjectName("actionSystem")
self.actionShearch = QtWidgets.QAction(Form)
self.actionShearch.setObjectName("actionShearch")
self.menuAbout_CRM_sys.addSeparator()
self.menuAbout_CRM_sys.addAction(self.actionShearch)
self.menuAbout_CRM_sys.addAction(self.actionUnique_items)
self.menuAbout_CRM_sys.addAction(self.actionSystem)
self.menuInfo.addAction(self.actionFullScreen)
self.menuInfo.addAction(self.menuAbout_CRM_sys.menuAction())
self.menuInfo.addAction(self.actionHelp)
self.menuInfo.addAction(self.actionExit)
self.menubar.addAction(self.menuInfo.menuAction())

self.retranslateUi(Form)
self.tab.setCurrentIndex(0)
self.Con_add_btn.clicked.connect(self.widget_3.show)
self.Cu_add_btn.clicked.connect(self.widget_2.show)
self.Cha_add_btn.clicked.connect(self.widget_4.show)
self.Cust_cl_btn.clicked.connect(self.widget_2.hide)
self.A_add_btn.clicked.connect(self.widget.show)
self.Con_cl_btn.clicked.connect(self.widget_3.hide)
self.A_cl_btn.clicked.connect(self.widget.hide)
self.cl_2.clicked.connect(self.lineEdit_12.clear)
self.cl_2.clicked.connect(self.lineEdit_9.clear)
self.cl_2.clicked.connect(self.lineEdit_13.clear)
self.cl.clicked.connect(self.lineEdit_17.clear)
self.cl.clicked.connect(self.lineEdit_18.clear)
self.cl.clicked.connect(self.lineEdit_15.clear)
self.cl.clicked.connect(self.lineEdit_14.clear)
self.cl_3.clicked.connect(self.lineEdit_7.clear)
```

```
        self.cl_3.clicked.connect(self.lineEdit.clear)
        self.cl_3.clicked.connect(self.lineEdit_5.clear)
        self.cl_3.clicked.connect(self.lineEdit_6.clear)
        self.cl_3.clicked.connect(self.lineEdit_8.clear)
        self.cl_4.clicked.connect(self.el_id.clear)
        self.cl_4.clicked.connect(self.el_name.clear)
        self.Cha_cl_btn.clicked.connect(self.widget_4.hide)
        self.up_btn_con.clicked.connect(self.widget_5.show)
        self.con_cl_btn.clicked.connect(self.widget_5.hide)
        self.up_btn_cha.clicked.connect(self.widget_6.show)
        self.cha_cl_btn.clicked.connect(self.widget_6.hide)
        self.up_ar_btn.clicked.connect(self.widget_7.show)
        self.pushButton.clicked.connect(self.widget_7.hide)
        self.cust_btn_update.clicked.connect(self.widget_8.show)
        self.cust_btn_close.clicked.connect(self.widget_8.hide)
        self.checkBox_2.toggled['bool'].connect(self.dateEdit_4.setEnabled)
        self.checkBox_2.toggled['bool'].connect(self.dateEdit_3.setEnabled)
        self.checkBox.toggled['bool'].connect(self.dateEdit_2.setEnabled)
        self.checkBox.toggled['bool'].connect(self.dateEdit.setEnabled)
        self.home_cl.clicked.connect(self.lineEdit_27.clear)
        self.home_cl.clicked.connect(self.lineEdit_28.clear)
        self.home_cl.clicked.connect(self.lineEdit_32.clear)
        self.home_cl.clicked.connect(self.lineEdit_33.clear)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Clover "))
        self.label_51.setText(_translate("Form", "Compare your customer consumption :"))
        self.label_47.setText(_translate("Form", "Welcome to Clover"))
        self.label_4.setText(_translate("Form", "Customer 1 "))
        self.label_31.setText(_translate("Form", "Customer 2 "))
        self.label_32.setText(_translate("Form", "Customer 3 "))
        self.home_cl.setText(_translate("Form", "Clear"))
        self.btn_h_gr_2.setText(_translate("Form", "Graph"))
        self.label_52.setText(_translate("Form", "Profit by Year"))
        self.label_48.setText(_translate("Form", "Add Year"))
        self.label_46.setText(_translate("Form", "Add Customer ID"))
        self.label_34.setText(_translate("Form", "Close Values By Month of :"))
        self.btn_h_gr.setText(_translate("Form", "Graph"))
        self.label_45.setText(_translate("Form", "If you wont a Help scanned a QR code or click
Ctrl+ H and go to\n"
"There you can find:"))
        self.label_50.setText(_translate("Form", "Clover Website"))
        self.label_53.setText(_translate("Form", "How you can update your data base?\n"
"\n"
"How you can delete an item?\n"
```

