Ατομική Διπλωματική Εργασία

ΕΦΑΡΜΟΓΗ ΔΙΑΧΕΙΡΗΣΗΣ ΤΑΜΕΙΟΥ ΚΑΦΕΤΕΡΙΑΣ/ΜΠΑΡ (POS)

Όμηρος Κλείτου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2013

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Εφαρμογή διαχείρισης ταμείου καφετέριας/μπαρ (POS)

Όμηρος Κλείτου

Επιβλέπων Καθηγητής

Δρ.Παρασκευάς Ευριπίδου

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2013

Ευχαριστίες

Η φοίτηση μου στο Πανεπιστήμιο Κύπρου ολοκληρώνεται με τη παράδοση αυτής της Διπλωματικής Εργασίας, που αποτελεί τη δουλεία μιας ολόκληρης χρονιάς. Μιας χρονιάς, που αποτέλεσε ίσως τα πιο δύσκολα συναισθηματικά και ψυχολογικά, χρόνια της ζωής μου.

Θα ήθελα να ευχαριστήσω το κύριο Παρασκευά Ευριπίδου για την ευκαιρία καταρχήν που μου έδωσε να ασχοληθώ με αυτό το πολύ ενδιαφέρον project καθώς και την υποστήριξη που είχα από αυτόν καθόλη την διάρκεια της διπλωματικής.

Επίσης θα ήθελα να ευχαριστήσω ιδιαίτερα, την ομάδα της εταιρίας Starbucks, που με βοήθησε σε μέγιστο βαθμό στην επίτευξη αυτής της εργασίας. Τα παιδιά εκεί, μου παρείχαν την οποιαδήποτε πληροφορία αλλά και βοήθεια αλλά πάνω απ' όλα για τους ευχαριστώ για την στήριξη τους και τη βοήθεια τους σε πρακτικό επίπεδο.

Τέλος θέλω να ευχαριστήσω την οικογένεια μου που με στήριξε στις δύσκολες στιγμές.

Περίληψη

Η Ατομική Διπλωματική Εργασία, έχει ως στόχο της, τη δημιουργία ενός συστήματος POS(Point Of Sales) για μια καφετέρια/μπαράκι.

Συγκεκριμένα, το σύστημα αυτό έχει ως στόχο του, να εξυπηρετεί τον πελάτη με άνεση και ευκολία, και να απλοποιεί την δουλειά του χειριστή του, καθώς προτείνετε για καφετέριες με self service οργάνωση. Δηλαδή ο πελάτης κάνει την παραγγελία του, πληρώνει και εξυπηρετείται από τον υπάλληλο και αμέσως ετοιμάζεται η παραγγελία του, για να τα παραλάβει στο τέλος μπαρ.

Το σύστημα θα διαθέτει διεπηφάνεια από την οποία μπορεί να αλληλεπιδρά ο χρήστης (υπάλληλος καφετέριας) και να διαλέγει από τις διάφορες κατηγορίες ανάλογα με την παραγγελία του πελάτη, που θα απαρτίζονται από: Hot Drinks, Cold Drinks, Blended, Alcohol, Snacks και Other. Επιλέγοντας μέσα από αυτές τις κατηγορίες ανάλογα τα ροφήματα θα φορτώνονται σε ένα διπλανό παράθυρο όπου θα δημιουργείται η παραγγελία του πελάτη. Σε αυτό το παράθυρο θα εμφανίζονται τα προϊόντα μαζί με τις τιμές του καθενός ξεχωριστά και στο τέλος το συνολικό κόστος της παραγγελίας, έπειτα ο χειριστής του ταμείου θα είναι σε θέση να εκδώσει την απόδειξη . Επίσης ο χρήστης θα έχει την δυνατότητα να προσθέσει και άλλα προϊόντα στην βάση δεδομένων του και να φιλτράρει τα προϊόντα που έχει ήδη στην βάση του. Επιπρόσθετη ιδιότητα του συστήματος θα είναι να μπορεί να βλέπει τα transactions(αποδείξεις) που έχουν γίνει ανά πάσα στιγμή για κάθε προϊόν.

Περιεχόμενα

Κεφάλαιο **1 Εισαγωγή**..... **7**

1.1 Ορισμός καφετέριας 7

1.2 Ιστορία 8

1.3 Σημερινές καφετέριες 11

1.4 Εσπρέσο μπαρ 12

Κεφάλαιο **2 Πλάνο σχεδιασμού**..... **14**

2.1 Ανάλυση Απαιτήσεων 14

2.2 Συλλογή και εκτίμηση των απαιτήσεων 15

2.3 Λίστα με τις λειτουργικές απαιτήσεις του συστήματος 16

2.4 Λίστα με τις μη λειτουργικές απαιτήσεις του συστήματος 17

Κεφάλαιο **3 Προδιαγραφές**.....**19**

3.1 Use cases και Actors 19

3.2 Διαγράμματα ροής 26

3.3 Γλωσσάριο 28

Κεφάλαιο **4 Σχεδιασμός**.....**31**

4.1 Αποφάσεις σχεδιασμού 31

4.2 Διαγράμματα Ακολουθίας(Sequence diagrams) 35

4.3 Διαχείριση των δεδομένων 54

Κεφάλαιο **5 Υλοποίηση**..... **56**

5.1 Διαγράμματα κλάσεων 56

5.2 Τεχνολογίες που χρησιμοποιήθηκαν 59

5.3 Γλώσσες προγραμματισμού 60

5.4 Εφαρμογές 61

Κεφάλαιο **6 Εγκατάσταση και εγχειρίδιο εφαρμογής**..... **63**

6.1 Εγκατάσταση εφαρμογής 63

6.2 Εγχειρίδιο εφαρμογής 66

Κεφάλαιο **7 Συμπεράσματα** **74**

7.1 Ανασκόπηση εφαρμογής 74

7.2 Αξιολόγηση - Συμπεράσματα 75

Β ι β λ ι ο γ ρ α φ ί α **76**

Π α ρ ά ρ τ η μ α Α Απόσπασμα Κώδικα..... **Α-1**

Κεφάλαιο 1

Εισαγωγή

1.1 Ορισμός καφετέριας 7
1.2 Ιστορία 8
1.3 Σημερινές καφετέριες 11
1.4 Εσπρέσο μπαρ 12

1.1 Ορισμός καφετέριας

Καφετέρια και το Cafe είναι συναφείς όροι για μια εγκατάσταση που εξυπηρετεί πρωτίστως παρασκευασμένους καφέδες ή άλλα ζεστά ροφήματα. Μοιράζεται μερικά από τα ίδια χαρακτηριστικά ενός μπαρ ή εστιατόριου, αλλά είναι διαφορετικά από μια καφετέρια. Όπως υποδηλώνει το όνομα, καφετέρια επικεντρωθεί στην παροχή καφέ και τσάι, καθώς και ελαφριά σνακ. Πολλά καφέ στη Μέση Ανατολή και στη Δυτική Ασία περιοχές στο δυτικό κόσμο, προσφέρουν και shisha (Nargile στην ελληνική και τουρκική διάλεκτο), με γεύση καπνού που καταναλώνεται από ένα ναργιλέ. Espresso bar είναι ένα είδος καφετέριας που ειδικεύονται στην εξυπηρέτηση εσπρέσο και ροφήματα με βάση το εσπρέσο

.

Από πολιτιστική άποψη, οι καφετέριες σε μεγάλο βαθμό λειτουργούν ως κέντρα κοινωνικής αλληλεπίδρασης: η καφετέρια παρέχει σε κοινωνικά μέλη ένα μέρος για να συναθροίζονται, να μιλήσουν, να γράψουν, να διαβάσουν, να ψυχαγωγήσουν ο ένας τον άλλο, ή να περάσει απλά η ώρα, είτε ατομικά ή σε μικρές ομάδες των δύο ή τριών ατόμων. Μια καφετέρια λειτουργεί ως ένα άτυπο membership club για τα τακτικά μέλη της.

1.2 Ιστορία

Τον 17ο αιώνα, ο καφές εμφανίστηκε για πρώτη φορά στην Ευρώπη εκτός της Οθωμανικής Αυτοκρατορίας, και καφενεία ιδρύθηκαν και γρήγορα έγιναν δημοφιλής. Τα πρώτα καφενεία εμφανίστηκαν στη Βενετία το 1729, λόγω της κυκλοφορίας μεταξύ των La Serenissima και των Οθωμανών. Η πρώτη καφετέρια καταγράφεται το 1645. Το πρώτο καφενείο στην Αγγλία ιδρύθηκε στην Οξφόρδη το 1652, από ένα εβραίο άνδρα που ονομάζεται Jacob at the Angel στην ενορία του Αγίου Πέτρου στην Ανατολή. Ένα κτίριο στην ίδια περιοχή στεγάζει σήμερα ένα καφέ-μπαρ που ονομάζεται The Grand Cafe. Το Lane House καφετέρια της Βασίλισσας της Οξφόρδης, που ιδρύθηκε το 1654, εξακολουθεί να υφίσταται μέχρι και σήμερα. Το πρώτο καφενείο στο Λονδίνο άνοιξε το 1652 στο Alley του Αγίου Μιχαήλ, Cornhill. Ο δικαιούχος ήταν Pasqua Rosee, υπηρέτης ενός εμπόρου σε τουρκικά εμπορεύματα που ονομάζεται Daniel Edwards, ο οποίος εισήγαγε τον καφέ και βοήθησε την Rosee στο στήσιμο της εγκατάστασης στο Alley του Αγίου Μιχαήλ, Cornhill. Από το 1670 - 1685, ο αριθμός καφετεριών στο Λονδίνο άρχισε να πολλαπλασιάζεται, και άρχισαν επίσης να αποκτούν πολιτική σημασία λόγω της δημοτικότητάς τους ως τόποι συζήτησης. Από 1675, υπήρχαν πάνω από 3.000 καφενεία στην Αγγλία. Η Pasqua Rosee δημιούργησε επίσης την πρώτη καφετέρια στο Παρίσι το 1672 και πραγματοποίησε ένα μονοπώλιο στον καφέ σε ολόκληρη την πόλη μέχρι που ο Procopio Cuto άνοιξε το Procope Café το 1686. Αυτό το καφενείο υπάρχει ακόμα και σήμερα και ήταν ένα σημαντικό σημείο συνάντησης του Γαλλικού Διαφωτισμού. Το 1667, Kara Hamie, ένας πρώην Οθωμανικής Γενίτσαρος από την Κωνσταντινούπολη, άνοιξε το πρώτο καφενείο στο Βουκουρέστι (τότε πρωτεύουσα του Πριγκιπάτου της Βλαχίας), στο κέντρο της πόλης, σε αυτό που σήμερα βρίσκεται το κεντρικό κτίριο της Εθνικής Τράπεζας της Ρουμανίας. Η Αμερική είχε την πρώτη καφετέρια στη Βοστώνη, το 1676.

Ιστορία του πρώτου καφέ της Βιέννης, δήλωσε ότι ιδρύθηκε το 1683 από μια πολωνή κάτοικο, Jerzy Franciszek Kulczycki. Σε γενικές γραμμές, η πρώτη πολωνική καφετέρια ιδρύθηκε στη Βαρσοβία το 1724 από έναν από τους αυλικούς του βασιλιά της Πολωνίας Αύγουστος II Sass. Ωστόσο, η όλη κουλτούρα του καφέ ήταν το ίδιο διαδεδομένη στη χώρα κατά το δεύτερο μισό του 18ου αιώνα. Η πρώτη εγγεγραμμένη καφετέρια στη Βιέννη ιδρύθηκε από τον Έλληνα Johannes Theodat

(επίσης γνωστή ως Johannes Diodato) το 1685. Δεκαπέντε χρόνια αργότερα, τέσσερις Έλληνες έχουν καφετέρια και είχαν το προνόμιο να σερβίρουν καφέ.

Jerzy Franciszek Kulczycki άνοιξε το πρώτο καφενείο στη Βιέννη, με κόκκους καφέ που άφησαν οι Οθωμανοί Τούρκοι το 1683.
Αν και αργότερα ο Charles II προσπάθησε να καταστείλει τις καφετέριες στο Λονδίνο ως "μέρη όπου η συναντιούνται δυσαρεστημένοι, κ α την εξάπλωση σκανδάλων σχετικά με της Αυτού Μεγαλειότητας του και τους υπουργούς του», το κοινό συνέρρεαν σε αυτά. Για αρκετές δεκαετίες μετά την αποκατάσταση, οι Wits μαζεύονταν στην καφετέρια του John Dryden, στο Russell Street, Covent Garden. Οι καφετέριες ήταν μεγάλες κοινωνικές αναμοχλευτήρες, ανοικτές σε όλους τους άνδρες και αδιάφορες για την κοινωνική θέση, και ως αποτέλεσμα βοήθησε με την ισότητα και τον ρεπουμπλικανισμό. Γενικότερα, οι καφετέριες έγιναν χώροι συνάντησης όπου οι επιχειρήσεις θα μπορούσαν να ασκούνται, ειδήσεις ανταλλάσσονται και τα London Gazette (ανακοινώσεις της κυβλερνησης) να διαβάζονται. LLOYD'S του Λονδίνου είχε τις ρίζες της σε μια κ αφετέρια που διευθυν όταν από τον Edward Lloyd, όπου ασφαλιστές της ασφάλισης των πλοίων συναντήθηκαν για να κάνουν επιχειρήσεις. Με το 1739, υπήρχαν 551 καφετέριες στο Λονδίνο. Κάθε μια προσέλκυε ένα συγκεκριμένο πελατολόγιο, χωρίζον ται με βάση το επάγγελμα ή τη στάση, όπως Συντηρητικοί και Whigs, και stockjobbers, οι έμποροι και οι δικηγόροι, οι βιβλιοπώλες και οι συγγραφείς, οι άνδρες της μόδας ή τα "CITS" από το παλιό κέντρο της πόλης.

Η απαγόρευση των γυναικών από τις καφετέριες δεν ήταν παγκόσμια, αλλά φαίνεται να ήταν κοινή στην Ευρώπη. Στη Γερμανία οι γυναίκες σύχναζαν να επισκέπτονται τις καφετέριες, αλλά στην Αγγλία και στη Γαλλία είχαν απαγορευτεί. Σε ένα πολύ γνωστό παριζιάνικου καφέ το 1700, οι κύριοι κρέμαζαν τα καπέλα τους με μανταλάκια και να κάθονταν σε μεγάλα κοινά τραπέζια στρωμένα με χαρτιά και γραφική ύλη.

Η παραδοσιακή ιστορία της προέλευσης του βιεννέζικο καφέ ξεκινά με τους μυστηριώδεις σάκους των πράσινων κόκκων που άφισαν πίσω οι Τούρκοι όταν νικήθηκαν στη μάχη της Βιέννης το 1683. Όλα τα σακιά του καφέ χορηγήθηκαν στον νικηφόρο βασιλιά της Πολωνίας Jan III Sobieski, ο οποίος με τη σειρά του τα έδωσε σε έναν από τους αξιωματικούς του, Jerzy Franciszek Kulczycki. Ο Kulczycki ξεκίνησε τη πρώτη καφετέρια στη Βιέννη με τον σωρό. Ωστόσο, είναι πλέον ευρέως

αποδεκτό ότι την πρώτη καφετέρια είχε πράγματι ανοιχτεί από έναν Έλληνα έμπορο που ονομάζεται Johannes Diodato.

Η καφετέρια του Jonathan το 1698 είδε την εισαγωγή των μετοχών και των βασικών προϊόντων που εξελίχθηκε στο London Stock Exchange. Η καφετέρια Lloyd, προσφερόταν ως χώρος για τους εμπόρους και τους φορτωτές, για να συζητήσουν ευκαιρίες ασφάλισης, και είχε οδηγήσει στη δημιουργία του Lloyd της ασφαλιστικής αγοράς του Λονδίνου, το Lloyd Register, καθώς και άλλες συναφείς επιχειρήσεις.

Κατά τη διάρκεια του 18ου αιώνα, οι παλαιότερες σωζόμενες καφετέριες στην Ιταλία που ιδρύθηκαν ήταν: Caffè Florian στη Βενετία, Antico Caffè Greco στη Ρώμη, Caffè Pedrocchi στην Πάδοβα, Caffè dell'Ussero στην Πίζα και Caffè Fiorio στο Τορίνο.

Στη βικτοριανή Αγγλία, οι οργανώσεις που αναπτύσσουν την αντιαλκοολική στάση είχαν δημιουργήσει καφετέριες για τα λαϊκά στρώματα, ως τόπος χαλάρωσης χωρίς αλκοόλ, μια εναλλακτική λύση για τα Public House (pub). Κατά τον 19ο και 20ο αιώνα, οι καφετέριες ήταν συνήθως σημείο συνάντησης για τους συγγραφείς και τους καλλιτέχνες, σε όλη την Ευρώπη.

1.3 Σημερινές καφετέριες

Στις περισσότερες ευρωπαϊκές χώρες, όπως η Αυστρία, η Δανία, τη Γερμανία, τη Νορβηγία, τη Σουηδία, την Πορτογαλία και άλλους, ο όρος καφέ σημαίνει ένα εστιατόριο που σερβίρει κυρίως καφέ, καθώς και γλυκά όπως κέικ, τάρτες, πίτες, δανέζικα αρτοσκευάσματα, ή κουλούρι. Πολλές καφετέριες σερβίρουν επίσης ελαφριά γεύματα, όπως σάντουιτς. Ευρωπαϊκά καφέ συνήθως έχουν τραπεζάκια στο πεζοδρόμιο, καθώς και σε εσωτερικούς χώρους. Μερικά καφέ σερβίρουν επίσης αλκοολούχα ποτά, ιδίως στις χώρες της Νότιας Ευρώπης.
Στο Ηνωμένο Βασίλειο και την Δημοκρατία της Ιρλανδίας ένα καφέ (με τόνο, «café») είναι παρόμοια με αυτά των άλλων ευρωπαϊκών χωρών, ενώ ένα καφέ (χωρίς τόνο, και συχνά προφέρεται "caff") είναι πιθανό να είναι τύπου fast food, που σερβίρουν κυρίως τηγανητά τρόφιμα, ιδίως πιάτα πρωινού.

Στην Ολλανδία και το Βέλγιο, ένα καφέ είναι ισοδύναμο με ένα μπαρ, καθώς επίσης πωλεί οινοπνευματώδη ποτά. Στην Ολλανδία τα Koffiehuis (NL), σερβίρουν καφέ, ενώ ένα κατάστημα καφέ (χρησιμοποιώντας την αγγλική λέξη, coffee shop) πωλεί μαλακά ναρκωτικά (κάνναβη και χασίς) και γενικά δεν επιτρέπεται να πωλούν αλκοολούχα ποτά.

Στη Γαλλία οι περισσότερες καφετέριες λειτουργούν ως εστιατόρια την ημέρα, και μπαρ το βράδυ. Εν γένει δεν έχουν αρτοσκευάσματα εκτός κατά τη διάρκεια των πρωινών, όταν ένα κρουασάν ή pain au chocolat μπορούν να αγοραστούν με τον πρωινό καφέ.

Στην Ιταλία τα καφέ είναι παρόμοια με εκείνα που βρέθηκαν στη Γαλλία και είναι γνωστά ως bar. Σερβίρουν συνήθως μια ποικιλία από καφέ εσπρέσο, κέικ και αλκοολούχα ποτά. Μπαρ στα κέντρα των πόλεων έχουν συνήθως διαφορετικές τιμές για την κατανάλωση στο μπαρ και την κατανάλωση σε ένα τραπέζι.

1.4 Εσπρέσο μπαρ

Τα εσπρέσο μπαρ είναι ένας τύπος καφετέριας που ειδικεύεται σε ροφήματα καφέ εσπρέσο. Κατάγονται από την Ιταλία, το εσπρέσο μπαρ έχει εξαπλωθεί σε όλο τον κόσμο με διάφορες μορφές. Παραδείγματα που είναι διεθνώς γνωστά είναι Starbucks Coffee, με έδρα το Σιάτλ, Ουάσιγκτον, ΗΠΑ και Costa Coffee, με έδρα το Dunstable, Ηνωμένο Βασίλειο (η πρώτη και η δεύτερη μεγαλύτερη αλυσίδα καφετέριας, αντίστοιχα), αν και υπάρχουν πολλά εσπρέσο μπαρ υπό κάποια μορφή σε ένα μεγάλο μέρος του κόσμου.

Το εσπρέσο μπαρ συνήθως επικεντρώνεται γύρω από ένα μεγάλο πάγκο με μια υψηλής απόδοσης μηχανή εσπρέσο (συνήθως αυτόματες ή ημιαυτόματες τύπου αντλία μηχανή, αν και μερικές φορές ένας χειροκίνητος μοχλός-και-εμβόλου) και μια βιτρίνα που περιέχει γλυκά και περιστασιακά είδη αλμυρών, όπως σάντουιτς. Στο παραδοσιακό ιταλικό μπαρ, οι πελάτες είτε παραγγέλλουν στο μπαρ και καταναλώνουν τα ροφήματα τους στο πόδι ή, εάν επιθυμούν να καθίσουν και να εξυπηρετηθούν, χρεώνονται συνήθως υψηλότερη τιμή. Σε μερικά μπαρ υπάρχει μια επιπλέον χρέωση για ροφήματα που σερβίρονται σε ένα τραπέζι έξω. Σε άλλες χώρες, ιδίως οι Ηνωμένες Πολιτείες, το καθιστικό για τους πελάτες να χαλαρώσουν και να εργαστούν παρέχονται δωρεάν. Μερικά μπαρ εσπρέσο πωλούν επίσης σύνεργα καφέ, γλυκά, ακόμα και μουσική. Στην Βόρεια Αμερική τα εσπρέσο μπαρ ήταν πρωτοπόρα στην υιοθέτηση των δημόσιων σημείων πρόσβασης WiFi για την παροχή υπηρεσιών Internet για τους ανθρώπους που κάνουν εργασία σε φορητούς υπολογιστές στις εγκαταστάσεις τους.

Οι προσφορές σε ένα τυπικό εσπρέσο μπαρ είναι γενικά αρκετά ιταλικού τύπου. Biscotti, cannoli και pizzelle είναι κοινά παραδοσιακά συνοδευτικά σε ένα caffe latte ή καπουτσίνο. Μερικά αναβαθμισμένα εσπρέσο μπαρ προσφέρουν ακόμα και αλκοολούχα ποτά, όπως grappa και sambuca. Παρ 'όλα αυτά, παραδοσιακά αρτοσκευάσματα δεν είναι πάντοτε απολύτως ιταλικού τύπου και κοινές προσθήκες περιλαμβάνουν scones, muffins, κρουασάν, ακόμα και ντόνατς. Υπάρχει συνήθως μια μεγάλη ποικιλία τσαγιών, καθώς και τα βορειοαμερικάνικα εσπρέσο μπαρ είναι υπεύθυνα για την διάδοση του Ινδικού τσαγιού spiced tea masala chai. Παγωμένα ροφήματα είναι επίσης δημοφιλής σε ορισμένες χώρες, συμπεριλαμβανομένων τόσο

παγωμένο τσάι και παγωμένο καφέ, καθώς και blended ροφήματα όπως frappucino Starbucks ».

Ένας εργαζόμενος σε ένα εσπρέσο μπαρ αναφέρεται ως barista. Ο barista είναι μια εξειδικευμένη θέση που απαιτεί εξοικείωση με τα ροφήματα που γίνονται, μια εύλογη εγκατάσταση με κάποιον εσωτερικό εξοπλισμό, καθώς και τις συνήθεις δεξιότητες εξυπηρέτησης πελατών.

Κεφάλαιο 2

Πλάνο σχεδιασμού

2.1 Ανάλυση Απαιτήσεων 14
2.2 Συλλογή και εκτίμηση των απαιτήσεων 15
2.3 Λίστα με τις λειτουργικές απαιτήσεις του συστήματος 16
2.4 Λίστα με μη τις λειτουργικές απαιτήσεις του συστήματος 17

2.1 Ανάλυση Απαιτήσεων

Ο ορισμός των απαιτήσεων software μιας εφαρμογής είναι μια συμφωνία η οποία διαβεβαιώνει πως και ο πελάτης και οι αναλυτές του συστήματος κατανοούν πλήρως τις απαιτήσεις του συστήματος προς ανάπτυξη, υπό την ίδια οπτική γωνία. Ένα έγγραφο SRS (Software Requirements Specification) ορίζεται συχνά και ως το "πατρικό" έγγραφο διότι όλα τα έγγραφα που ακολουθούν κατά την ανάπτυξη του Project (όπως αποφάσεις αρχιτεκτονικής, σχεδιασμός, υλοποίηση, έλεγχος και εκτιμήσεις) βασίζονται σε αυτό το "πατρικό" έγγραφο SRS.