```
"\n"
"How you can insert items in a data base?"))
        self.tab.setTabText(self.tab.indexOf(self.tab_4), _translate("Form", "Home"))
        self.areaId.setText(_translate("Form", "Area ID     "))
        self.areaName.setText(_translate("Form", "Area Name"))
        self.label_2.setText(_translate("Form", "* Search all edit Line"))
        self.cl_4.setText(_translate("Form", "Clear"))
        self.btn_area.setText(_translate("Form", "Search"))
        self.A_add_btn.setText(_translate("Form", "Add Values"))
        self.up_ar_btn.setText(_translate("Form", "Update"))
        self.label_39.setText(_translate("Form", "Area Name"))
        self.a_btn_up.setText(_translate("Form", "Save"))
        self.pushButton.setText(_translate("Form", "Close"))
        self.table.setSortingEnabled(False)
        self.A_sent_btn.setText(_translate("Form", "Save"))
        self.A_cl_btn.setText(_translate("Form", "Close"))
        self.label_21.setText(_translate("Form", "Area Name"))
        self.label_58.setText(_translate("Form", "Add Value"))
        self.tab.setTabText(self.tab.indexOf(self.tab_area), _translate("Form", "Area"))
        self.label_7.setText(_translate("Form", "PricePerKW "))
        self.label.setText(_translate("Form", "Customer Id"))
        self.label_5.setText(_translate("Form", "PostCode    "))
        self.label_6.setText(_translate("Form", "City           "))
        self.label_8.setText(_translate("Form", "Area Id       "))
        self.comboBox_2.setItemText(1, _translate("Form", "=="))
        self.comboBox_2.setItemText(2, _translate("Form", ">="))
        self.comboBox_2.setItemText(3, _translate("Form", "<="))
        self.label_11.setText(_translate("Form", "* Search 4 edit line"))
        self.cl_3.setText(_translate("Form", "Clear"))
        self.btn_cust.setText(_translate("Form", "Search"))
        self.cust_btn_update.setText(_translate("Form", "Update"))
        self.Cu_add_btn.setText(_translate("Form", "Add Values"))
        self.Cust_del_btn.setText(_translate("Form", "Delete"))
        self.label_43.setText(_translate("Form", "Price Per KW"))
        self.cu_btn_up.setText(_translate("Form", "Save"))
        self.cust_btn_close.setText(_translate("Form", "Close"))
        self.table_cust.setSortingEnabled(False)
        self.label_22.setText(_translate("Form", "First Name"))
        self.label_23.setText(_translate("Form", "Last Name"))
        self.label_44.setText(_translate("Form", "Phone"))
        self.label_35.setText(_translate("Form", "e-mail"))
        self.label_24.setText(_translate("Form", "Address"))
        self.label_25.setText(_translate("Form", "PostCode"))
        self.label_26.setText(_translate("Form", "City"))
        self.label_27.setText(_translate("Form", "PricePerKW"))
        self.label_28.setText(_translate("Form", "Area ID"))
        self.Cust_sent_btn.setText(_translate("Form", "Save"))
```