Είναι πολύ σημαντικό να αναφέρουμε πως το έγγραφο SRS περιέχει μόνο τις λειτουργικές και μη λειτουργικές απαιτήσεις του συστήματος. Οι λειτουργικές απαιτήσεις του συστήματος ικανοποιούνται από τις περιπτώσεις χρήσης (Use Cases) του συστήματος και συνήθως ορίζουν τι λειτουργίες μπορεί να υλοποιήσει το σύστημα. Οι μη λειτουργικές απαιτήσεις περιγράφουν τα γενικότερα χαρακτηριστικά που πρέπει να έχει το σύστημα (κριτήρια οικονομικά ,δομικά, hardware,κλπ). Ένα έγγραφο SRS δεν προσφέρει προτάσεις σχεδιασμού ούτε πιθανές λύσεις σε θέματα τεχνολογίας.

Ένα έγγραφο SRS καλοσχεδιασμένο και καλογραμμένο ικανοποιεί τα παρακάτω τέσσερα κριτήρια:

• Αποτελεί ένα feedback για τον πελάτη. Ένα έγγραφο SRS είναι η εγγύηση του πελάτη ότι οι δημιουργοί του συστήματος καταλαβαίνουν τα προβλήματα που πρέπει να επιλύσουν και την συμπεριφορά που πρέπει να έχει το σύστημα για να επιλύσει αυτά τα προβλήματα.

• Τεμαχίζει το πρόβλημα σε μικρότερα μέρη. Μια καλο-δομημένη λίστα με τις απαιτήσεις software του συστήματος. οργανώνει την πληροφορία, οριοθετεί το πρόβλημα και συγκεκριμενοποιεί τις ιδέες.

• Εισάγει τον αναλυτή στον ορισμό του σχεδιασμού του συστήματος.

• Στην φάση των δοκιμών και εκτιμήσεων του συστήματος, μπορεί να χρησιμοποιηθεί ως σημείο σύγκρισης για να ελεγχθεί αν το σύστημα ικανοποιεί τις αρχικές απαιτήσεις.

2.2 Συλλογή και εκτίμηση των απαιτήσεων

Η συλλογή των απαιτήσεων ήταν μια πολύ σημαντική φάση, για την κατανόηση του τι ακριβώς έπρεπε να υλοποιηθεί και ποιες λειτουργίες έπρεπε να έχει η εφαρμογή.

Για να συγκεντρωθούν οι λειτουργικές απαιτήσεις του συστήματος έγιναν τα ακόλουθα:

- Επίσκεψη διάφορων self service καφετεριών.
- Συνάντηση με διευθυντή υποκαταστήματος καφετέριας Starbucks.
- Συνάντηση με υποδιευθυντή υποκαταστήματος καφετέριας Starbucks.
- Παρακολούθηση εργασίας υπαλλήλων καφετέριας.

Οι συναντήσεις βοήθησαν πολύ στην κατανόηση του τρόπου λειτουργίας των καφετεριών και στο ποιες βασικές λειτουργίες θα έπρεπε να έχει η εφαρμογή. Κύριο ρόλο έπαιξε η γνώμη των υπαλλήλων που χρησιμοποιούν καθημερινά τέτοια εφαρμογή. Σημαντικός παράγοντας για όλους ήταν η ευχρηστία της εφαρμογής, ώστε να τους βοηθά στην εξυπηρέτηση των πελατών χωρίς λάθη και στην ταχύτητα της διεκπεραίωσης της παραγγελίας.

Στην φάση αυτή μελετήθηκε το υπάρχον σύστημα που χρησιμοποιείται στην καφετέρια και αντιλήφθηκε ότι η πολυπλοκότητα του συστήματος είναι αχρείαστη. Παρόλα αυτά μελετήθηκε η δομή του, οι λειτουργίες του και συζητήθηκε ποια μέρη τις εφαρμογής φαίνονται χρήσιμα και ποια όχι.

2.3 Λίστα με τις λειτουργικές απαιτήσεις του συστήματος

Add Product

Ο χρήστης πρέπει να έχει την δυνατότητα να κάνει τα πιο κάτω:

- Εισάγει νέο προϊόν.
- Εισάγει περιγραφή προϊόντος.
- Εισάγει τιμή.
- Εισάγει εικόνα προϊόντος.

View Products

- Ο χρήστης πρέπει να έχει την δυνατότητα να δει λίστα με τα προϊόντα που έχει σε ολόκληρη την βάση του.
- Ο χρήστης πρέπει να μπορεί να φιλτράρει την λίστα με τα προϊόντα ανάλογα με τις κατηγορίες προϊόντων που έχει η εφαρμογή.

Open POS

- Το σύστημα πρέπει να έχει την δυνατότητα να ανοίγει καινούργιο παράθυρο για εκτέλεση παραγγελίας.
- Το σύστημα πρέπει να εμφανίζει όλες τις κατηγορίες προϊόντων και όλα τα προϊόντα σε κάθε κατηγορία.
- Το σύστημα πρέπει να αφήνει τον χρήστη να επιλέξει ανάμεσα από τις διάφορες κατηγορίες και τα προϊόντα, και να τα εμφανίζει σε λίστα παραγγελίας με τις τιμές του κάθε προϊόν.
- Το σύστημα πρέπει να έχει την δυνατότητα να αφήνει τον χρήστη να σβήνει προϊόντα από την λίστα παραγγελίας.
- Το σύστημα πρέπει να εμφανίζει το τελικό ποσό παραγγελίας.
- Το σύστημα πρέπει να δίνει την δυνατότητα πληρωμής παραγγελίας και να εμφανίζει τυχόν ρέστα που πρέπει να δοθούν.
- Το σύστημα πρέπει να παράγει αυτόματα την απόδειξη παραγγελίας.

Display Sales

- Το σύστημα πρέπει να έχει την δυνατότητα να εμφανίζει τις πωλήσεις κάθε προϊόντος.

Report Sales

- Το σύστημα πρέπει να έχει την δυνατότητα να εμφανίζει report με τις πωλήσεις ανά προϊόν και ανά ημερομηνία και ώρα που έχει πωληθεί.

2.4 Λίστα με μη τις λειτουργικές απαιτήσεις του συστήματος

Οι μη λειτουργικές απαιτήσεις του συστήματος ορίζουν ιδιότητες και περιορισμούς του συστήματος, και μπορούν να κατηγοριοποιηθούν με τον ακόλουθο τρόπο:

Χρηστικότητα (Usability)

Οι χρήστες του συστήματος έχουν βασική μόρφωση και αλλάζουν συνεχώς. Είναι πολύ σημαντικό για το σύστημα να είναι εύκολο στη χρήση, απλό στην εκμάθηση και διαδραστικό με τον χρήστη. Πρέπει να είναι σχεδιασμένο με γραφικά εύκολα να κατανοηθούν και καλά δομημένα.

• Οποιαδήποτε περίπτωση χρήσης, δεν πρέπει να χρειάζεται πάνω από τρία επίπεδα σελίδων μέχρι να ολοκληρωθεί.
• Οι χρήστες του συστήματος πρέπει να προστατεύονται από το να διαπράττουν λάθη. Το σύστημα πρέπει να προειδοποιεί τον χρήστη όταν αυτός εισάγει λάθος δεδομένα.
• Είναι πολύ σημαντικό το σύστημα να είναι φορητό και εύκολο να εγκατασταθεί(δηλαδή να είναι εύκολο να μετακινηθεί σε διαφορετικά hardware ή σε διαφορετικά λειτουργικά συστήματα).

Εκτέλεση

• Το σύστημα πρέπει να είναι μια windows εφαρμογή.
• Η εφαρμογή δεν πρέπει να αργή να ανταποκριθεί πάνω από 8 δευτερόλεπτα, κατά την διάρκεια μιας φυσιολογικής χρήσης του συστήματος.

Αξιοπιστία

• Το σύστημα πρέπει να είναι προσβιβάσιμο το 99.999% των φορών που οι χρήστες ανοίγουν την εφαρμογή.

Ασφάλεια

• Αυτό το σύστημα θα διαχειρίζεται δεδομένα που πρέπει να είναι προστατευμένα όπως τα προσωπικά δεδομένα των πελατών, τους λογαριασμούς των υπαλλήλων, τις διαμονές των πελατών και τις κινήσεις χρήματος στα ταμεία του ξενοδοχείου.
• Η βάση δεδομένων πρέπει να είναι πολύ καλά προστατευμένη. Το σύστημα πρέπει να κάνει καθημερινά backups καθώς και να έχει όλη την πληροφορία της βάσης δεδομένων επαναλαμβανόμενη σε διαφορετικά σημεία εντός και εκτός συστήματος.

Hardware

• Για την πλήρη λειτουργία του συστήματος χρειάζεται ένας υπολογιστής.

1. Pentium IV
2. Σκληρός δίσκος 80GB
3. 256 MB μνήμης RAM

Οικονομικό κόστος

• Ο προϋπολογισμός της διπλωματικής, πρέπει να είναι μειωμένος στο ελάχιστο. Είναι σημαντικό να χρησιμοποιηθεί λογισμικό open source, ελεύθερης διανομής και αδειών για να μειωθεί το κόστος συντήρησης του προϊόντος

Προσαρμοστικότητα

• Το σύστημα πρέπει να είναι εύκολο να δεχθεί μελλοντικές μετατροπές και επεκτάσεις στον κωδικό του.
• Το σύστημα θα είναι εκτεθειμένο σε σκληρές περιβαλλοντικές συνθήκες όπως υψηλές θερμοκρασίες και πολύ υγρασία.

Κεφάλαιο 3

Προδιαγραφές

3.1 Use Cases και Actors 19
3.2 Διάγραμμα ροής 26
3.3 Γλωσσάριο 28

Σε αυτό το κεφάλαιο θα περιγράψουμε τι πρέπει να κάνει το προς ανάπτυξη σύστημα, χωρίς να μπούμε στην διαδικασία να εξηγήσουμε πώς θα το κάνει. Με την βοήθεια της γλώσσας UML (Unified Modeling Language) θα παρουσιάσουμε τους actor και τις περιπτώσεις χρήσης (use cases) του συστήματος. Η φιλοσοφία των use cases επικεντρώνεται στην συμπεριφορά των εισόδων και εξόδων του συστήματος και αφήνει τις λεπτομέρειες του σχεδιασμού της εφαρμογής στους αναλυτές και σχεδιαστές του συστήματος.

3.1 Use Cases και Actors

Ένας "Actor" παριστάνει τον γενικό ρόλο ενός χρήστη. Οι χρήστες μπορούν να αναπαραστήσουν πολλούς actors του συστήματος. Μια περίπτωση χρήσης περιγράφει την αλληλεπίδραση μεταξύ ενός actor και του συστήματος. Μια περίπτωση χρήσης πάντα περιέχει την περιγραφή ενός περιβάλλοντος όπου ο actor παρέχει μια είσοδο στο σύστημα και το σύστημα επιστρέφει μια έξοδο. Μέσα σε κάθε περίπτωση χρήσης πάντα υπάρχει η περιγραφή ενός βασικού σεναρίου (happy path). Τα υπόλοιπα σενάρια που περιγράφονται ονομάζονται εναλλακτικά σενάρια ή ροές.

Το μοντέλο περιπτώσεων χρήσης (use case model) περιγράφει την λειτουργικότητα του συστήματος, όπως το σύστημα φαίνεται σε έναν εξωτερικό παρατηρητή. Η γλώσσα UML (Unified Modeling Language) διαθέτει ειδικά σύμβολα για τους actors και τις περιπτώσεις χρήσεως. Μια απλή έλλειψη συμβολίζει τις περιπτώσεις χρήσης του συστήματος σε σχέση με τους actors που συμμετέχουν σε αυτές τις περιπτώσεις χρήσης.

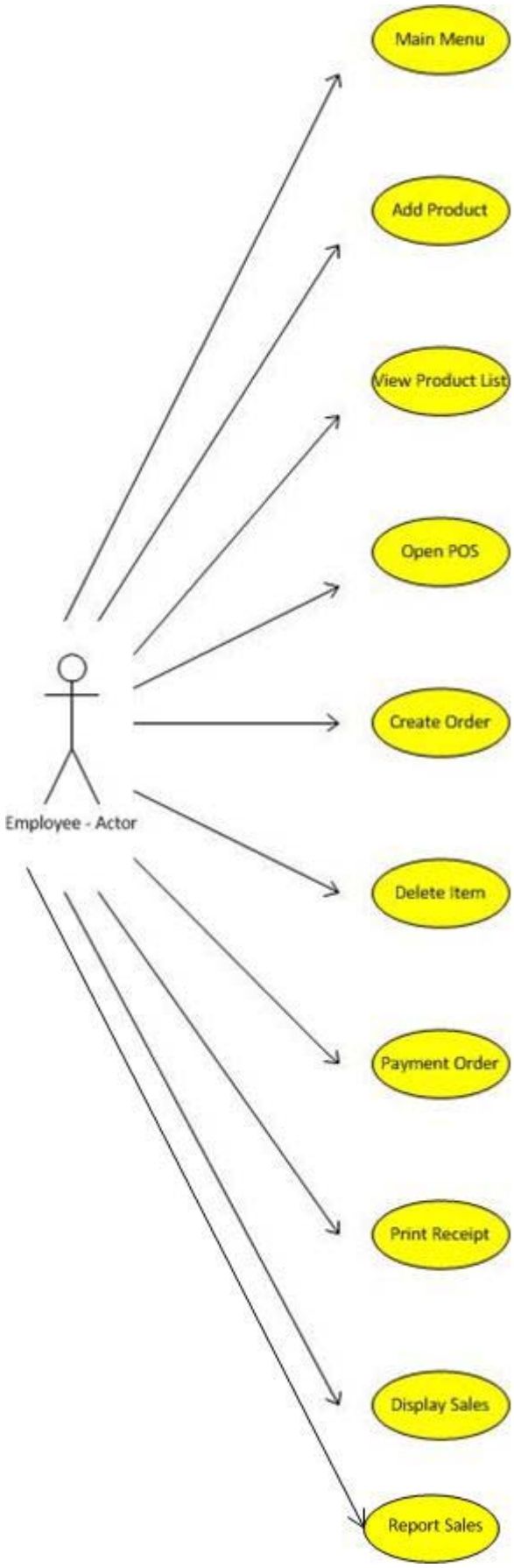
Κατασκευάστηκε ένα διαγράμματα των περιπτώσεων χρήσεως με βάση ένα περίγραμμα που ορίζει ένα όριο γύρω από το σύστημα. Μέσα στο περίγραμμα βρίσκονται όλες οι περιπτώσεις χρήσεις και όλοι οι actors βρίσκονται έξω.

Ακολούθως παρουσιάζονται όλες οι περιπτώσεις χρήσεως που υλοποιεί το σύστημα.

Λίστα περιπτώσεων χρήσεως:

1. Main Menu
2. Προσθήκη Προϊόντος
3. Λίστα Προϊόντων
4. Άνοιγμα POS
5. Δημιουργία Παραγγελίας
6. Ακύρωση Προϊόντος από λίστα παραγγελίας
7. Πληρωμή Παραγγελίας
8. Εκτύπωση Απόδειξης
9. Γραφική απεικόνιση αποδείξεων (transactions) ανά προϊόν
10. Report πωλήσεων

Ακολούθως παρουσιάζονται τα διαγράμματα όλων των περιπτώσεων χρήσεως του συστήματος καθώς και μια σύντομη περιγραφή. Παραθέτουμε τα διαγράμματα όλων των περιπτώσεων χρήσεως ώστε ο αναγνώστης να έχει μια συνολική εικόνα των λειτουργιών που φέρνει εις πέρας το σύστημα. Αυτά καταγράφτηκαν σύμφωνα με τις συνεντεύξεις που πάρθηκαν από υπαλλήλους και υπευθύνων καταστημάτων καφετεριών.



Εικόνα 1: Διάγραμμα των περιπτώσεων χρήσεως

1. **Main Menu**

Actor	Σύστημα
Ο χρήστης μπορεί να κάνει όποια επιλογή του δίνει το κύριο μενού.	Το σύστημα δίνει την δυνατότητα στον χρήστη να επιλέξει από το κύριο μενού τις ακόλουθες επιλογές: • Add Product • View Products • Open POS • Display Sales • Report Sales

2. **Προσθήκη Προϊόντος**

Actor	Σύστημα
Ο χρήστης θέλει να εισάγει καινούργιο προϊόν στην βάση δεδομένων.	Το σύστημα εμφανίζει μια φόρμα εισαγωγής καινούργιου προϊόντος την οποία καλείται να συμπληρώσει ο χρήστης.
Ο χρήστης συμπληρώνει την φόρμα εισαγωγής καινούργιου προϊόντος, εισάγοντας περιγραφή, τιμή, κατηγορία προϊόντος, και ανέβασμα φωτογραφίας του προϊόν.	Το σύστημα ενημερώνει την βάση δεδομένων.

3. Λίστα Προϊόντων

Actor	Σύστημα
Ο χρήστης θέλει να δει λίστα με όλα τα προϊόντα που υπάρχουν στην βάση.	Το σύστημα εμφανίζει στον χρήστη λίστα με όλα τα προϊόντα που υπάρχουν στην βάση, εμφανίζοντας περιγραφή, τιμή και εικόνα προϊόντος.
Ο χρήστης θέλει να δει προϊόντα συγκεκριμένης κατηγορίας.	Το σύστημα δίνει την δυνατότητα στον χρήστη να φιλτράρει την λίστα προϊόντων ανάλογα με τις κατηγορίες: • Hot Drinks • Cold Drinks • Blended • Alcohol • Other • Snacks

4. Άνοιγμα POS

Actor	Σύστημα
Ο χρήστης Θέλει να ανοίξει την οθόνη POS του ταμείου για να παραλάβει παραγγελία.	Το σύστημα εμφανίζει την οθόνη ταμείου με όλες τις επιλογές μια παραγγελίας.

5. <u>Δημιουργία Παραγγελίας</u>

Actor	Σύστημα
Ο χρήστης Θέλει να δημιουργήσει μια παραγγελία.	Το σύστημα δίνει την δυνατότητα στον χρήστη να επιλέξει από τα διάφορα προϊόντα τα οποία εισήγαγε στην βάση ανάλογα με τις κατηγορίες των προϊόντων.
Ο χρήστης επιλέγει προϊόντα από τις διάφορες κατηγορίες.	Τα προϊόντα εισάγονται σε μια διπλανή λίστα προϊόντων της παραγγελίας μαζί με την περιγραφή και την τιμή τους.

6. <u>Ακύρωση προϊόντως από λίστα παραγγελίας</u>

Actor	Σύστημα
Ο χρήστης θέλει να ακυρώσει κάποιο προϊόν από την λίστα παραγγελίας.	Το σύστημα δίνει την δυνατότητα στον χρήστη να σβήσει κάποιο προϊόν το οποίο έχει επιλέξει στην λίστα παραγγελίας, και να αφαιρεθεί το προϊόν από την λίστα με την περιγραφή και την τιμή του και ακολούθως δεν θα υπολογιστεί στο τελικό ποσό της παραγγελίας.

7. Πληρωμή Παραγγελίας

Actor	Σύστημα
Ο χρήστης θέλει να κλίσει την παραγγελία με την πληρωμή της.	Το σύστημα εμφανίζει το τελικό ποσό της παραγγελίας το οποίο πρέπει να πληρωθεί ο χρήστης.
Ο χρήστης εισάγει το ποσό το οποίο παραλαμβάνει από τον πελάτη.	Το σύστημα υπολογίζει τυχόν ρέστα που πρέπει να επιστραφούν στον πελάτη ή υπολειπόμενο ποσό που πρέπει να πληρωθεί.

8. Εκτύπωση Απόδειξης

Actor	Σύστημα
Ο χρήστης θέλει να εκτυπώσει απόδειξη για την παραγγελία που μόλις παράλαβε.	Το σύστημα αυτόματα μετά την πληρωμή της παραγγελίας εμφανίζει επιλογή για την εκτύπωση της απόδειξης, και εκτυπώνει την απόδειξη.

9. Γραφική απεικόνιση αποδείξεων (transactions) ανά προϊόν

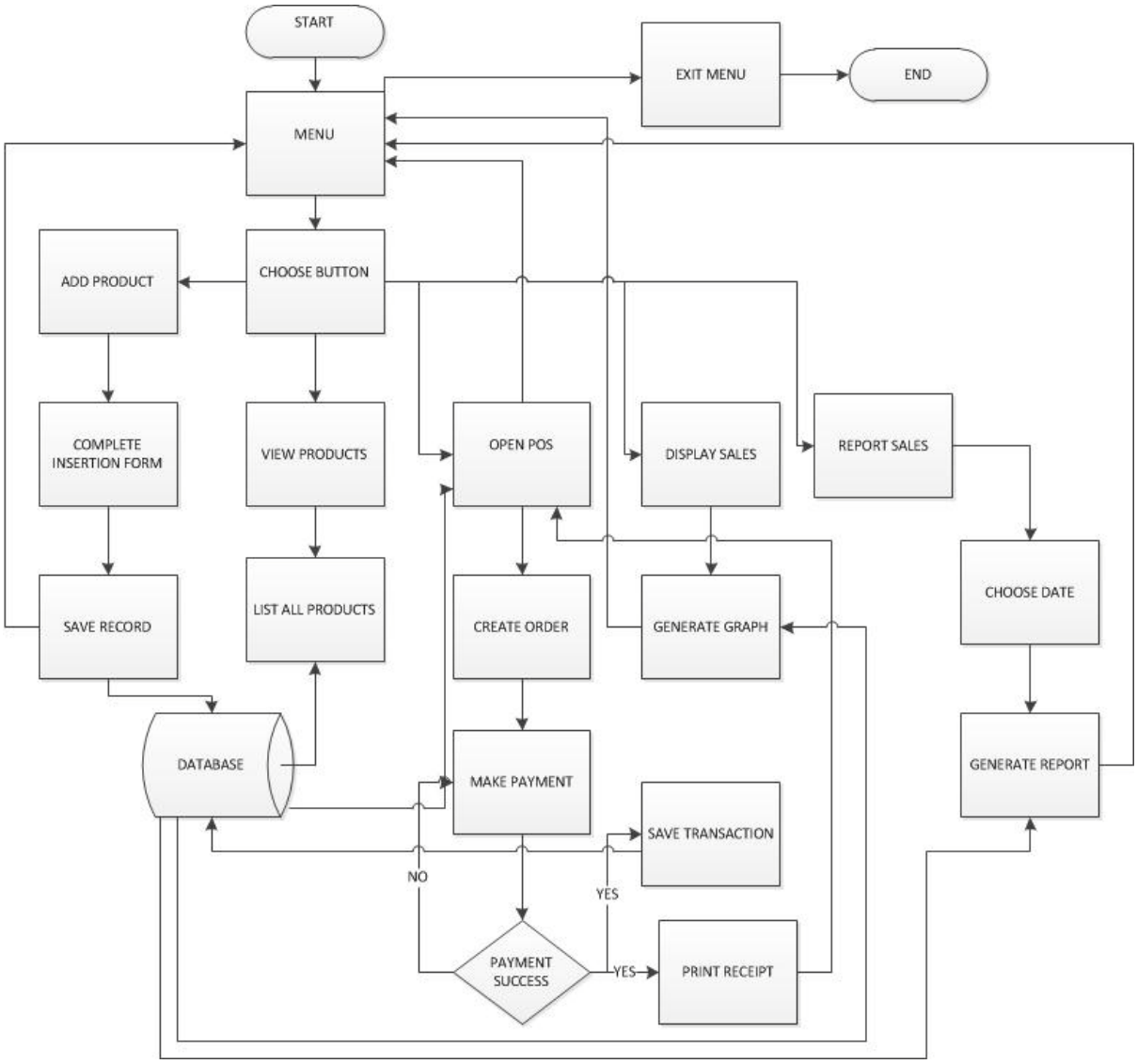
Actor	Σύστημα
Ο χρήστης θέλει να δει τα transactions (αποδείξεις) που έχει κάνει κάθε προϊόν .	Το σύστημα εμφανίζει γραφική απεικόνιση των transaction (αποδείξεων) που έχει κάνει κάθε προϊόν.

10. Report Πωλήσεων

Actor	Σύστημα
Ο χρήστης θέλει να δει τα transactions (αποδείξεις) που έχει κάνει κάθε προϊόν ανά ημερομηνία και ώρα της ημέρας για να ελέγξει παλιά με καινούργια transactions καθώς και να δει ποιες είναι οι “busy” ώρες της ημέρας.	Το σύστημα εμφανίζει report ανάλογα με την ημερομηνία που θα επιλέξει ο χρήστης με την ποσότητα των προϊόντων που πωληθήκαν αναγράφοντας συνολικά την ποσότητα και τις τιμές καθώς και τον μέσο όρο αλλά και τον συνολικό αριθμό των transactions.

3.2 Διάγραμμα ροής

Διάγραμμα ροής (flowchart) είναι ένα κοινού τύπου διάγραμμα που αναπαριστά έναν αλγόριθμο ή μια διαδικασία, δείχνοντας τα βήματα ως κουτιά διαφόρων ειδών που συνδέονται μεταξύ τους με βέλη. Αυτή η διαγραμματική παρουσίαση μπορεί να δώσει λύση βήμα προς βήμα σε ένα γνωστό πρόβλημα. Τα δεδομένα αναπαριστώνται σε κουτιά και τα βέλη δείχνουν τη ροή των δεδομένων. Τα διαγράμματα ροής χρησιμοποιούνται στην ανάλυση, το σχεδιασμό, την τεκμηρίωση ή τον έλεγχο μιας διαδικασίας ή ενός προγράμματος σε διάφορα πεδία



Εικόνα 2: Διάγραμμα ροής

3.3 Γλωσσάριο

Ένα γλωσσάριο είναι απλώς μια συλλογή από όρους και ορισμούς, με τους οποίους συνεννοείται η ομάδα ανάπτυξης του λογισμικού, ενώ δημιουργεί και χρησιμοποιεί τις απαιτήσεις του συστήματος. Ένα καλογραμμένο γλωσσάριο βοηθάει τα καινούργια μέλη της ομάδας καθώς και τον οποιοδήποτε επιθυμεί να μελετήσει την διαδικασία ανάπτυξης του συστήματος, να καταλάβει γρήγορα τις λειτουργίες του νέου συστήματος. Στην αρχή του project όλοι οι βασικοί όροι και ορισμοί είναι κατανοητοί, αλλά καθώς το project αναπτύσσεται, οι λεπτομέρειες των απαιτήσεων μπορούν να δημιουργήσουν προβλήματα επικοινωνίας. Το κόστος διατήρησης ενός γλωσσάριου είναι πολύ μικρό σε σχέση με τα πλεονεκτήματα του.

Ορισμοί:

Προσθήκη Προϊόντος: Ο χρήστης, υπάλληλος καφετέριας μπορεί να προσθέσει καινούργιο προϊόν στην βάση δεδομένων του. Η προσθήκη του προϊόντος γίνεται στο σύστημα όπου καλείται ο υπάλληλος να εισάγει περιγραφή, τιμή, κατηγορία και εικόνα προϊόντος όπως αυτά θα εμφανίζονται στο POS.

Λίστα προϊόντων: Αυτή η λίστα είναι όλα τα προϊόντα που υπάρχουν στην βάση, στην οποία μπορεί να ψάξει ο χρήστης για οποιοδήποτε προϊόν, φιλτράροντας ακόμα και τις κατηγορίες προϊόντος.

Κατηγορίες Προϊόντων: Τα προϊόντα χωρίζονται σε κάποιες βασικές κατηγορίες όπου απαρτίζονται από, Hot Drinks, Cold Drinks, Blended, Alcohol, Other, και Snacks.

Hot Drinks: Κατηγορία προϊόντων που αναφέρεται σε ζεστά ροφήματα με βάση το εσπρέσο ή άλλα ζεστά ροφήματα όπως τσάι ή ζεστή σοκολάτα.

Cold Drinks: Κατηγορία προϊόντων που αναφέρεται σε κρύα ροφήματα με βάση το εσπρέσο ή άλλα κρύα ροφήματα όπως φρέσκος χυμός, παγωμένο τσάι.

Blended: Κατηγορία προϊόντων που αναφέρεται σε κρύα crashed ice ροφήματα με βάση το εσπρέσο.

Alcohol: Κατηγορία προϊόντων που αναφέρεται σε αλκοολούχα ποτά.

Other: Κατηγορία προϊόντων που αναφέρεται σε εμφιαλωμένα ποτά όπως νερό ή αναψυκτικά.

Snacks: Κατηγορία προϊόντων που αναφέρεται σε τρόφιμα ή γλυκά.

Εκτέλεση παραγγελίας: Ο υπάλληλος παίρνει παραγγελία από τον πελάτη και επιλέγει προϊόντα από την οθόνη του POS και εισάγονται στην λίστα παραγγελίας.

Λίστα Παραγγελίας: Με την εκτέλεση της παραγγελίας τα προϊόντα εισάγονται σε μια διπλανή λίστα όπου συνοψίζεται η παραγγελία του πελάτη μαζί με τις περιγραφές και τιμές των προϊόντων και έπειτα το συνολικό ποσό της παραγγελίας στο τέλος.

Διαγραφή προϊόντος: Πολύ συχνά πελάτες αλλάζουν γνώμη στην παραγγελία τους ή απλά ο υπάλληλος κτύπησε λάθος προϊόν στην παραγγελία, έτσι μπορεί να σβήσει κάποιο προϊόν από την παραγγελία.

Πληρωμή παραγγελίας: Για να κλίσει μια παραγγελία ο υπάλληλος πρέπει να πληρωθεί και έτσι εμφανίζει το τελικό ποσό της παραγγελίας, όπου εισάγει ποσό που πήρε από τον πελάτη και υπολογίζει τυχόν ρέστα ή υπολειπόμενο ποσό και αυτομάτως τυπώνεται η απόδειξη του πελάτη.

Τελικό ποσό: Αναφέρεται στο αθροισμένο ποσό από όλα τα προϊόντα μιας παραγγελίας.

Display Sales: Συχνά ο υπάλληλος της καφετέριας η υπεύθυνος θέλει να ελέγξει τα transactions που έχει κάνει κάποιο προϊόν ώστε να ελέγξει τις πωλήσεις του και να αυξήσει τις πωλήσεις σε κάποιο συγκεκριμένο προϊόν.

Transactions: Ο ορισμός αυτός αναφέρεται σε ένα κόστος που συνεπάγεται μια οικονομική ανταλλαγή, δηλαδή τις αποδείξεις που εκτελούνται από τις παραγγελίες των πελατών.

UML (Unified Modeling Language):

UML είναι η πιο γνωστή και χρησιμοποιήσιμη γλώσσα μοντελοποίησης συστημάτων software. Είναι μια γραφική γλώσσα που σχηματοποιεί, διασαφηνίζει, κατασκευάζει και καταγράφει ένα σύστημα. Η UML προσφέρει ένα standard για να περιγράψει ένα μοντέλο του συστήματος, περιλαμβάνοντας εννοιολογικά ζητήματα όπως τις διαδικασίες και λειτουργίες του συστήματος, και μπορεί να ορίσει την έκφραση των γλωσσών προγραμματισμού, βάσεων δεδομένων και επαναχρησιμοποιήσιμα σχήματα. Είναι σημαντικό να σημειωθεί ότι η UML είναι μια "γλώσσα" για να προσδιοριστούν και όχι για να περιγραφούν μέθοδοι ή διεργασίες. Χρησιμοποιείται για τον καθορισμό ενός συστήματος, την τεκμηρίωση και την οικοδόμησή του.

POS(Point-Of-Sale): Σημείο πώλησης (POS) ή το ταμείο είναι ο τόπος όπου ένα κατάστημα λιανικής πώλησης ολοκληρώνεται μια συναλλαγή. Είναι το σημείο στο οποίο ένας πελάτης κάνει μια πληρωμή σε έναν έμπορο σε αντάλλαγμα για αγαθά ή υπηρεσίες. Στο σημείο πώλησης, ο έμπορος θα χρησιμοποιήσει οποιαδήποτε από μια σειρά από πιθανές μεθόδους για τον υπολογισμό του οφειλόμενου ποσού, όπως ένα χειροκίνητο σύστημα, ζυγιστικές μηχανές, σαρωτές ή μια ηλεκτρονική ταμειακή μηχανή. Ο έμπορος συνήθως παρέχει το υλικό και τις επιλογές για χρήση από τον πελάτη να κάνει την πληρωμή, όπως ένα τερματικό EFTPOS. Ο έμπορος θα εκδώσει κανονικά απόδειξη για τη συναλλαγή

Κεφάλαιο 4

Σχεδιασμός

4.1 Αποφάσεις σχεδιασμού 31
4.2 Διαγράμματα ακολουθίας 35
4.3 Διαχείριση των δεδομένων 54

Ο σχεδιασμός είναι η διαδικασία κατά την οποία η ασάφεια της φάσης των προδιαγραφών και των απαιτήσεων ανάλυσης, διαλύεται και πραγματοποιείται το πρώτο του βήμα προς την υλοποίηση του λογισμικού. Μέχρι στιγμής η ανάπτυξη του συστήματος είναι ανεξάρτητη από τις τεχνολογίες με τις οποίες μπορεί να υλοποιηθεί. Η κύρια δραστηριότητα του σχεδιασμού είναι η επεξεργασία του μοντέλου ανάλυσης ούτως ώστε το τελευταίο να μπορεί να υλοποιηθεί κάτω από τους κανόνες της αρχιτεκτονικής και τεχνολογικών πόρων που έχουμε στη διάθεση μας. Οι δραστηριότητες του σχεδιασμού επικεντρώνονται γύρω από τους χάρτες πλοήγησης και τα διαγράμματα ακολουθίας. Κατά τη διάρκεια της διαδικασίας σχεδιασμού οι κλάσεις περιγράφονται κάθε φορά αναλυτικότερα με τις ιδιότητες και τις συναρτήσεις τους καλά ορισμένες. Το τελικό μοντέλο σχεδιασμού που προκύπτει είναι κάτι που μπορεί να μετατραπεί απευθείας σε κώδικα.

4.1 Αποφάσεις σχεδιασμού

Γιατί το σύστημα να είναι windows form εφαρμογή.

Μπορούμε να αναπτύξουν εφαρμογές με τα Windows Forms όταν θέλουμε η εφαρμογή του πελάτη να είναι υπεύθυνη για μεγάλο μέρος του φόρτου επεξεργασίας. Αυτές οι εφαρμογές client περιλαμβάνουν Win32 desktop εφαρμογές που παραδοσιακά είχαν αναπτυχθεί σε προηγούμενες εκδόσεις της Visual Basic και Visual C++. Παραδείγματα περιλαμβάνουν σχέδιο ή γραφικών εφαρμογών, συστήματα εισόδου δεδομένων, το point-of-sale συστήματα, και παιχνίδια.

Αυτές οι εφαρμογές, βασίζονται στην επιφάνεια εργασίας, στην υπολογιστική ισχύ για την επεξεργασία και υψηλής απόδοσης οθόνες περιεχομένου. Μερικές μορφές εφαρμογών των Windows θα μπορούσε να είναι εντελώς αυτόνομες και να εκτελούν κάθε επεξεργασία στον υπολογιστή του χρήστη. Παιχνίδια γράφονται συχνά με αυτόν τον τρόπο. Άλλες εφαρμογές μπορεί να είναι μέρος ενός μεγαλύτερου συστήματος και να χρησιμοποιείται στην επιφάνεια εργασίας του υπολογιστή κυρίως για την είσοδο του χρήστη για επεξεργασία. Για παράδειγμα, ένα point-of-sale σύστημα απαιτεί συχνά έναν ευέλικτο, εκλεπτυσμένο περιβάλλον χρήσης που έχει δημιουργηθεί στην επιφάνεια εργασίας του υπολογιστή, αλλά συνδέεται με άλλα στοιχεία που εκτελούν back-end επεξεργασίες

.

Επειδή χτίζουμε ένα Windows Form εφαρμογή των Windows γύρω από έναν υπολογιστή με Windows, η εφαρμογή έχει πρόσβαση στους πόρους του συστήματος στον υπολογιστή-πελάτη, συμπεριλαμβανομένων των τοπικών αρχείων, το μητρώο των Windows, τον εκτυπωτή, και ούτω καθεξής. Αυτό το επίπεδο πρόσβασης μπορεί να περιορίζεται στην εξάλειψη των κινδύνων για την ασφάλεια ή των δυνητικών προβλημάτων που προκύπτουν από την ανεπιθύμητη πρόσβαση. Επιπλέον, τα Windows Forms μπορούν να χρησιμοποιήσουν τα. NET Framework GDI + κλάσεις γραφικών για να δημιουργήσουμε μια πλούσια σε γραφικά περιβάλλον, η οποία είναι συχνά μια προϋπόθεση για την εξόρυξη δεδομένων και εφαρμογές παιχνιδιών.

Για να μετατραπεί η εφαρμογή σε μια υψηλής χρηστικότητας εφαρμογή, ληφθήκαν υπόψη οι ακόλουθες γραμμές σχεδιασμού:

• Λιτότητα έκφρασης:

Οι χρήστες χρειάζονται να διαβάζουν μόνο τα απολύτως απαραίτητα. Ούτε παραπάνω ούτε παρακάτω. Όπως επίσης λήφθηκε υπόψη ότι ο χρήστης πρέπει να μπορεί να πραγματοποιεί τις δραστηριότητες του με την ελάχιστη δυνατή πλοήγηση (navigability) μεταξύ των σελίδων της εφαρμογής.

Έτσι λοιπόν ο ταμίας χρησιμοποιεί ουσιαστικά μόνο την οθόνη του ταμείου. Χρησιμοποιεί βασικά την οθόνη της δημιουργίας παραγγελιών POS (Terminal point of Sale). Σε περίπτωση που οι χρήστες χρειάζονται να χρησιμοποιήσουν άλλες οθόνες, υπάρχουν στην διάθεση τους κάποια menu πάντα ορατά και εύκολα σε

πρόσβαση. Καμία περίπτωση χρήσης δεν απαιτεί πάνω από τρία επίπεδα παραθύρων για να ολοκληρωθεί.

• Απλοποίηση της εμφάνισης:

Αυτό δεν σημαίνει ότι δεν φροντίσαμε το design της εφαρμογής μας. Έχει να κάνει με την βελτιστοποίηση και επίτευξη ενός καλού design μέσω λίγων γραφικών μέσων. Η χρήση γραφικών κλάσεων που υπάρχουν στα windows forms έχουν βοηθήσει πολύ.

• Συνοχή και standards :

Με τον όρο συνοχή αναφερόμαστε στο ότι για παράδειγμα δεν χρησιμοποιούμε δύο διαφορετικούς γραφικούς τρόπους για να αναφερθούμε στο ίδιο περιεχόμενο, ούτε χρησιμοποιούμε διαφορετικά στυλ μέσα στον ίδιο χώρο. Ακριβώς για τον λόγο αυτό, προσαρμόστηκε ένα στυλ σε ολόκληρη την εφαρμογή.

Εκτός από αυτό χρησιμοποιήθηκε στάνταρ σχεδιασμού ευρέως αποδεκτό. Όσο πιο πολύ μοιάζει ο σχεδιασμός και η λειτουργία του συστήματος στις υπόλοιπες εφαρμογές που κυκλοφορούν, τόσο πιο οικεία και εύκολη θα είναι η χρήση του από έναν καινούργιο χρήστη. Δουλεύτηκαν στοιχεία όπως scroll στις οθόνες τις εφαρμογής, κουμπιά διαφόρων ειδών, λίστες, menu και γρήγορες αναζητήσεις, με σκοπό η εφαρμογή μας να γίνει χρηστική.

Όσων αφορά τις γραμματοσειρές και τις ιδιότητές τους ακολουθήθηκε η χρήση γραμματοσειρών μεγάλου μεγέθους που παρέχουν άνεση στην ανάγνωση. Χωρίς αμφιβολία αποφεύχθηκε η υπερβολική χρήση μεγάλων γραμματοσειρών για να εκμεταλλευτούμε σωστά τους ελεύθερους χώρους που είναι τόσο χρήσιμοι στον σχεδιασμό μιας εφαρμογής. Για το λόγο αυτό χρησιμοποιήθηκε μια ποικιλία μεγεθών και χρωμάτων, μαζί με τις κλασσικές ιδιότητες των γραμματοσειρών (bold, πλάγια, κ.λπ.), ώστε να κάνουμε τα μηνύματα μας κατανοητά με μια μόνο ματιά.

• Διαφάνεια της κατάστασης στην οποία βρίσκεται το σύστημα :

Το σύστημα πάντα ενημερώνει τον χρήστη γύρω από το τι ακριβώς συμβαίνει. Ένα τέχνασμα που χρησιμοποιήσαμε αρκετά για να ενισχύσουμε τον έλεγχο του χρήστη πάνω στο σύστημα, είναι το μπλοκάρισμα της οθόνης ενόσω το σύστημα χρειάζεται χρόνο για να ολοκληρώσει μια αίτηση του χρήστη.

• Χρήση μιας κοινής γλώσσας μεταξύ του συστήματος και του χρήστη:

Το σύστημα μιλάει στην ίδια γλώσσα με τον χρήστη, αποφεύγοντας ακατανόητους τεχνικισμούς και κρυπτικά μηνύματα.

• Ο χρήστης μπορεί να λύνει παραγόμενα λάθη :

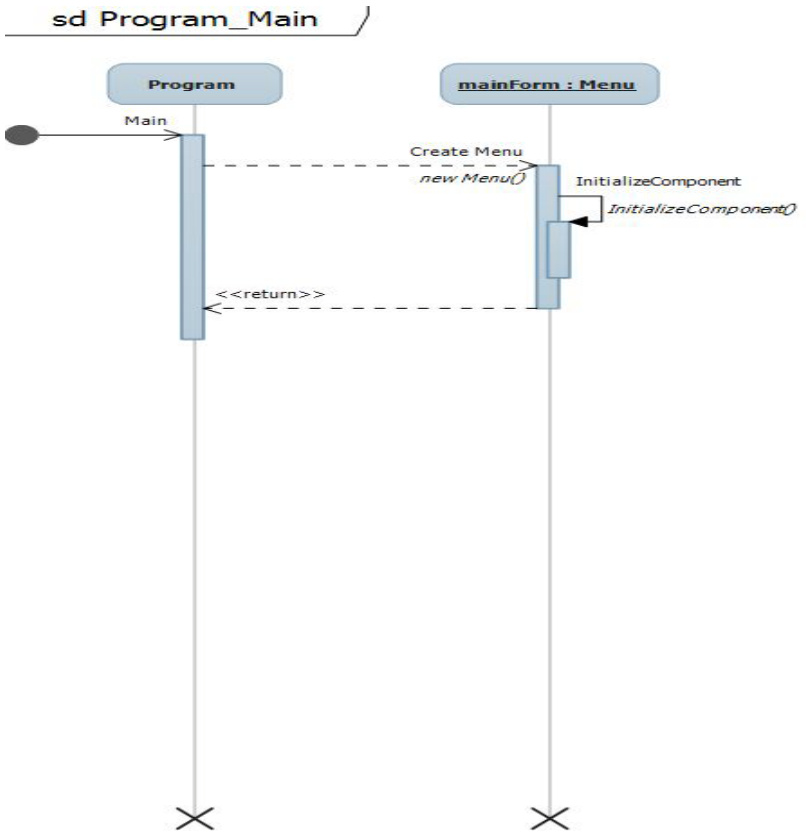
Για παράδειγμα, όταν ο χρήστης εισάγει λάθος προϊόν στην λίστα παραγγελίας, και δεν το θέλει ο πελάτης ο χρήστης έχει την δυνατότητα να διαγράψει το προϊόν αυτό από την παραγγελία και να μην συμπεριλήφθη στο τελικό ποσό.

• Αποφυγή λαθών:

Δεν πραγματοποιείται μια παραγγελία αν δεν πληρωθεί το τελικό ποσό. Το σύστημα δεν αφήνει τον χρήστη να κλίσει μια παραγγελία αν δεν πληρωθεί το όλο το οφειλόμενο ποσό και μετά να εκτυπωθεί η απόδειξη. Σχετικό μήνυμα εμφανίζεται για τυχόν υπολειπόμενο ποσό σε μια παραγγελία.

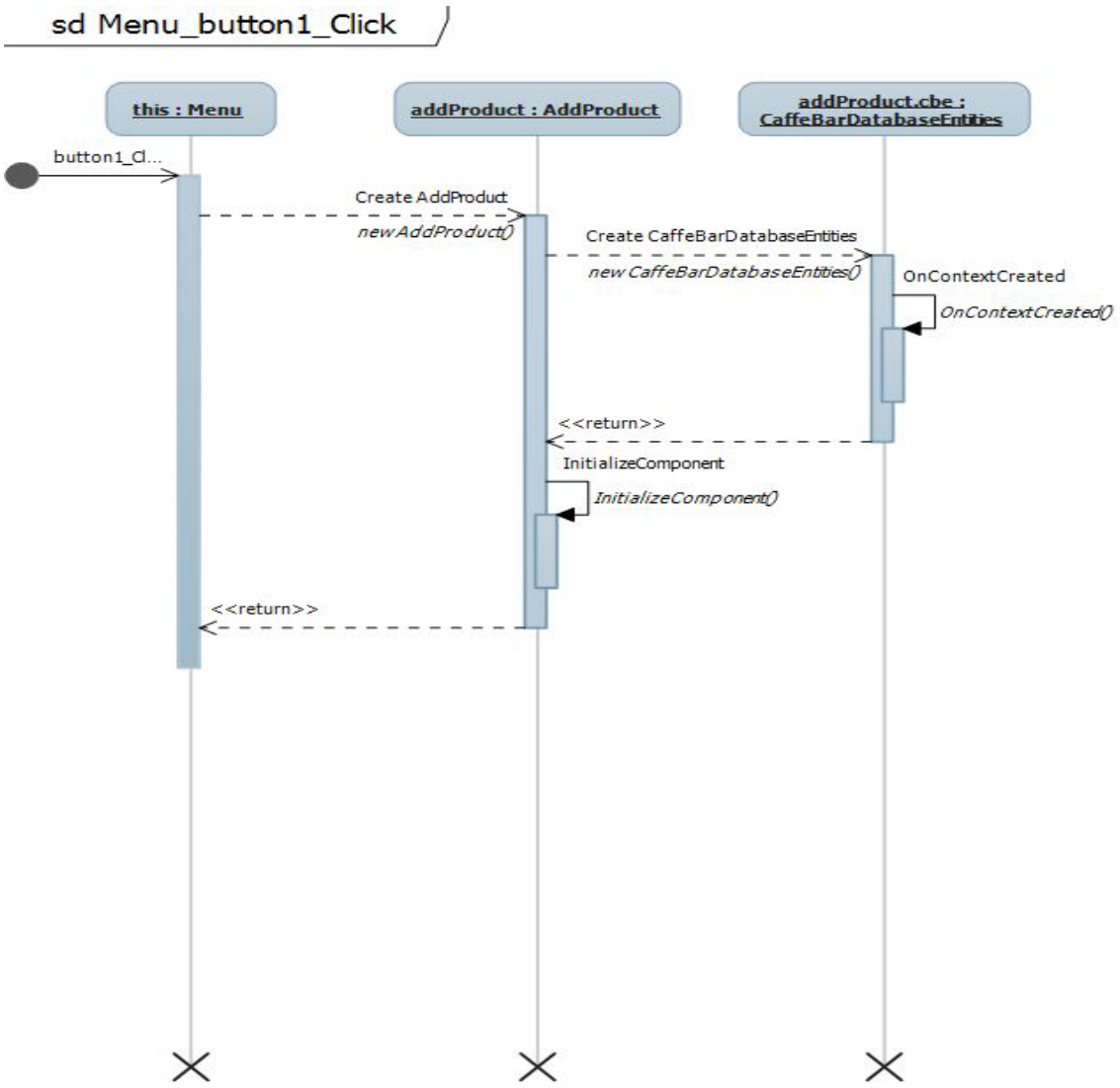
4.2 Διαγράμματα ακολουθίας (Sequence diagrams)

Ο μηχανισμός της UML για να εκφράσει την δυναμική σχέση μεταξύ των κλάσεων μιας εφαρμογής, είναι τα διαγράμματα αλληλεπίδρασης. Τα διαγράμματα αλληλεπίδρασης αποτελούν μια κατηγορία διαγραμμάτων στην οποία ανήκουν τα διαγράμματα συνεργασίας, ακολουθίας και δραστηριότητας.