```
        self.Cust_cl_btn.setText(_translate("Form", "Close"))
        self.label_57.setText(_translate("Form", "Add Value"))
        self.tab.setTabText(self.tab.indexOf(self.tab_customer), _translate("Form",
"Customer"))
        self.label_10.setText(_translate("Form", "Date Between Start"))
        self.label_12.setText(_translate("Form", "TotalKW"))
        self.label_9.setText(_translate("Form", "Consume Id"))
        self.label_13.setText(_translate("Form", "Customer Id"))
        self.dateEdit_3.setDisplayFormat(_translate("Form", "yyyy-MM-dd"))
        self.label_42.setText(_translate("Form", "and ending"))
        self.dateEdit_4.setDisplayFormat(_translate("Form", "yyyy-MM-dd"))
        self.checkBox_2.setText(_translate("Form", "Searching Date"))
        self.radioButton.setText(_translate("Form", "By 6 Months"))
        self.radioButton_2.setText(_translate("Form", "By 1 Year"))
        self.comboBox_4.setItemText(1, _translate("Form", "=="))
        self.comboBox_4.setItemText(2, _translate("Form", ">="))
        self.comboBox_4.setItemText(3, _translate("Form", "<="))
        self.label_30.setText(_translate("Form", "* Search 3 edit line"))
        self.cl_2.setText(_translate("Form", "Clear"))
        self.btn_con.setText(_translate("Form", "Search"))
        self.up_btn_con.setText(_translate("Form", "Update"))
        self.Con_add_btn.setText(_translate("Form", "Add Values"))
        self.Con_del_btn.setText(_translate("Form", "Delete"))
        self.label_38.setText(_translate("Form", "Total KW"))
        self.co_btn_up.setText(_translate("Form", "Save"))
        self.con_cl_btn.setText(_translate("Form", "Close"))
        self.btn_graph.setText(_translate("Form", "Graph"))
        self.table_con.setSortingEnabled(False)
        self.label_33.setText(_translate("Form", "Customer ID"))
        self.Con_sent_btn.setText(_translate("Form", "Save"))
        self.Con_cl_btn.setText(_translate("Form", "Close"))
        self.label_55.setText(_translate("Form", "Add Value"))
        self.label_29.setText(_translate("Form", "Start Date"))
        self.label_20.setText(_translate("Form", "End Date     "))
        self.label_3.setText(_translate("Form", "Total KW "))
        self.dateEdit_5.setDisplayFormat(_translate("Form", "yyyy-MM-dd"))
        self.dateEdit_6.setDisplayFormat(_translate("Form", "yyyy-MM-dd"))
        self.tab.setTabText(self.tab.indexOf(self.tab_2), _translate("Form", "Consumption"))
        self.label_17.setText(_translate("Form", "KW   "))
        self.label_18.setText(_translate("Form", "Total Price"))
        self.label_16.setText(_translate("Form", "Charge Date Between"))
        self.label_15.setText(_translate("Form", "Charge Id"))
        self.label_19.setText(_translate("Form", "Customer Id  "))
        self.comboBox_5.setItemText(1, _translate("Form", ">="))
        self.comboBox_5.setItemText(2, _translate("Form", "<="))
        self.comboBox_5.setItemText(3, _translate("Form", "=="))
        self.comboBox_6.setItemText(1, _translate("Form", ">="))
```

```
        self.comboBox_6.setItemText(2, _translate("Form", "<="))
        self.comboBox_6.setItemText(3, _translate("Form", "=="))
        self.dateEdit.setDisplayFormat(_translate("Form", "yyyy-MM-dd"))
        self.label_41.setText(_translate("Form", "And"))
        self.dateEdit_2.setDisplayFormat(_translate("Form", "yyyy-MM-dd"))
        self.checkBox.setText(_translate("Form", "Searching Date"))
        self.label_36.setText(_translate("Form", "* Search 3 edit line"))
        self.cl.setText(_translate("Form", "Clear"))
        self.btn_cha.setText(_translate("Form", "Search"))
        self.up_btn_cha.setText(_translate("Form", "Update"))
        self.Cha_add_btn.setText(_translate("Form", "Add Values"))
        self.Cha_del_btn.setText(_translate("Form", "Delete"))
        self.label_37.setText(_translate("Form", "Total Price"))
        self.cha_btn_up.setText(_translate("Form", "Save"))
        self.cha_cl_btn.setText(_translate("Form", "Close"))
        self.label_59.setText(_translate("Form", "Total used KW :"))
        self.label_61.setText(_translate("Form", "Total Price:"))
        self.btn_graph_2.setText(_translate("Form", "Graph"))
        self.table_cha.setSortingEnabled(False)
        self.label_56.setText(_translate("Form", "Add Value"))
        self.label_14.setText(_translate("Form", "Customer ID"))
        self.Cha_sent_btn.setText(_translate("Form", "Save"))
        self.Cha_cl_btn.setText(_translate("Form", "Close"))
        self.radioButton_5.setText(_translate("Form", "By ConsumptionId"))
        self.radioButton_3.setText(_translate("Form", "By 1 Month"))
        self.radioButton_4.setText(_translate("Form", "By 3 Months"))
        self.tab.setTabText(self.tab.indexOf(self.tab_3), _translate("Form", "Charge"))
        self.menuInfo.setTitle(_translate("Form", "Info"))
        self.menuAbout_CRM_sys.setTitle(_translate("Form", "About System"))
        self.actionHelp.setText(_translate("Form", "Help Online..."))
        self.actionHelp.setShortcut(_translate("Form", "Ctrl+H"))
        self.actionExit.setText(_translate("Form", "Exit"))
        self.actionExit.setShortcut(_translate("Form", "Ctrl+Q"))
        self.actionFullScreen.setText(_translate("Form", "FullScreen"))
        self.actionFullScreen.setShortcut(_translate("Form", "Ctrl+F"))
        self.actionUnique_items.setText(_translate("Form", "Unique item"))
        self.actionUpdate.setText(_translate("Form", "Update"))
        self.actionDelete.setText(_translate("Form", "Delete"))
        self.actionSystem.setText(_translate("Form", "System"))
        self.actionShearch.setText(_translate("Form", "Shearch"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QMainWindow()
    ui = Ui_Form()
```

```
ui.setupUi(Form)
Form.show()
sys.exit(app.exec_())
```