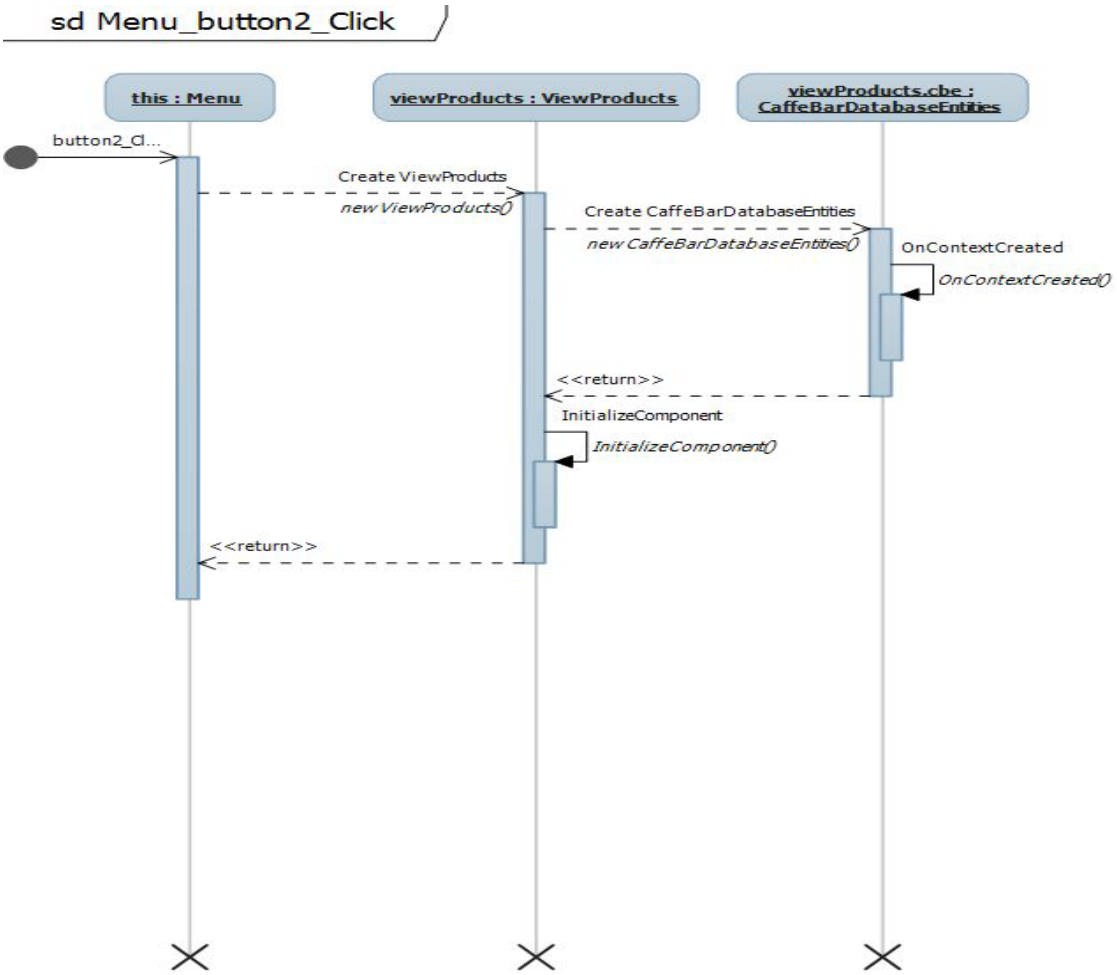
Εικόνα 3: Κύριο πρόγραμμα κυρίως μενού.

Με την εκκίνηση του προγράμματος εμφανίζεται η φόρμα του κυρίως μενού.



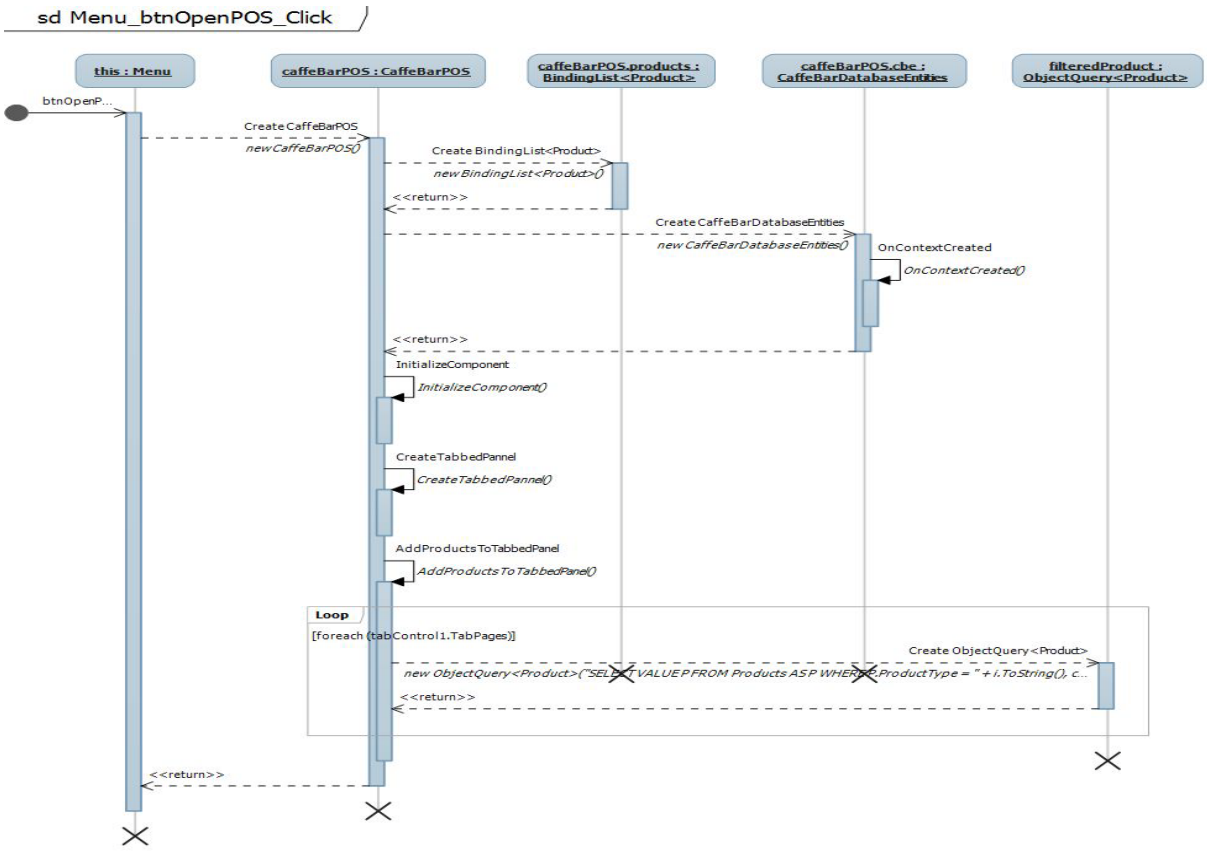
Εικόνα 4: Κυρίως μενού κουμπί προσθήκη προϊόντος.

Με το πάτημα του κουμπιού Add Product εμφανίζεται η φόρμα εισαγωγής νέου προϊόντος η οποία συμπληρώνεται και έπειτα ενημερώνετε η βάση δεδομένων.



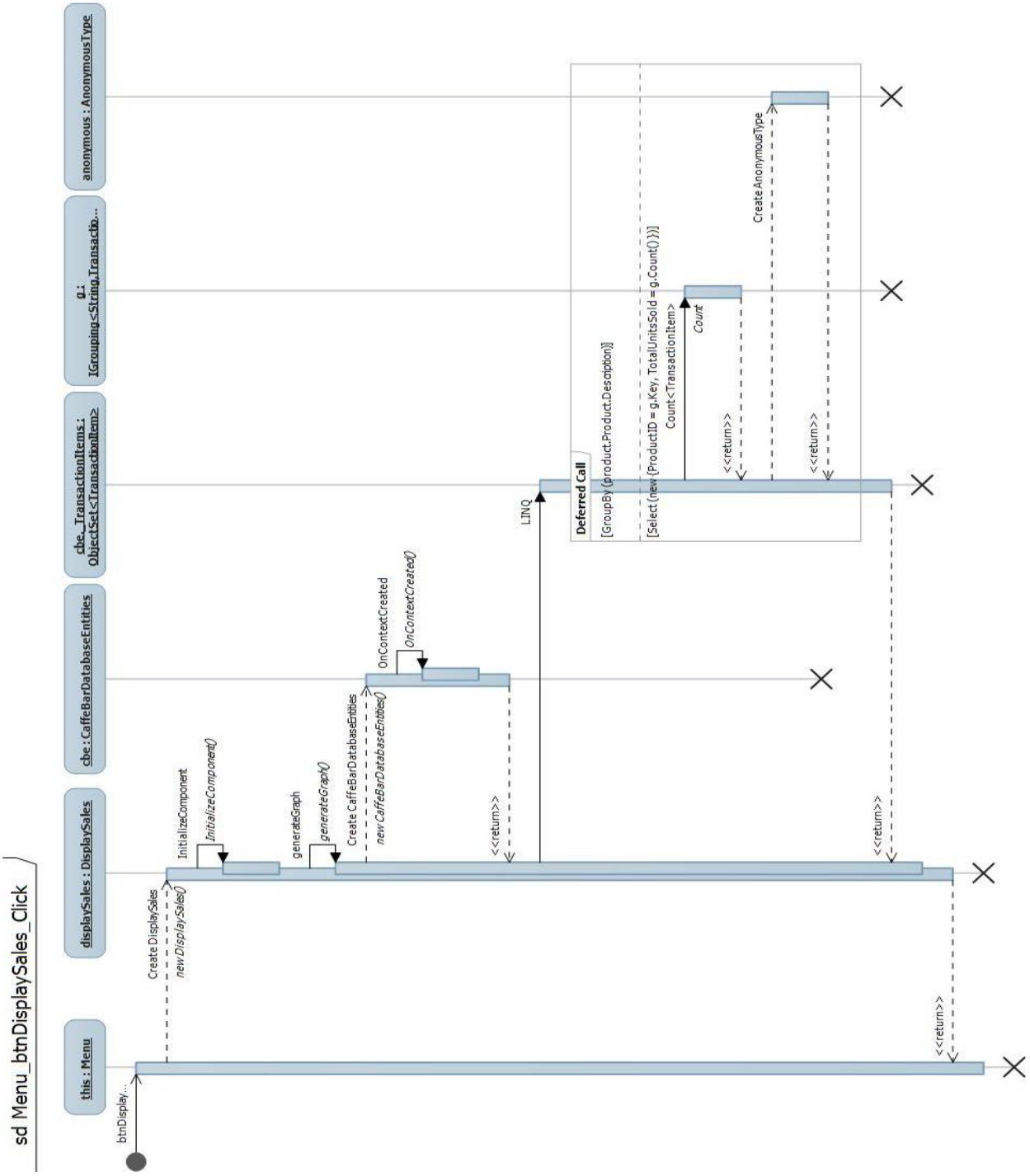
Εικόνα 5: Κυρίως μενού κουμπί λίστα προϊόντων.

Με το πάτημα του κουμπιού View Products εμφανίζεται λίστα με τα προϊόντα που υπάρχουν στην βάση δεδομένων, με το πάτημα δημιουργείται η λίστα παίρνοντας τα δεδομένα από την βάση δεδομένων.



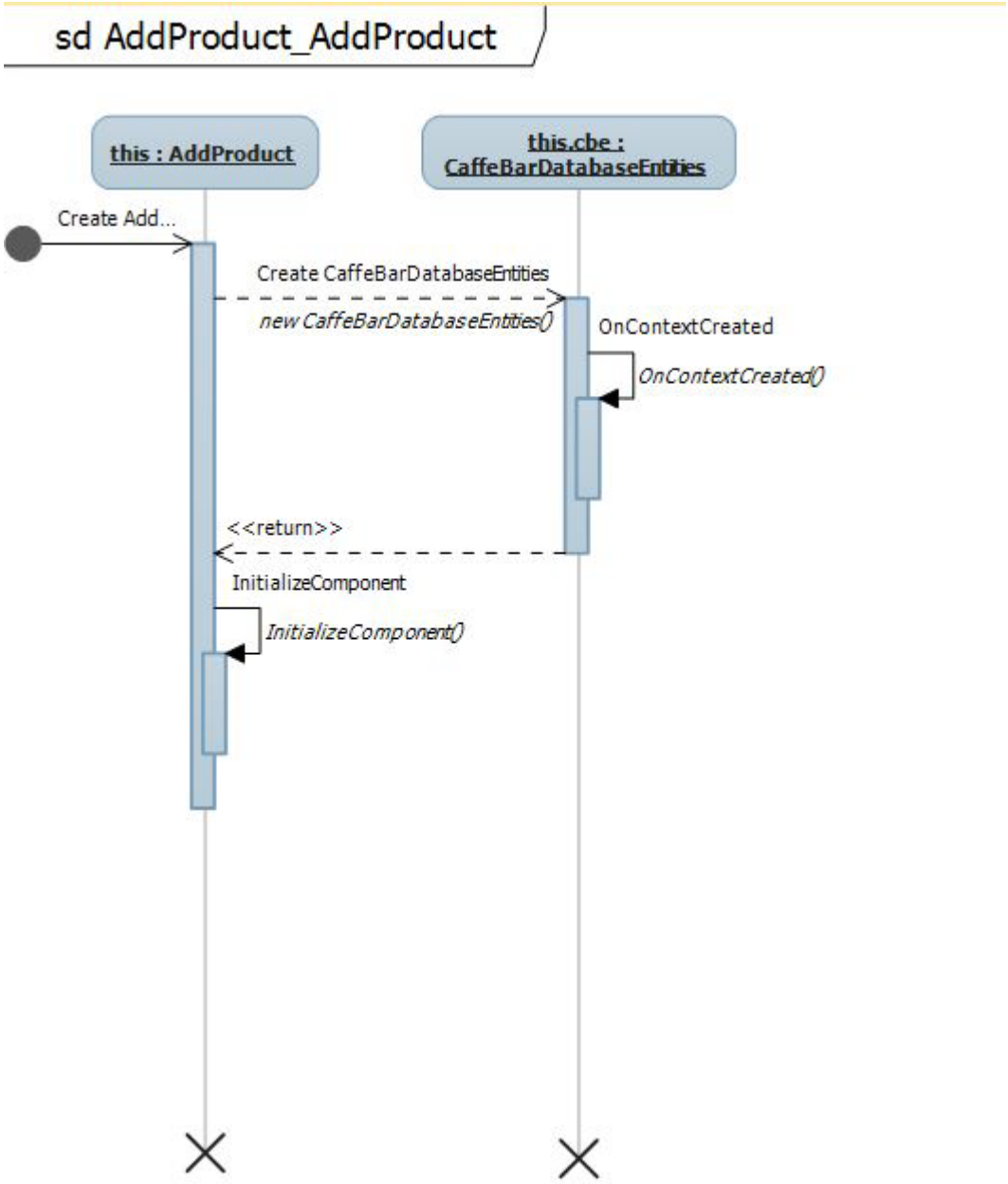
Εικόνα 6: Κυρίως μενού κουμπί άνοιγμα POS.

Με το πάτημα του κουμπιού Open POS εμφανίζεται η οθόνη του POS όπου δημιουργείται μια οθόνη με κουπιά με τα προϊόντα ανά κατηγορία παίρνοντας όλα τα δεδομένα από την βάση δεδομένων.



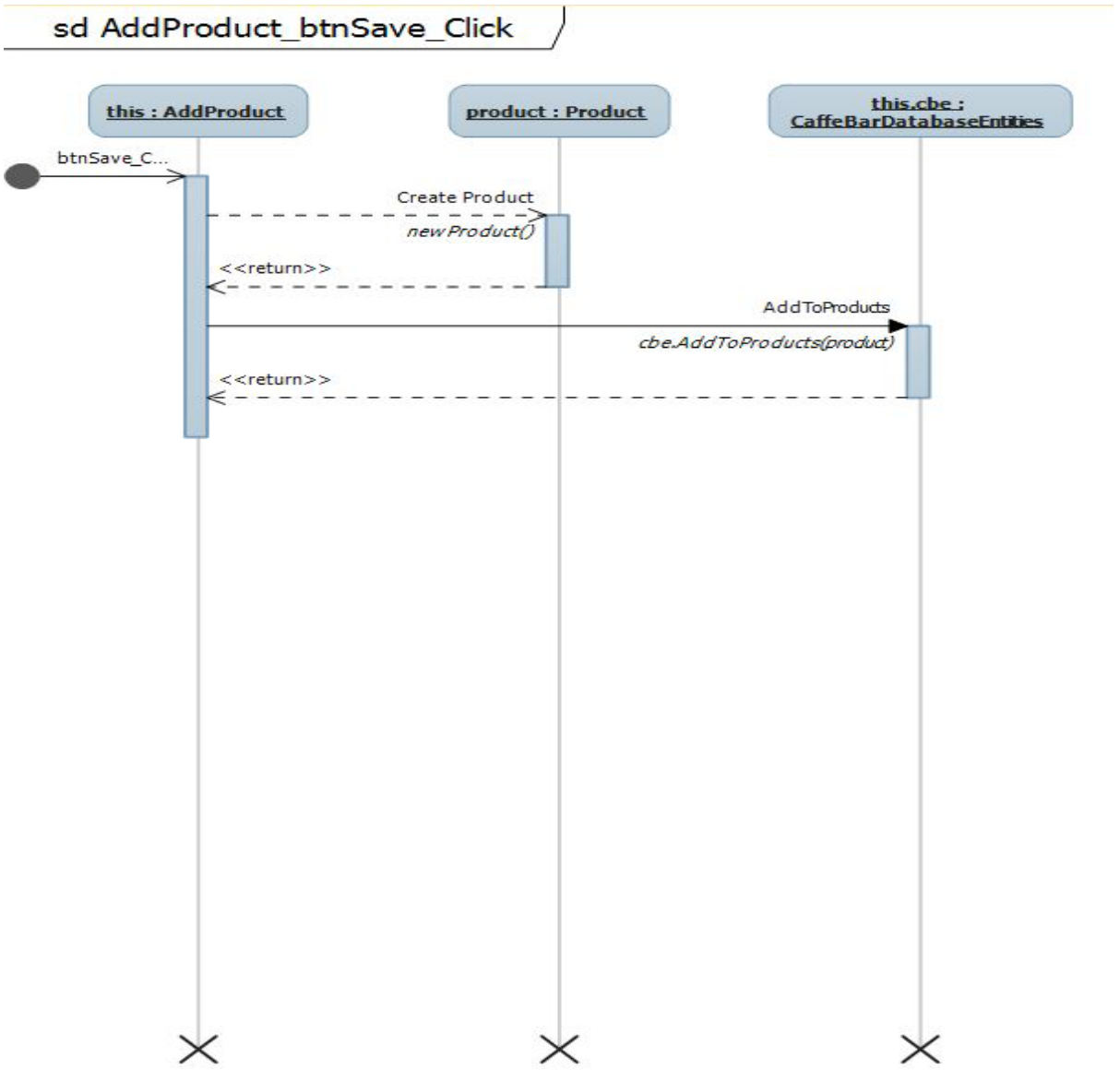
Εικόνα 7: Κυρίως μενού κουμπί Display Sales.

Με το πάτημα του Display Sales δημιουργείται μια γραφική παράσταση με τις γενικές πωλήσεις των προϊόντων ανακτώντας τα απαραίτητα δεδομένα από την βάση δεδομένων.



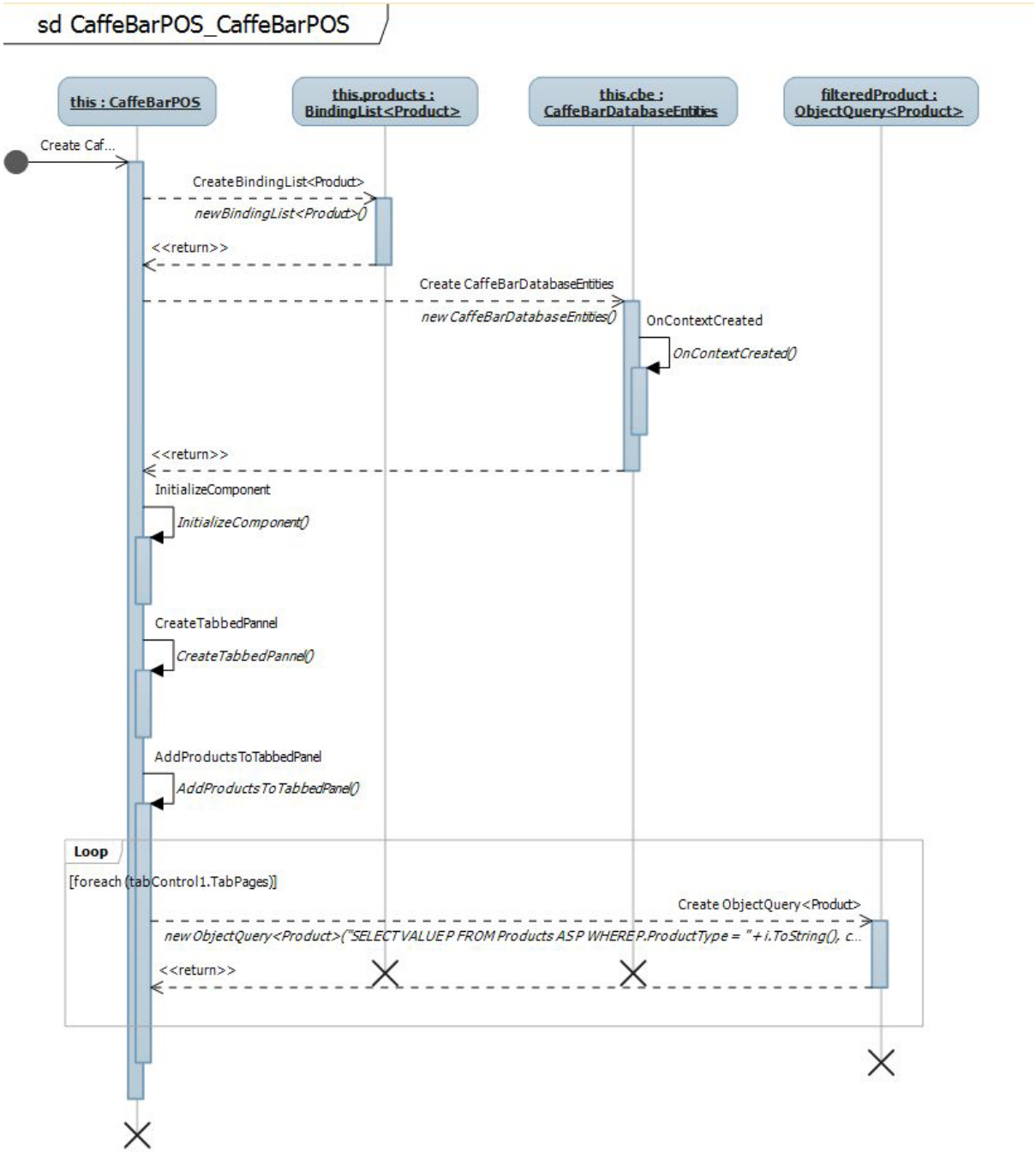
Εικόνα 8: Φόρμα Προσθήκη προϊόντος.

Με το πάτημα του κουμπιού Add Product εμφανίζεται η φόρμα εισαγωγής νέου προϊόντος



Εικόνα 9: Αποθήκευση προϊόντος στην βάση.

Όταν συμπληρωθεί η φόρμα εισαγωγείς νέου προϊόντος με το πάτημα του κουμπιού save αποθηκεύεται το νέο προϊόν στην βάση δεδομένων.

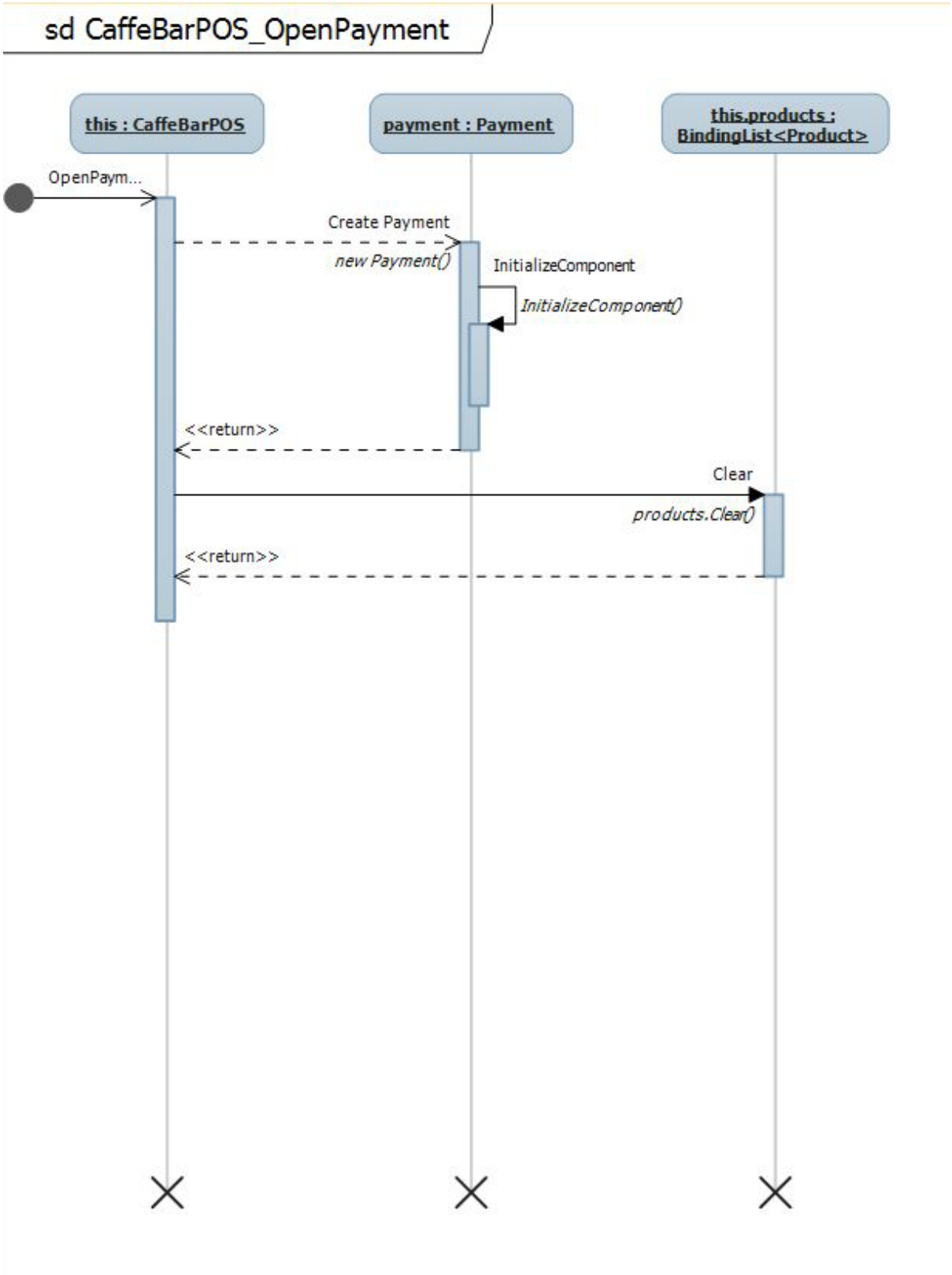


Εικόνα 10: Οθόνη POS.

Δημιουργία της οθόνης POS κατηγοριοποιώντας τα προϊόντα με εικονίδια σε tabs παίρνοντας τα απαραίτητα δεδομένα από την βάση δεδομένων.

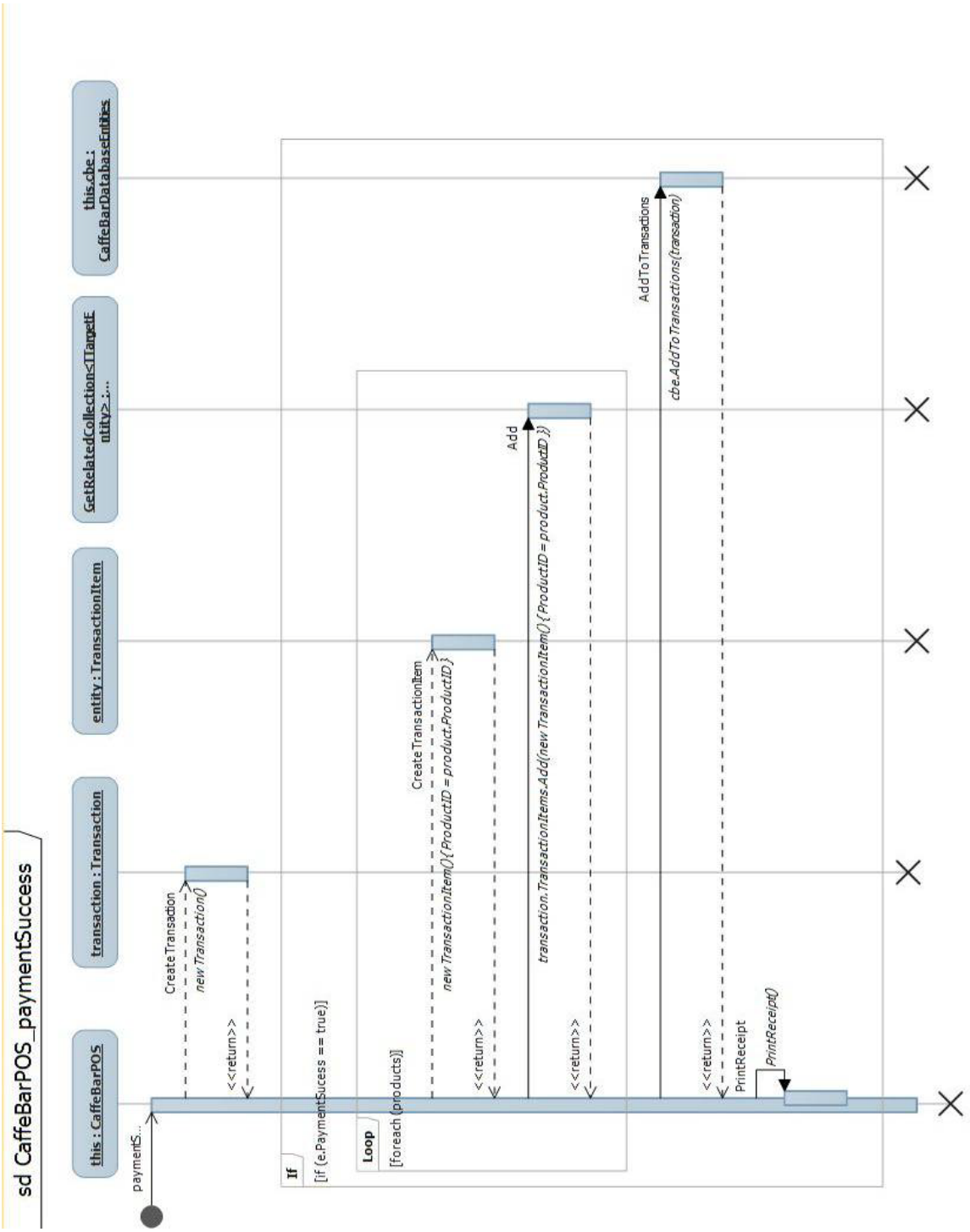
Εικόνα 11: POS update product list.

Με την επιλογή διάφορων προϊόντων από την οθόνη POS ενημερώνεται η λίστα παραγγελίας.



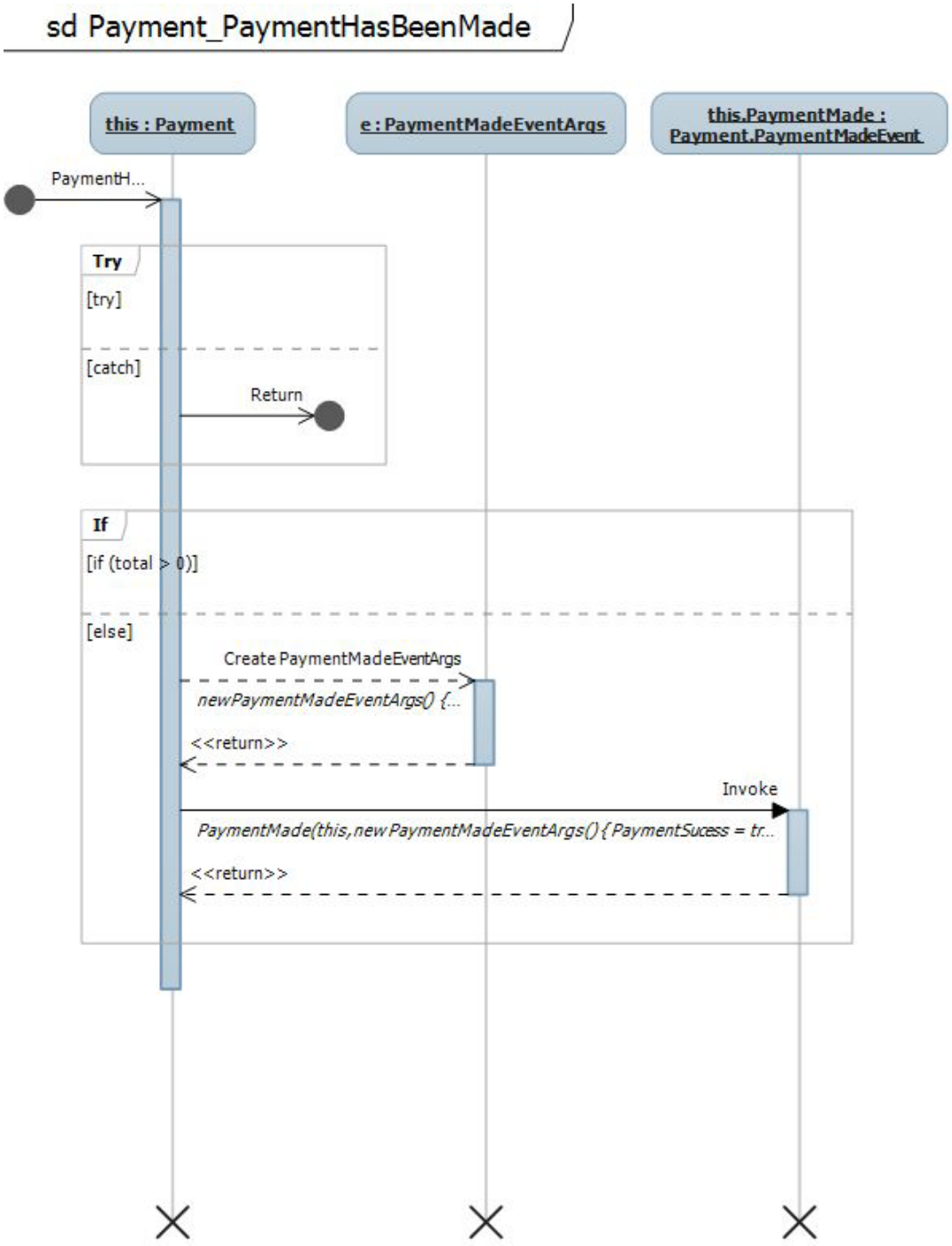
Εικόνα 12: Δημιουργία παραγγελίας.

Ανοίγει το παράθυρο της πληρωμής της παραγγελίας παίρνοντας το τελικό ποσό της παραγγελίας από την λίστα παραγγελίας.



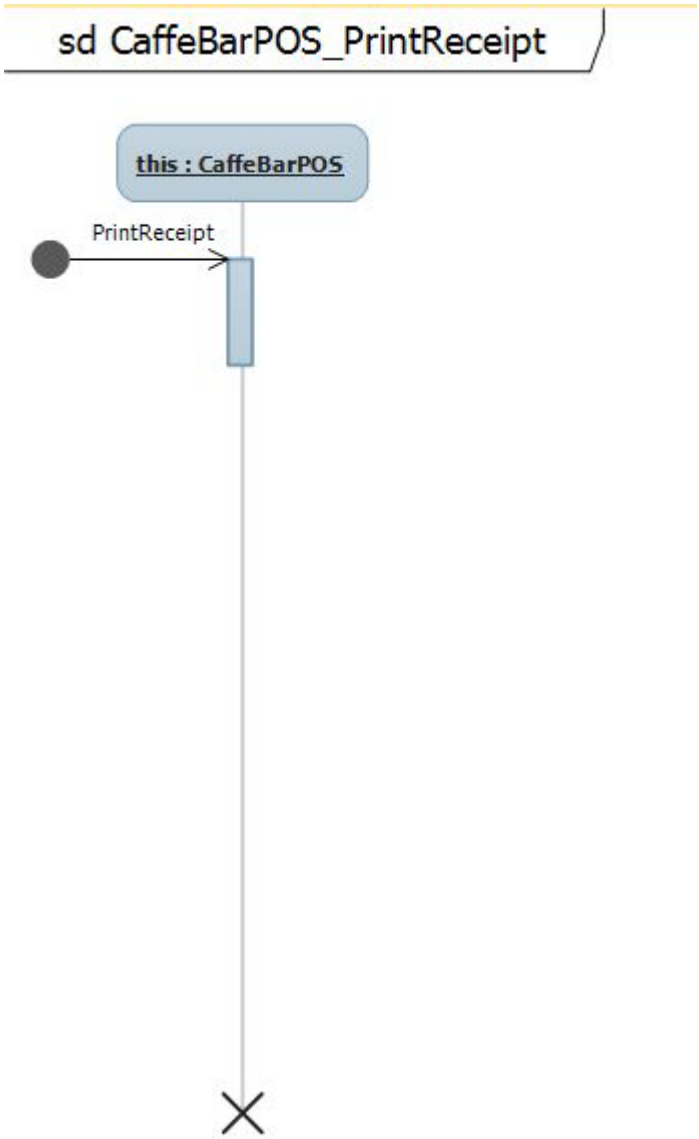
Εικόνα 13: Εκτέλεση παραγγελίας.

Όταν γίνει η πληρωμή και είναι σωστή αποθηκεύεται το transaction στην βάση δεδομένων.



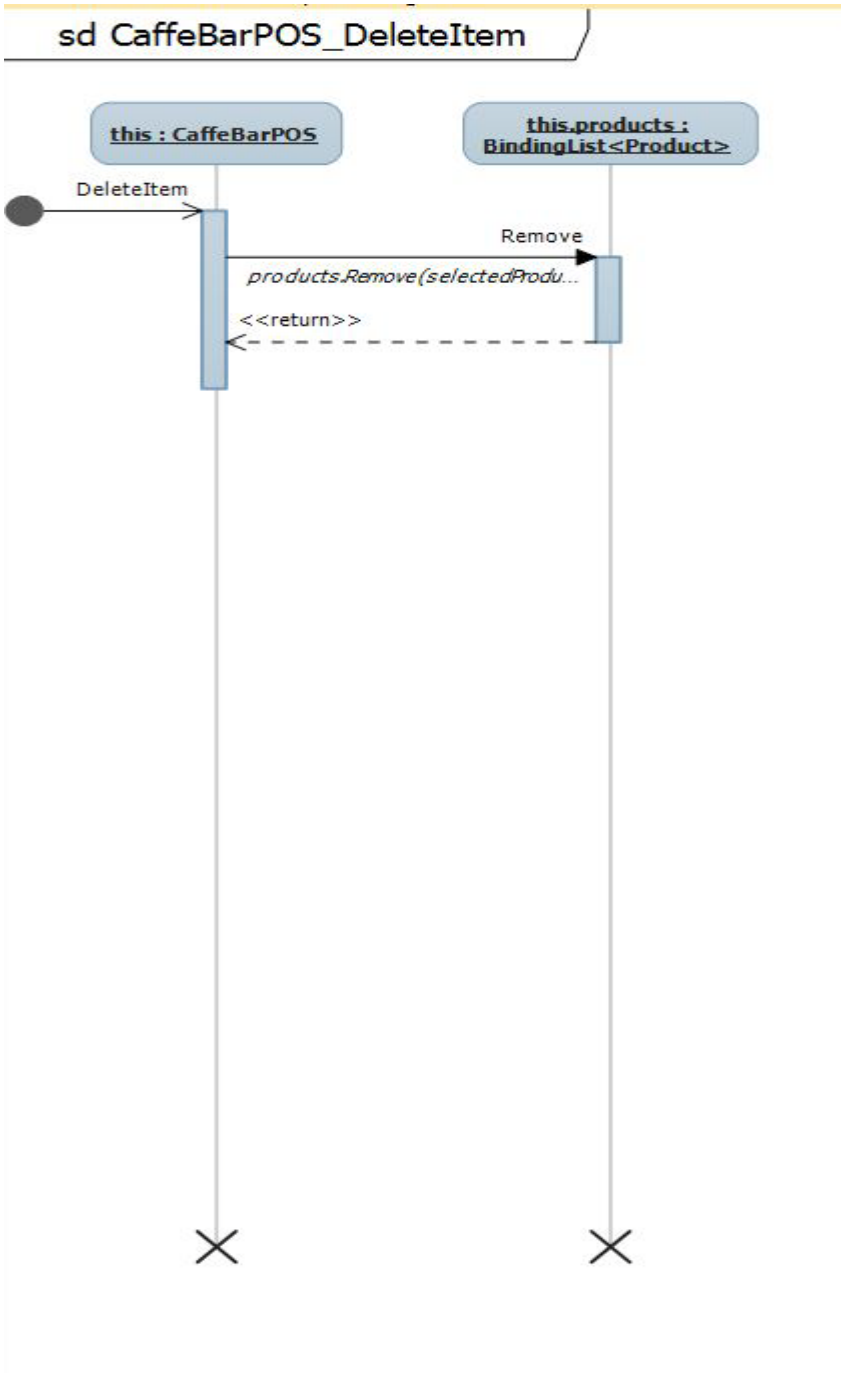
Εικόνα 14: Εκτέλεση πληρωμής.

Γίνεται η πληρωμή της παραγγελίας και είναι αποδεκτή στέλνει το μήνυμα ότι είναι επιτυχής η πληρωμή ώστε ακολουθήσει η εκτύπωση της απόδειξης.



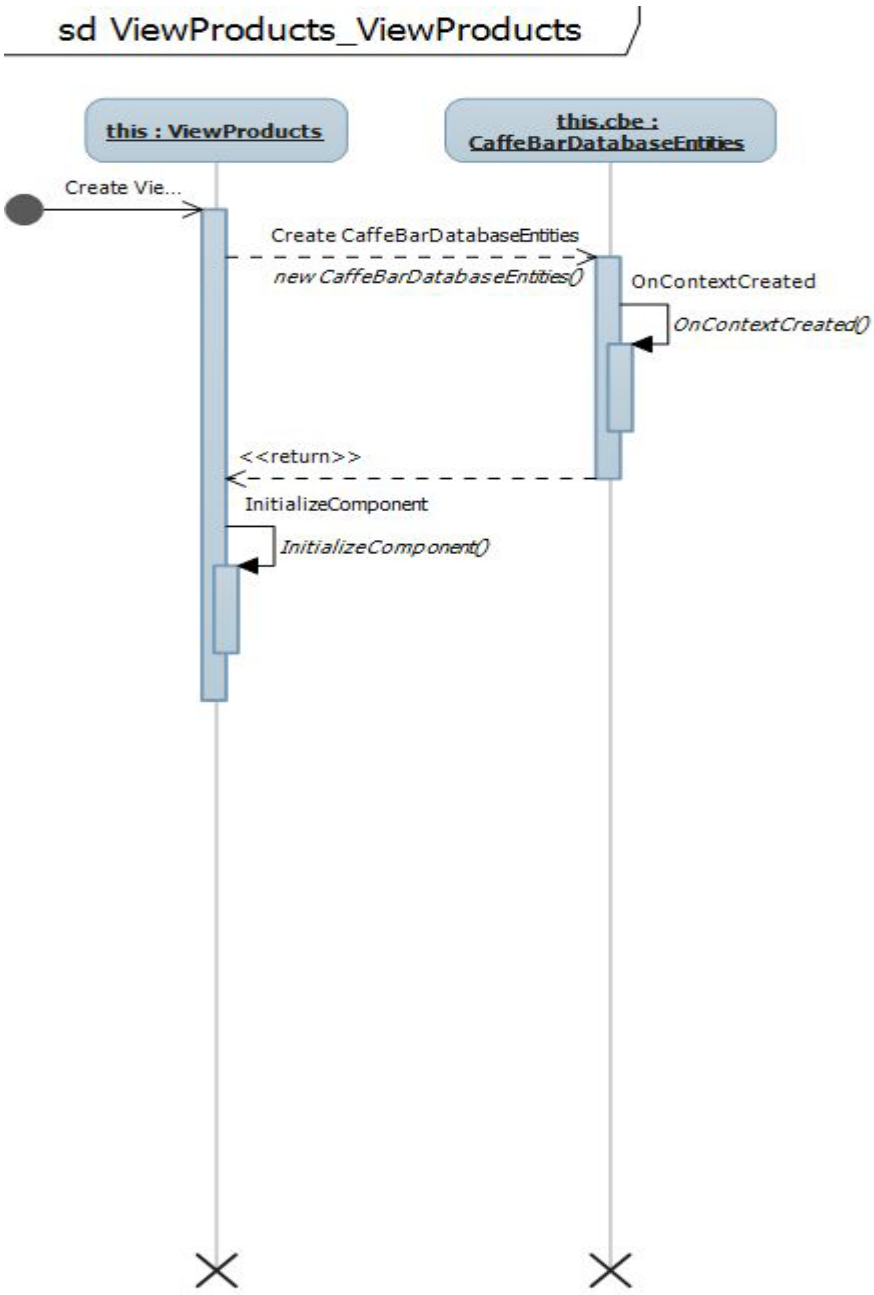
Εικόνα 15: Εκτύπωση απόδειξης.

Με το τέλος της πληρωμής της παραγγελίας εμφανίζεται η επιλογή εκτύπωση της απόδειξης.



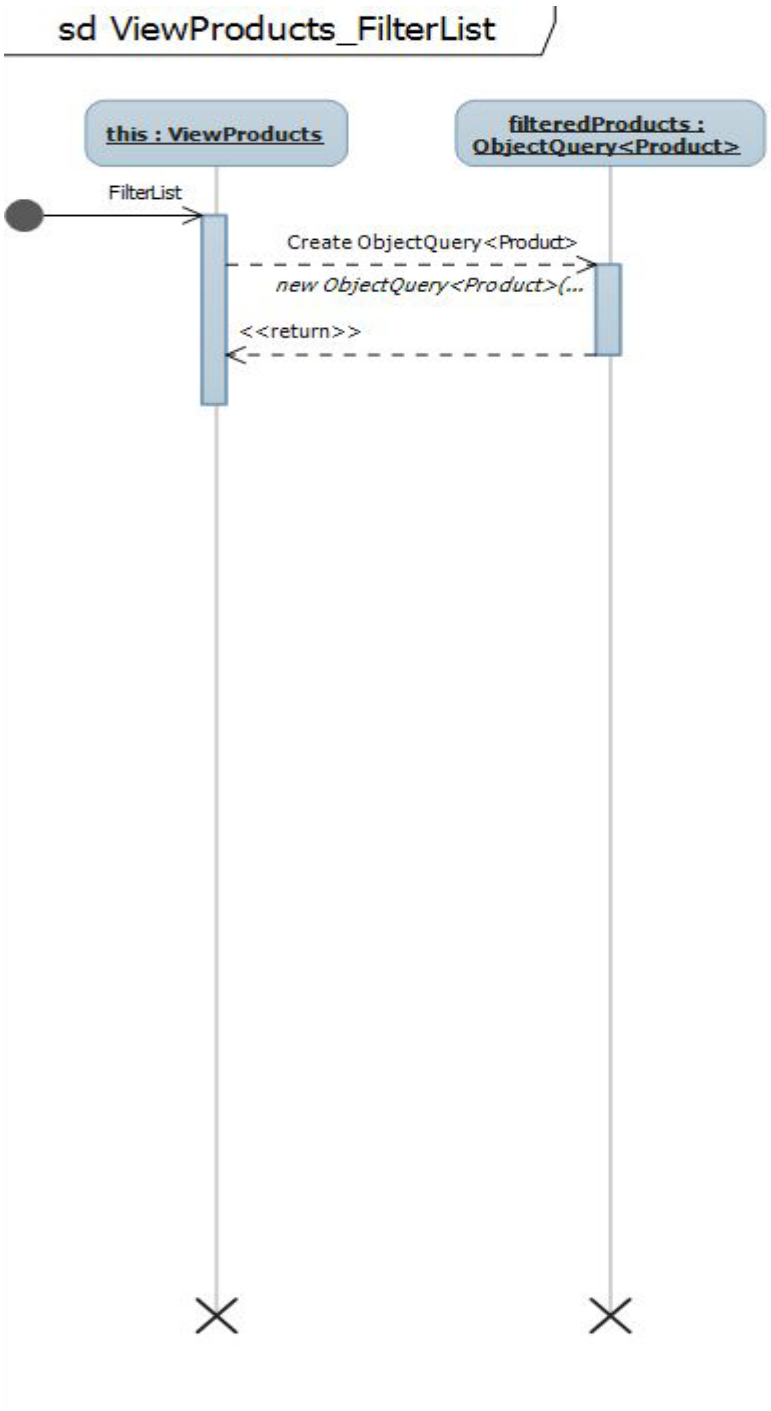
Εικόνα 16: Διαγραφή προϊόντος από λίστα παραγγελίας.

Έχουμε την δυνατότητα να διαγράψουμε ένα προϊόν από την λίστα παραγγελίας.



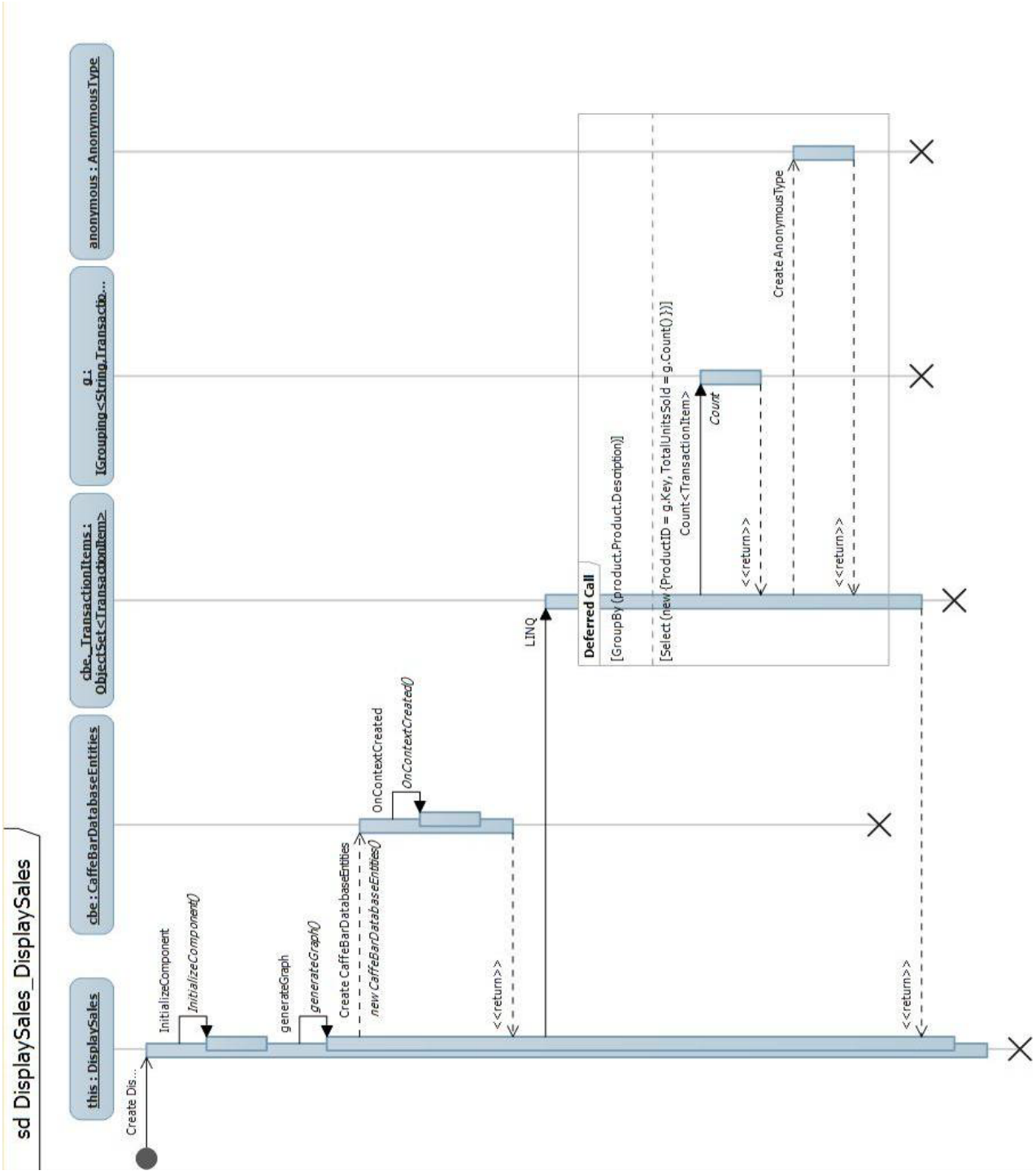
Εικόνα 17: Λίστα Προϊόντων.

Δημιουργία λίστα προϊόντων ανακτώντας τα δεδομένα από την βάση δεδομένων.



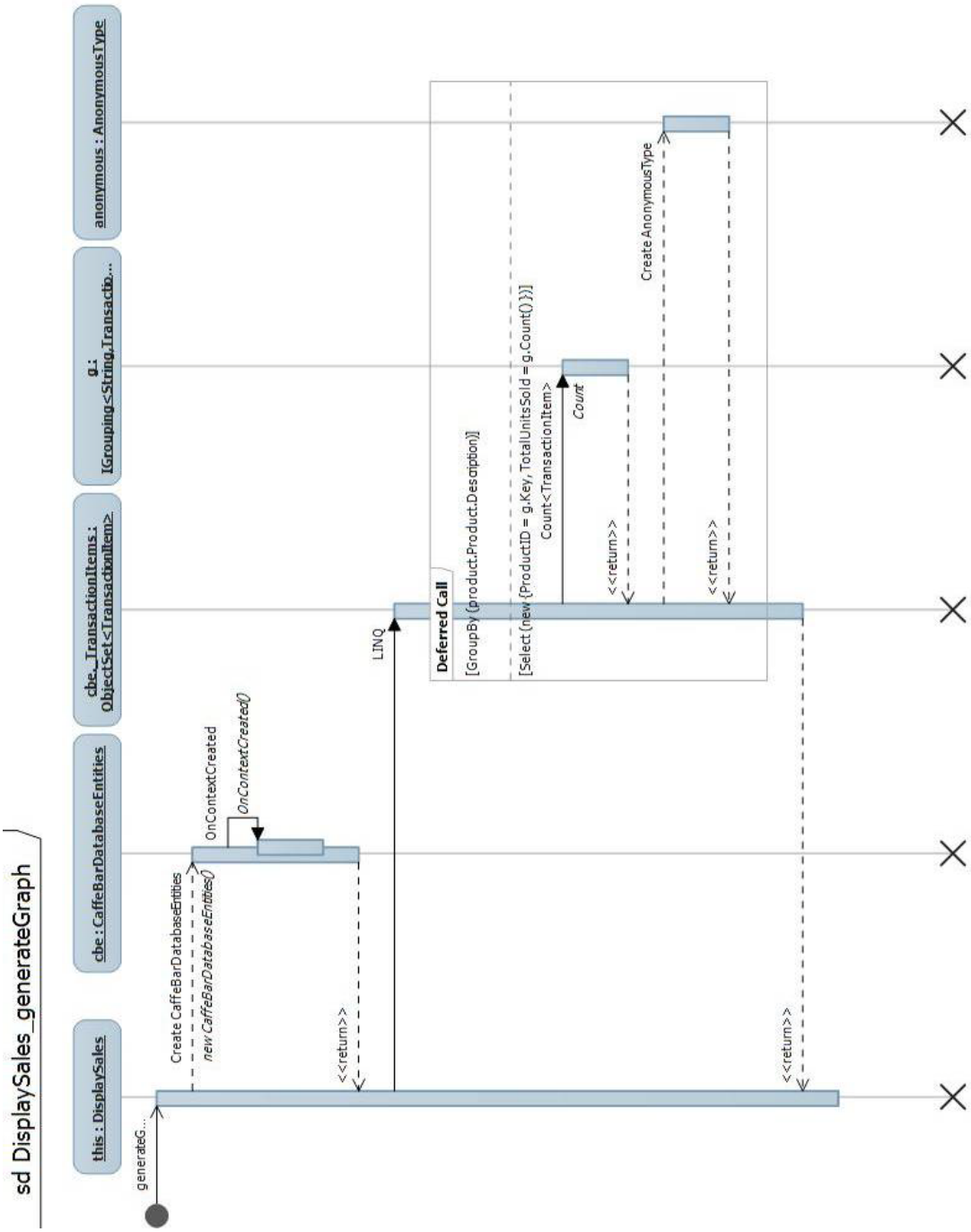
Εικόνα 18: Φιλτράρισμα λίστας προϊόντων.

Η λίστα προϊόντων μπορεί να φιλτραριστεί ανάλογα με τις κατηγορίες των προϊόντων.



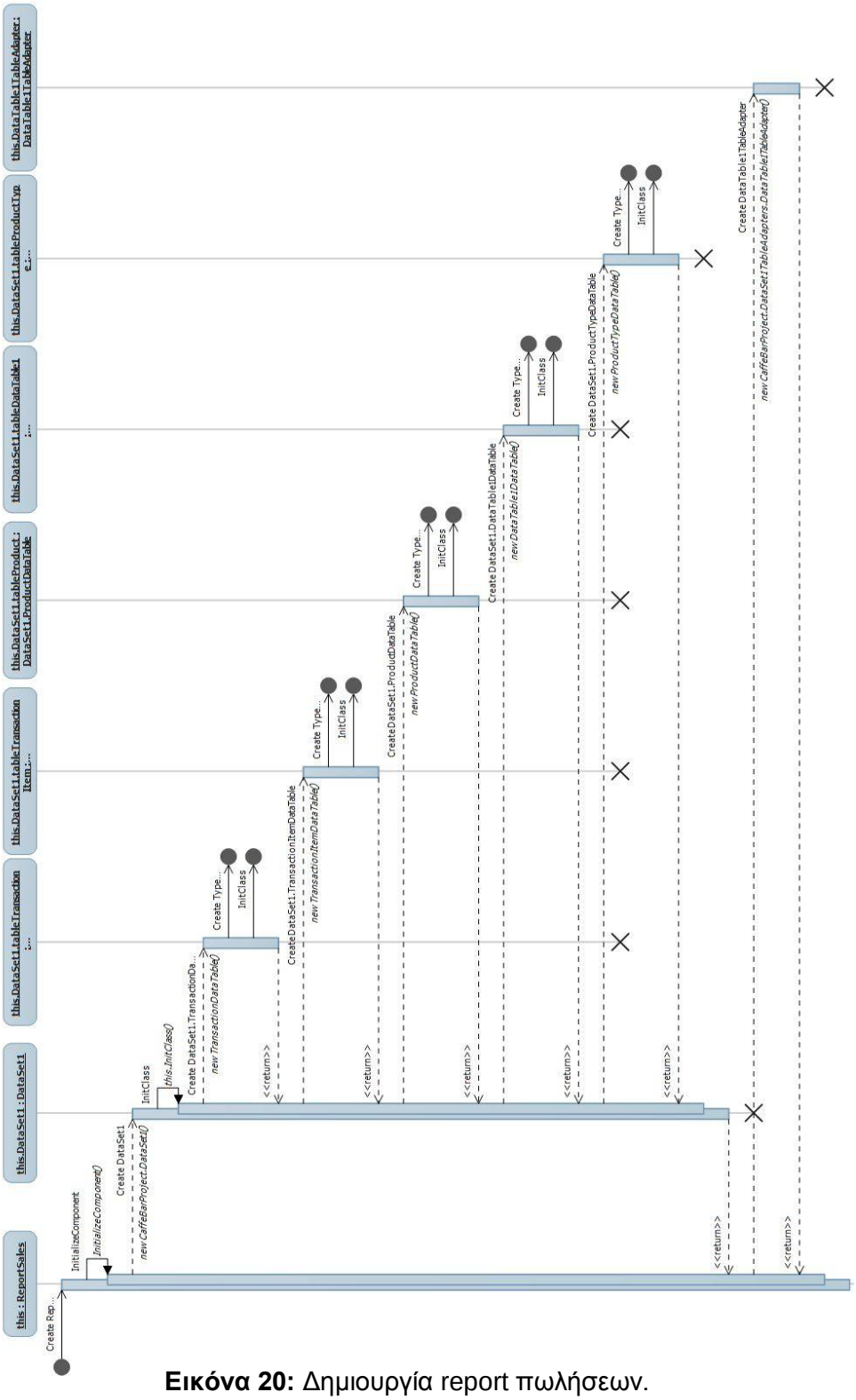
Εικόνα 19: Εμφάνιση πωλήσεων.

Για να δημιουργηθεί η γραφική παράσταση των πωλήσεων ανακτώνται τα δεδομένα από την βάση δεδομένων για transactions.



Εικόνα 19: Δημιουργία γραφικής παράστασης των πωλήσεων.

Δημιουργία γραφικής παράστασης των πωλήσεων με τα δεδομένα των transactions που ανακτήθηκαν από την βάση δεδομένων.



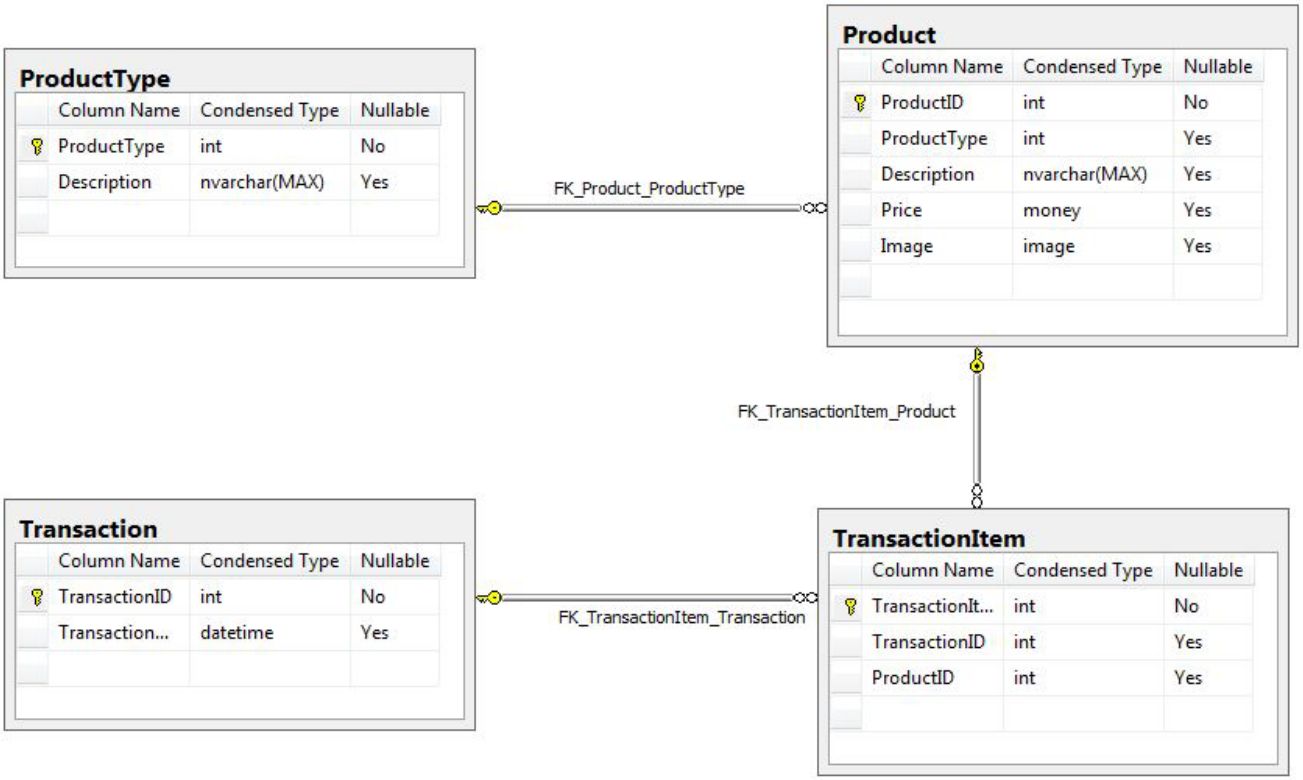
Εικόνα 20: Δημιουργία report πωλήσεων.

Με το πάτημα του κουμπιού Report Sales ο χρήστης επιλέγει ημερομηνία και δημιουργείται report με τις πωλήσεις της συγκεκριμένης ημερομηνίας ανά προϊόν και ανά ώρα παίρνοντας τα απαραίτητα δεδομένα από την βάση δεδομένων.

4.3 Διαχείριση των δεδομένων

Μία βάση δεδομένων είναι μία “αποθήκη” που μας επιτρέπει να καταγράφουμε, με τρόπο οργανωμένο, μεγάλες ποσότητες πληροφορίας, τις οποίες αργότερα μπορούμε να εντοπίσουμε και να χρησιμοποιήσουμε εύκολα και γρήγορα. Υπάρχουν πολλοί τρόποι οργάνωσης της πληροφορίας και αναπαράστασης των σχέσεων μεταξύ των δεδομένων μιας βάσης δεδομένων. Χρησιμοποιήθηκε το σχεσιακό μοντέλο ως λογικό μοντέλο για να περιγραφούν πιστά οι ιδιότητες και σχέσεις των δεδομένων. Το σχεσιακό μοντέλο δεδομένων αναπαριστά όλα τα δεδομένα της βάσης δεδομένων ως απλούς πίνακες με δύο διαστάσεις που ονομάζονται "σχέσεις". Κάθε πίνακας έχει μία ή παραπάνω στήλες και γραμμές. Οι στήλες περιγράφουν ένα κομμάτι πληροφορίας πάνω σε κάθε στοιχείο που θέλουμε να καταγράψουμε στον πίνακα και κάθε γραμμή του πίνακα αποτελεί μια καταγραφή. Τα δεδομένα των πινάκων μπορούν να συλλεχθούν και συνδυαστούν εύκολα.

Για να δημιουργηθεί μια βάση δεδομένων πρέπει να προηγηθούν τρεις επάλληλες φάσεις: ο εννοιολογικός, λογικός και ο φυσικός σχεδιασμός. Ο εννοιολογικός σχεδιασμός αναφέρεται στην ανάπτυξη ενός μοντέλου δεδομένων σε ανώτερο επίπεδο αφαίρεσης με αποκλειστικό γνώμονα τους κανόνες οργάνωσης και λειτουργίας του οργανισμού. Ο λογικός σχεδιασμός αναφέρεται στην ανάπτυξη ενός μοντέλου δεδομένων το οποίο αποτελεί προσαρμογή του εννοιολογικού μοντέλου στο μοντέλο δεδομένων που υποστηρίζεται από το DBMS(database management system) που προβλέπεται να χρησιμοποιηθεί για την υλοποίηση της βάσης δεδομένων. Ο φυσικός σχεδιασμός αναφέρεται σε θέματα υλοποίησης της βάσης δεδομένων στο διαθέσιμο DBMS. Ακολούθως παρουσιάζετε τον λογικό σχεδιασμό της βάσης, περιγράφοντας πώς τα στοιχεία της βάσης δεδομένων ομαδοποιούνται.



Εικόνα 21: Λογικός σχεδιασμός βάσης.

Κεφάλαιο 5

Υλοποίηση

5.1 Διαγράμματα κλάσεων 56
5.2 Τεχνολογίες που χρησιμοποιήθηκαν 59
5.3 Γλώσσες προγραμματισμού 60
5.4 Εφαρμογές 61

5.1 Διαγράμματα κλάσεων

Τα διαγράμματα κλάσεων είναι στατικής δομής και δείχνουν τις κλάσεις του συστήματος και τους δεσμούς μεταξύ τους (δεσμοί κληρονομικότητας, γενίκευσης και σύνδεσης). Τα διαγράμματα κλάσεων είναι ένα βασικό εργαλείο τηςUML και χρησιμοποιούνται κυρίως για να δείξουν όχι μόνο το τι μπορεί να κάνει το σύστημα (Φάση ανάλυσης), αλλά και το πώς μπορεί να κατασκευαστεί (Φάση σχεδιασμού).

Το εννοιολογικό μοντέλο, που περιέχεται στο διάγραμμα κλάσεων, εξηγεί τις σημαντικότερες ιδέες του προβλήματος, ταυτοποιώντας τις κλάσεις των αντικειμένων, τις ιδιότητες, τους δεσμούς και την πολλαπλότητα τους (πόσα αντικείμενα συμμετέχουν σε κάθε δεσμό) και τους περιορισμούς ακεραιότητας. Έχοντας υπόψη μας τις περιπτώσεις χρήσεως και τις λειτουργικές και μη λειτουργικές απαιτήσεις του συστήματος, ορίζουμε τα αντικείμενα στο ακόλουθο διάγραμμα κλάσεων.

Εικόνα 2: Διάγραμμα κλάσεων

Add Product

Κάθε φορά που εισάγεται καινούργιο προϊόν ενημερώνονται τα CaffeBarDatabaseEntities οι οντότητες της βάσεις δεδομένων. Δημιουργείται καινούργιο ProductID για το προϊόν και αποθηκεύεται στον πίνακα προϊόντων με περιγραφή, τιμή, κατηγορία και εικόνα.

Product Type

Οι κατηγορίες προϊόντων περιορίζονται στις βασικές κατηγορίες που είναι: Hot Drinks, Cold Drinks, Blended, Alcohol, Other και Snacks.

Menu

Δημιουργία γραφικού περιβάλλον μενού με επιλογές που μπορεί να κάνει ο χρήστης.

CaffeBarPOS

Δημιουργία παραθύρου POS στο οποίο μπορεί να εκτελέσει μια παραγγελία ο χρήστης.

Product

Πίνακας που αποθηκεύονται τα πρϊόντα.

TransactionItems

Πίνακας που αποθηκεύονται τα προϊόντα μιας απόδειξης.

Transaction

Πίνακας που αποθηκεύονται οι αποδείξεις των παραγγελιών. Ενημερώνεται κάθε φορά που εκτελείται μια παραγγελία.

View Products

Δημιουργία λίστας προϊόντων που υπάρχουν στην βάση δεδομένων.

Payment

Εκτέλεση πληρωμή παραγγελίας, υπολογίζει ολικό ποσό και εμφανίζει τυχόν ρέστα η υπόλοιπο πληρωμής.

Display Sales

Δημιουργία γραφικής απεικόνισης των αποδείξεων των προϊόντων το οποίο ενημερώνεται από τα transactions.

Report Sales

Δημιουργείται report με τις πωλήσεις των προϊόντων συγκεκριμένης ημερομηνίας, την οποία επιλέγει ο χρήστης, αν άμ ώρα με τα ποσοστά των transaction και τις συνολικές τιμές.

5.1 Τεχνολογίες που χρησιμοποιήθηκαν

Windows Forms

Τα Windows Forms (WinForms) είναι το όνομα που δίνεται στη γραφική διεπαφή προγραμματισμού εφαρμογών (API) που περιλαμβάνεται ως τμήμα της Microsoft .NET Framework, παρέχοντας πρόσβαση σε εγγενή Microsoft στοιχεία του περιβάλλοντος εργασίας των Windows με υπάρχοντα API των Windows σε διαχειριζόμενο κώδικα. Αν και θεωρείται ως αντικατάσταση για την προγενέστερη και πιο πολύπλοκη C + + based Microsoft Foundation Class Library, δεν προσφέρει ένα παράδειγμα σύγκρισης με Model-View-Controller. Ορισμένες after market βιβλιοθήκες έχουν δημιουργηθεί για να παρέχουν αυτή τη δυνατότητα. Η πιο διαδεδομένη από αυτές είναι το User Interface Process Block Εφαρμογή, η οποία

απελευθερώνεται από τα σχέδια της Microsoft patterns & practices group ως δωρεάν download που περιλαμβάνει τον πηγαίο κώδικα για γρήγορη εκκίνηση με παραδείγματα.

5.2 Γλώσσες προγραμματισμού

C#

C # (προφέρεται see sharp) είναι μια multi-paradigm γλώσσα προγραμματισμού που περιλαμβάνει ισχυρή δακτυλογράφηση, επιτακτική ανάγκη, δηλωτική, λειτουργική, διαδικαστικά, γενική, αντικειμενοστραφή (class-based) και component-oriented προγραμματισμό. Αναπτύχθηκε από τη Microsoft στο πλαίσιο. NET και αργότερα έχει εγκριθεί ως πρότυπο από το Ecma (ECMA-334) και το ISO (ISO / IEC 23270:2006). Η C # είναι μία από τις γλώσσες προγραμματισμού που έχουν σχεδιαστεί για την Υποδομή κοινής γλώσσας. Η C # προορίζεται να είναι ένα απλό, μοντέρνο, γενικής χρήσης, object-oriented γλώσσα προγραμματισμού. Η ομάδα ανάπτυξης του ήταν υπό την ηγεσία του Anders Hejlsberg. Η πιο πρόσφατη έκδοση είναι η C # 5.0, που κυκλοφόρησε στις 15 Αυγούστου, 2012.

Ακολουθεί το “Hello world” ως παράδειγμα κώδικα σε C#.

```
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine("Hello world!");
    }
}
```

SQL:

Είναι το ακρώνυμο Structured Query Language, που σημαίνει γλώσσα δομημένων ερωτημάτων. Είναι μια γλώσσα που ορίζει την πρόσβαση σε σχεσιακές βάσεις δεδομένων και επιτρέπει τον ορισμό διαφόρων τύπων λειτουργιών πάνω σε αυτές.

Ένα από τα χαρακτηριστικά της είναι η διαχείριση άλγεβρας και σχεσιακού υπολογισμού, επιτρέποντας την διαχείριση αιτήσεων με σκοπό την ανάκτηση πληροφορίας από την βάση δεδομένων, αλλά και την πραγματοποίηση αλλαγών σε αυτή. Είναι μια γλώσσα τέταρτης γενιάς (4GL). Χρησιμοποιήθηκε η MySQL για την διαχείριση όλης της βάσης δεδομένων. Δηλαδή για να δημιουργήσουμε, εισάγουμε και να ανακτήσουμε όλα τα δεδομένα της βάσης.

5.3 Εφαρμογές

MySQL:

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων, multithreading και multiuser. Σε αντίθεση με project όπως ο Apache, του οποίου το software είναι ανεπτυγμένο από έναν δημόσιο φορέα και το copyright του κώδικα είναι στην διάθεση του οποιοδήποτε ιδιώτη, η MySQL είναι ιδιοκτησία και έχει ως χορηγό μια ιδιωτική εταιρία, στην οποία ανήκουν τα δικαιώματα του copyright του μεγαλύτερου τμήματος του κώδικα.

Η MySQL χρησιμοποιείται πολύ σε εφαρμογές web και σε πλατφόρμες (Linux/Windows- Apache-MySQL-PHP/Perl/Python). Η MySQL είναι μια βάση δεδομένων πολύ γρήγορη στην ανάγνωση αλλά μπορεί να δημιουργήσει προβλήματα ακεραιότητας σε περιβάλλοντα όπου πραγματοποιούνται ταυτόχρονες τροποποιήσεις της βάσης. Σε εφαρμογές web όπου υπάρχουν λιγοστός συγχρονισμός στην τροποποίηση των δεδομένων και ταυτόχρονα υπάρχουν υψηλοί ρυθμοί ανάγνωσης δεδομένων, η MySQL είναι ιδανική για να χρησιμοποιηθεί.

Microsoft visual studio:

Το Microsoft Visual Studio είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) από τη Microsoft. Χρησιμοποιείται για την ανάπτυξη κονσόλας και γραφικές εφαρμογές διεπαφή χρήστη, μαζί με τα Windows Forms και WPF εφαρμογές, ιστοσελίδες, εφαρμογές web και υπηρεσιών web, τόσο εγγενή κώδικα μαζί με διαχειριζόμενο κώδικα για όλες τις πλατφόρμες που υποστηρίζονται από τα Microsoft Windows, Windows Mobile, Windows CE, . NET Framework,. NET Compact Framework και το Microsoft Silverlight.

Η Visual Studio περιλαμβάνει ένα πρόγραμμα επεξεργασίας κώδικα υποστηρίζει IntelliSense καθώς και refactoring κώδικα. Το ολοκληρωμένο πρόγραμμα εντοπισμού σφαλμάτων λειτουργεί τόσο ως ένα source-level debugger και ως ένα machine-level debugger. Άλλα ενσωματωμένα εργαλεία περιλαμβάνουν έναν σχεδιαστή για φόρμες για τη δημιουργία GUI εφαρμογών, web designer, designer κλάσεων, και database schema designer. Δέχεται plug-ins που ενισχύουν την λειτουργικότητα σχεδόν σε κάθε επίπεδο-όπως η προσθήκη υποστήριξης για source-control systems (όπως Subversion και Visual SourceSafe) και την προσθήκη νέων toolsets όπως editors and visual designers για domain-specific γλώσσες ή toolsets για άλλες πτυχές της ανάπτυξη λογισμικού κύκλου ζωής (όπως Team Foundation Server client: Team Explorer).

Η Visual Studio υποστηρίζει διαφορετικές γλώσσες προγραμματισμού μέσω των γλωσσικών υπηρεσιών, οι οποίες επιτρέπουν στον editor κώδικα και debugger την υποστήριξη (σε διάφορους βαθμούς) σχεδόν οποιασδήποτε γλώσσας προγραμματισμού, εφόσον μια γλώσσα ειδικής υπηρεσίας υπάρχει. Ενσωματωμένες γλώσσες περιλαμβάνουν C / C + + (μέσω του Visual C + +), VB.NET (μέσω της Visual Basic. NET), C # (μέσω του Visual C #), και F # (μέσω του Visual Studio 2010). Υποστήριξη για άλλες γλώσσες, όπως η M, Python, Ruby και, μεταξύ άλλων, είναι διαθέσιμες μέσω των υπηρεσιών γλώσσας που μπορούν να εγκατασταθούν ξεχωριστά. Υποστηρίζει, επίσης, XML / XSLT, HTML / XHTML, JavaScript και CSS. Ατομική γλώσσα-συγκεκριμένης εκδόσεις του Visual Studio, επίσης, υπάρχουν οι οποίες παρέχουν πιο περιορισμένες γλωσσικές υπηρεσίες για το χρήστη: Microsoft Visual Basic, Visual J #, Visual C # και Visual C + +.

Η Microsoft παρέχει "Express" εκδόσεις του Visual Studio 2010 του Visual Basic, Visual C #, Visual C + + και Visual Web Developer χωρίς κόστος. Οι Visual Studio 2012, 2010, 2008 και 2005 Professional Editions, μαζί με τις εκδόσεις που αφορούν συγκεκριμένες γλώσσες (Visual Basic, C + +, C #, J #) του Visual Studio Express 2010 είναι διαθέσιμα δωρεάν στους φοιτητές, με λήψεις μέσω του προγράμματος DreamSpark της Microsoft.

Κεφάλαιο 6

Εγκατάσταση και Εγχειρίδιο εφαρμογής

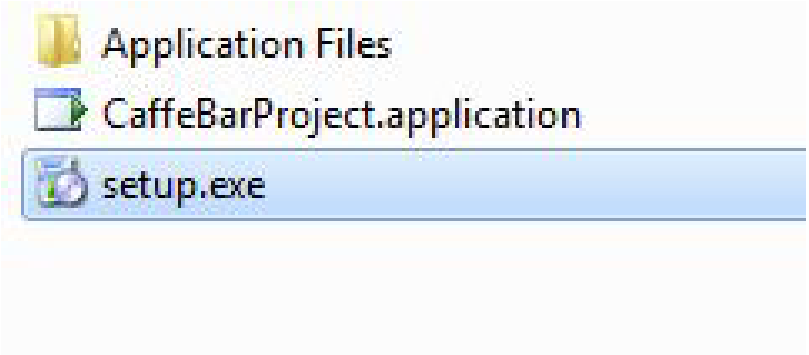
6.1 Εγκατάσταση εφαρμογής 63
6.2 Εγχειρίδιο εφαρμογής 66

6.1 Εγκατάσταση εφαρμογής

Η εφαρμογή είναι click once deployment, με απλά λόγια, μια εφαρμογή ClickOnce είναι οποιοδήποτε Windows Presentation Foundation, Windows Forms, ή εφαρμογή κονσόλας που δημοσιεύθηκε χρησιμοποιώντας την τεχνολογία ClickOnce. Μπορούμε να δημοσιεύσουμε μια εφαρμογή ClickOnce με τρεις διαφορετικούς τρόπους: από μια ιστοσελίδα, από ένα κοινόχρηστο αρχείο δικτύου, είτε από μέσα μαζικής ενημέρωσης, όπως ένα CD-ROM. Μια εφαρμογή ClickOnce μπορεί να εγκατασταθεί στον υπολογιστή του τελικού χρήστη και να τρέξει σε τοπικό επίπεδο, ακόμη και όταν ο υπολογιστής δεν είναι συνδεμένος, ή μπορεί να τρέξει σε απευθείας σύνδεση-μόνο λειτουργία, χωρίς μόνιμη εγκατάσταση οτιδήποτε στον υπολογιστή του τελικού χρήστη.

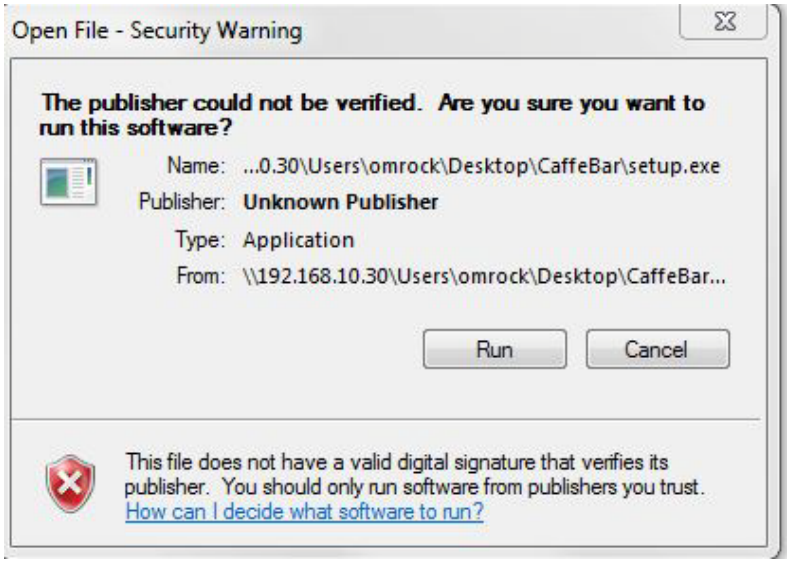
Εφαρμογές ClickOnce μπορεί να αυτο-ενημερώνονται. Μπορούν να ελέγξουν για νεότερες εκδόσεις, καθώς γίνονται διαθέσιμες και να αντικαθιστά αυτόματα τις ενημερωμένα αρχεία. Ο προγραμματιστής μπορεί να καθορίσει τη συμπεριφορά της ενημερωμένης έκδοσης. Ο διαχειριστής του δικτύου μπορεί να ελέγχει, επίσης, στρατηγικές ενημέρωσης, για παράδειγμα, σημειώνοντας ενημέρωση ως υποχρεωτικό. Ενημερώσεις μπορούν επίσης να επανέλθουν σε μια προηγούμενη έκδοση από τον τελικό χρήστη ή από έναν διαχειριστή.

Οι εφαρμογές ClickOnce είναι εγγενώς απομονωμένες, την εγκατάσταση ή την εκτέλεση μιας εφαρμογής ClickOnce δεν μπορούν να σπάσουν υπάρχουσες εφαρμογές. Εφαρμογές ClickOnce είναι εντελώς αυτόνομες. Κάθε αίτηση ClickOnce έχει εγκατασταθεί και τρέχει από μια ασφαλή ανά χρήστη, cache ανά αίτηση. Από προεπιλογή, οι εφαρμογές ClickOnce λειτουργούν στις ζώνες ασφαλείας του Internet ή Intranet.



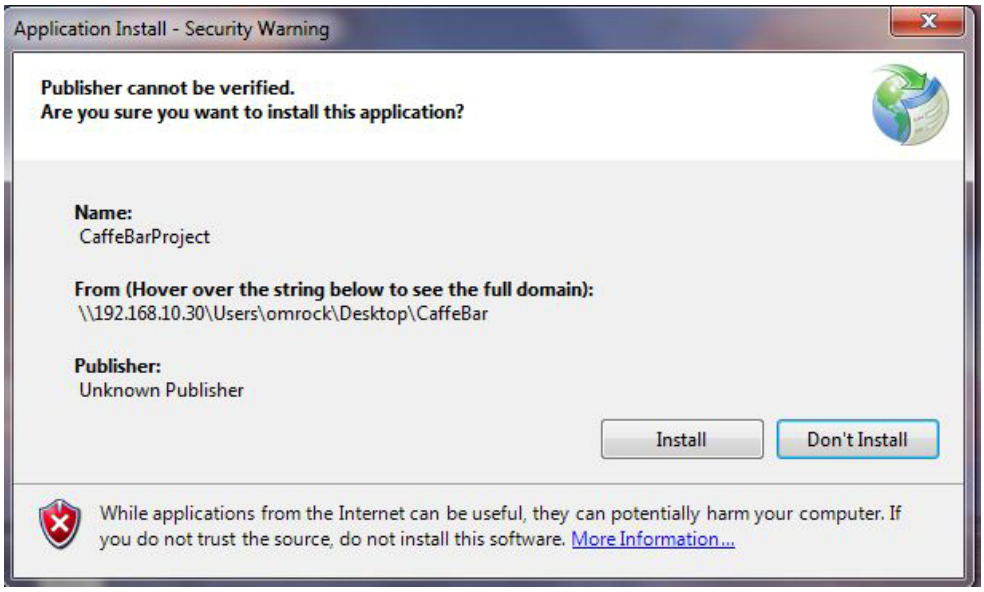
Εικόνα 21: Setup.exe

Πατάμε διπλό κλικ πάνω στο Setup.exe για να αρχίσει η εγκατάσταση τις εφαρμογής πάνω στον ηλεκτρονικό υπολογιστή.



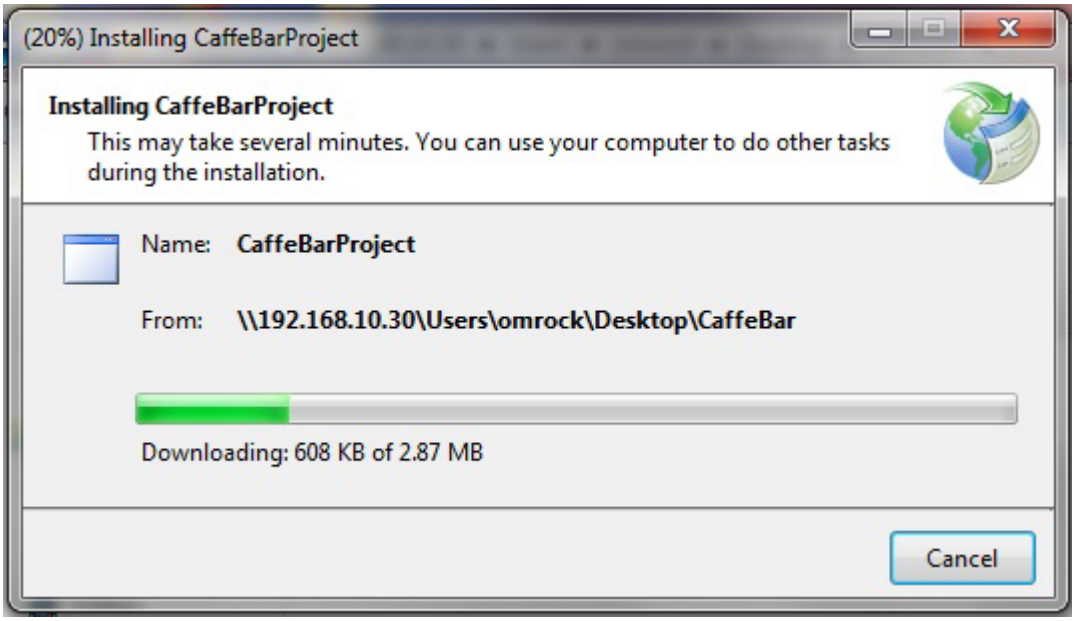
Εικόνα 22: Security warning.

Εμφανίζεται ένα παράθυρο προειδοποίησης στο οποίο πατάμε Run για να συνεχίσει η εγκατάσταση της εφαρμογής.



Εικόνα 23: Install.

Στην συνέχεια κάνουμε αριστερό κλικ πάνω στο Install για να ολοκληρωθεί η εγκατάσταση της εφαρμογής.

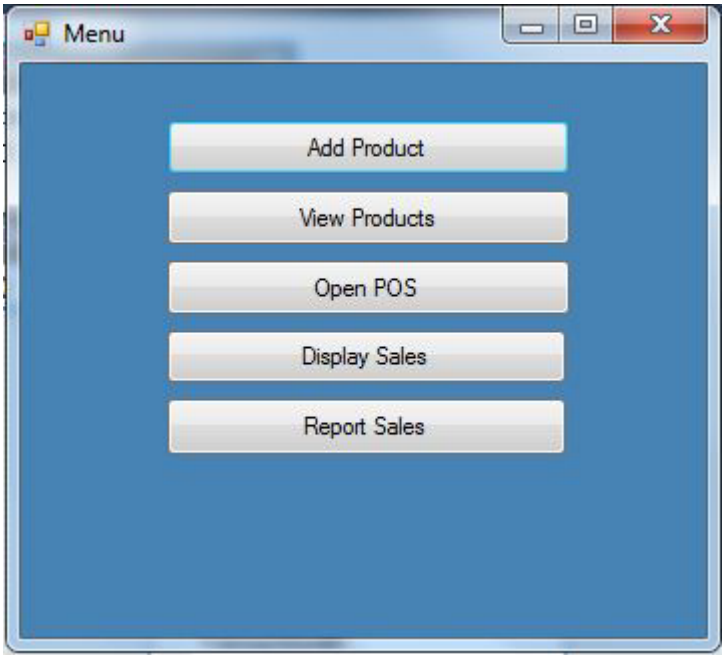


Εικόνα 24: Installing status.

Ακολούθως φορτώνεται η εφαρμογή στον ηλεκτρονικό υπολογιστή και όταν φτάσει στο 100% έχει ολοκληρωθεί η εγκατάσταση και είναι έτοιμο για χρήση.

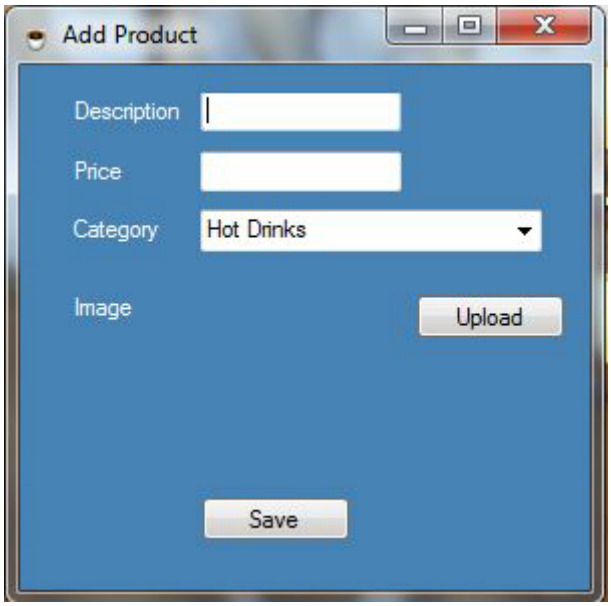
6.2 Εγχειρίδιο εφαρμογής

Με το άνοιγμα της εφαρμογής εμφανίζεται το κυρίως μενού όπου έχουμε διάφορες επιλογές να κάνουμε.



Εικόνα 25: Main Menu.

Πρώτα πρέπει να προσθέσουμε προϊόντα στην βάση δεδομένων μας. Αυτό το κάνουμε με το να πατήσουμε αριστερό κλικ στο Add Product στο κυρίως μενού.

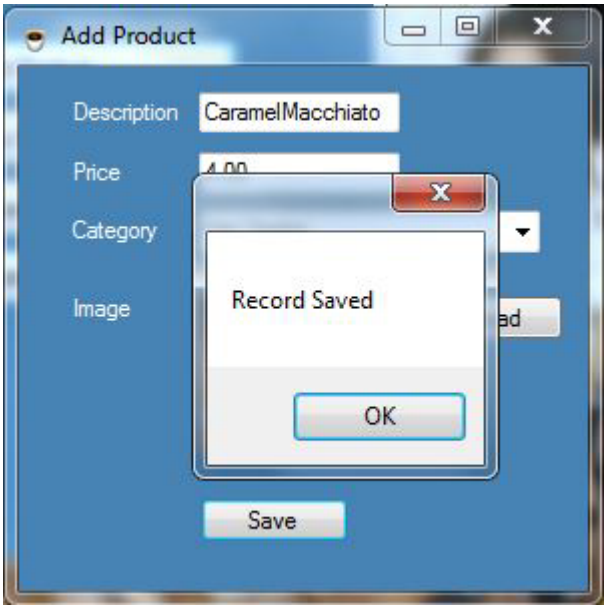


Εικόνα 26: Οθόνη προσθήκη προϊόντος.

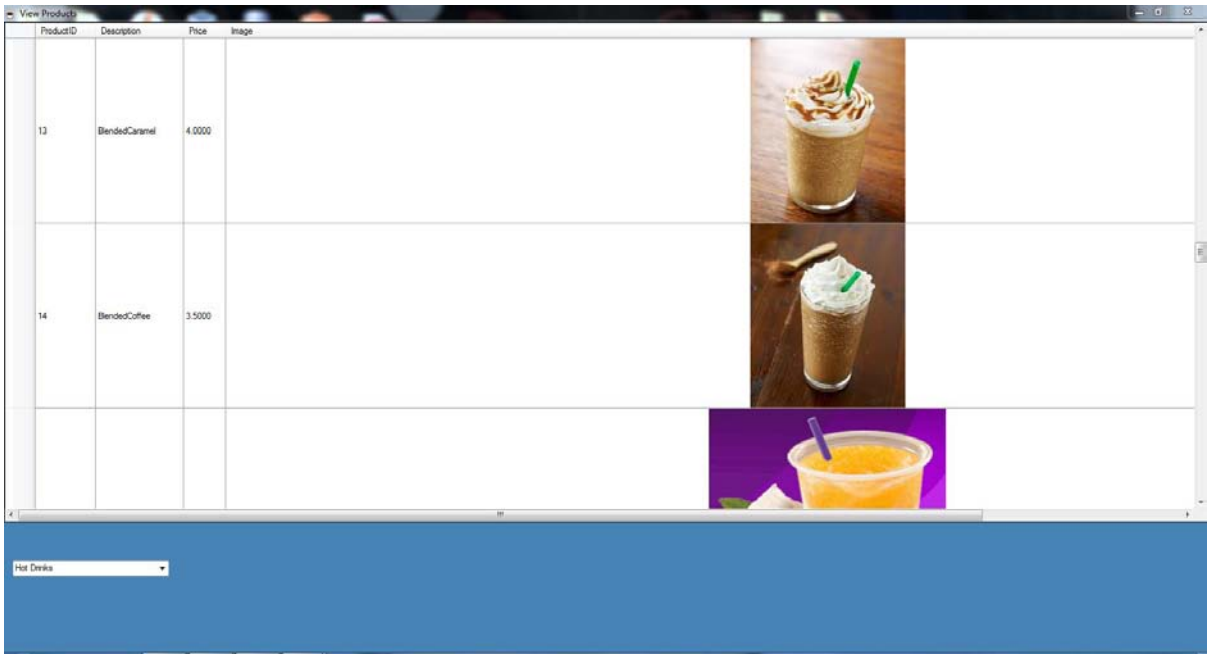
Εμφανίζεται η πιο πάνω οθόνη και καλείται ο χρήστης να συμπληρώσει τις πληροφορίες που ζητά η φόρμα εισαγωγείς προϊόντος ώστε να προστεθεί το προϊόν στην βάση δεδομένων.

Εικόνα 27: Συμπλήρωμα οθόνης προσθήκη προϊόντος.

Εικόνα 28: Συμπληρωμένη φόρμα εισαγωγής προϊόντος.

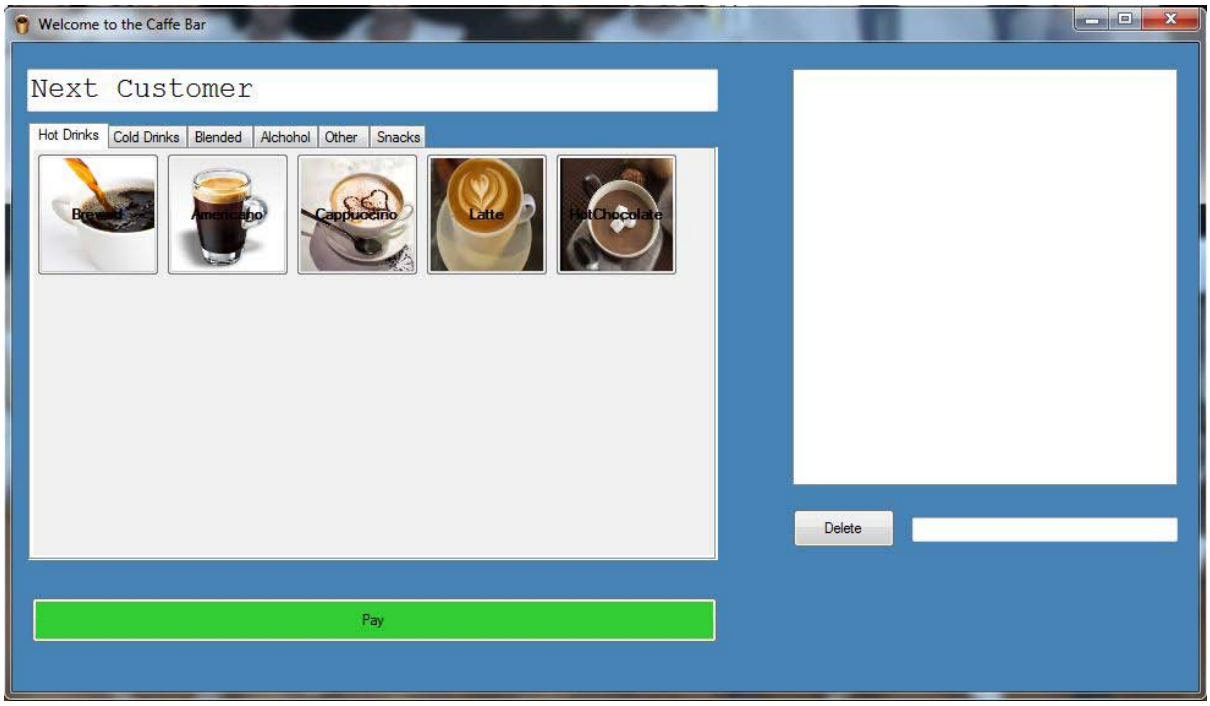


Εικόνα 29: Αποθήκευση προϊόντος.



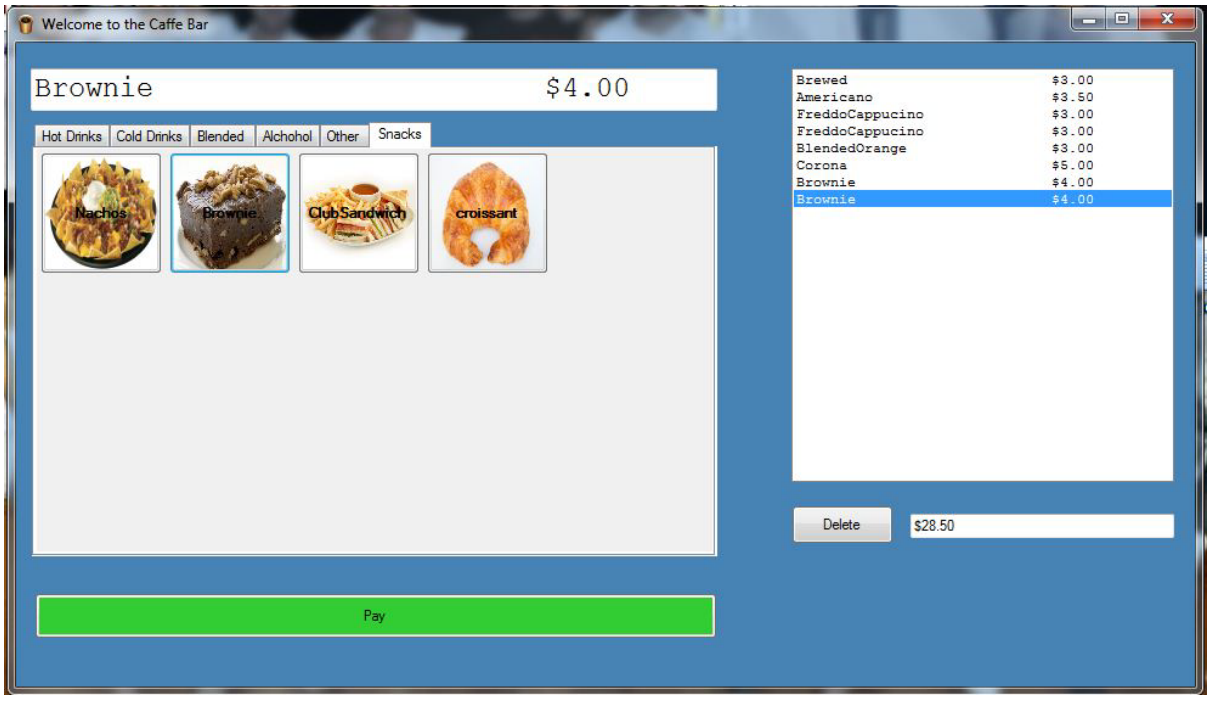
Εικόνα 30: Λίστα προϊόντων.

Πατώντας αριστερό κλικ στο View Products στο κυρίως μενού ο χρήστης μπορεί να δει λίστα με όλα τα προϊόντα που έχει στην βάση δεδομένων του.



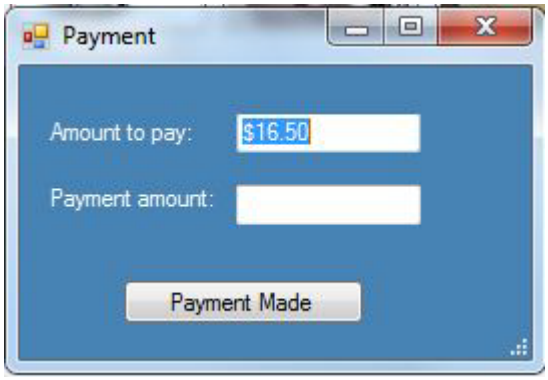
Εικόνα 31: POS.

Πατώντας αριστερό κλικ στο Open POS στο κυρίως μενού ο χρήστης μπορεί να μεταβεί στην οθόνη του POS.



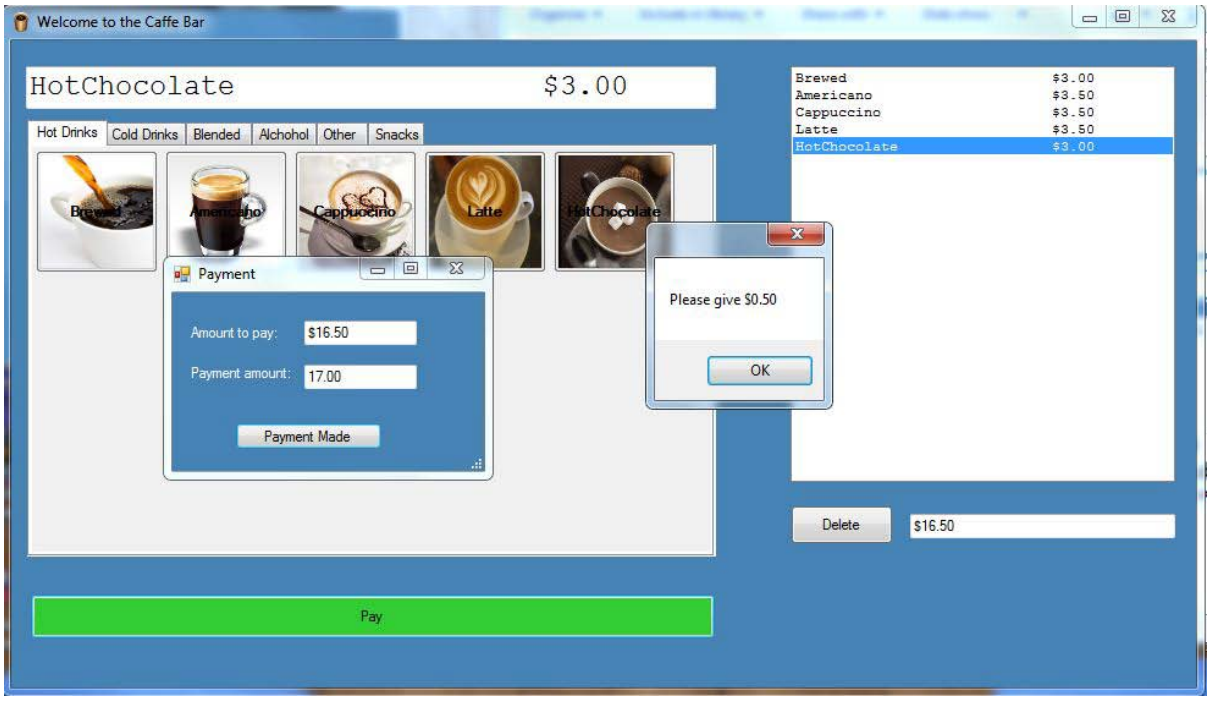
Εικόνα 32: Δημιουργία παραγγελίας.

Έπειτα ο χρήστης μπορεί να δημιουργήσει παραγγελία επιλέγοντας διάφορα προϊόντα από την οθόνη του POS.



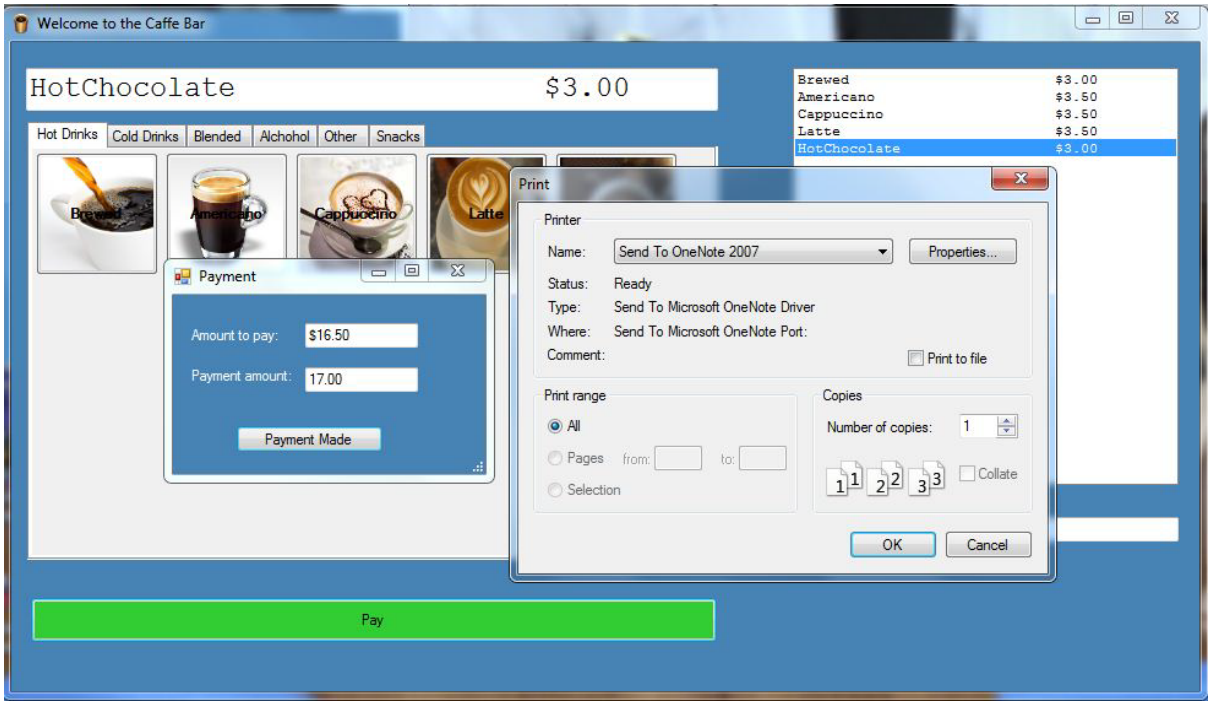
Εικόνα 33: Payment.

Πατώντας αριστερό κλικ στο Pay μπορεί να εκτελέσει την πληρωμή της παραγγελίας.



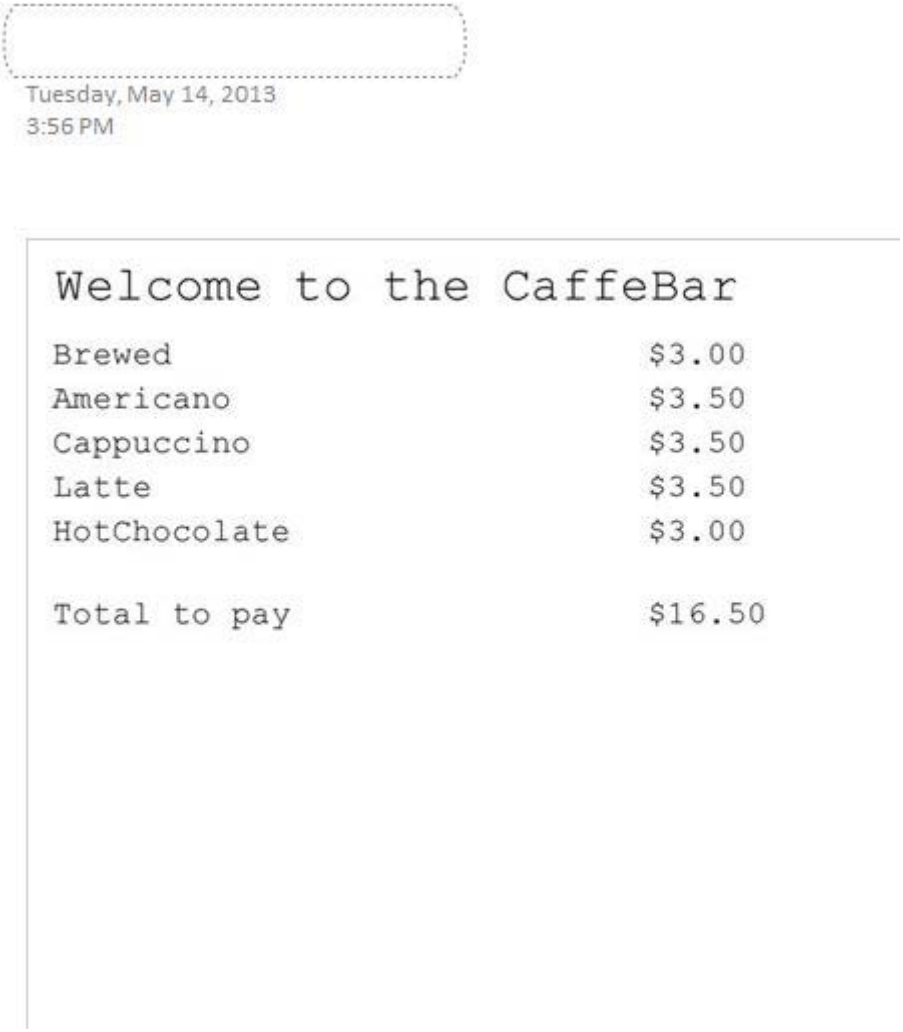
Εικόνα 34: Payment has been made.

Όταν εκτελεστεί η πληρωμή της παραγγελίας εμφανίζεται παράθυρο με τυχόν ρέστα που πρέπει να επιστραφούν και με την επιλογή του OK ετοιμάζεται να εκτυπωθεί η απόδειξη.



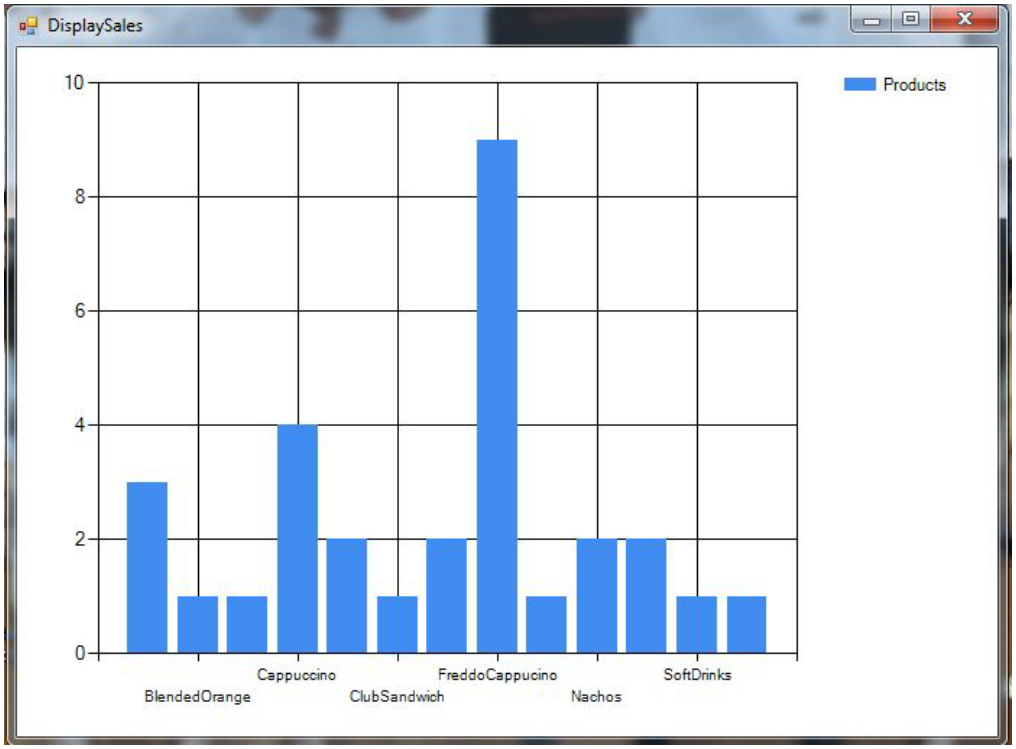
Εικόνα 35: Print receipt.

Επιλέγοντας μετά OK στο παράθυρο εκτύπωσης εκτυπώνεται η απόδειξη του πελάτη.



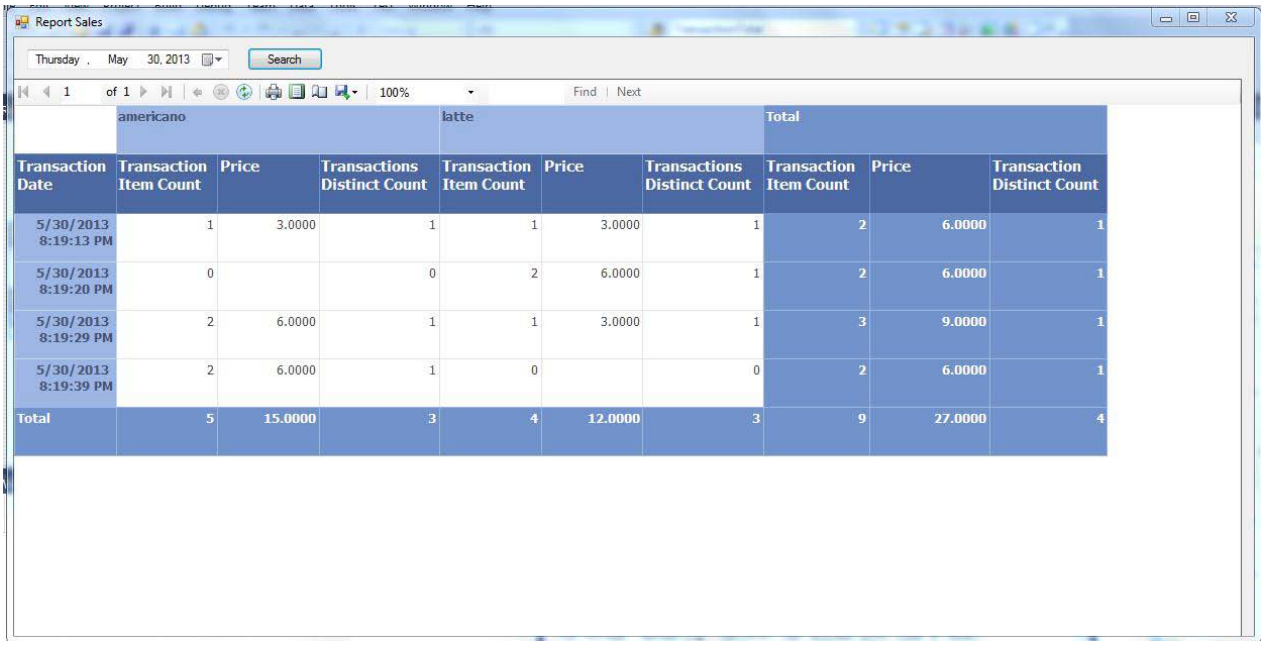
Εικόνα 36: Receipt.

Από το κυρίως μενού ο χρήστης με την επιλογή Display Sales μπορεί να δει γραφική παράσταση με τις πωλήσεις που έχει κάνει μέχρι στιγμής.



Εικόνα 37: Display Sales.

Με το πάτημα του κουμπιού Report Sales από το κυρίως μενού, ο χρήστης έχει την δυνατότητα να επιλέξει μια συγκεκριμένη ημερομηνία και να δει report με τις πωλήσεις των προϊόντων της συγκεκριμένης ημέρας ανά ώρα, αναγράφοντας τα transaction και συνολικές τιμές.



	americano			latte			Total		
Transaction Date	Transaction Item Count	Price	Transactions Distinct Count	Transaction Item Count	Price	Transactions Distinct Count	Transaction Item Count	Price	Transaction Distinct Count
5/30/2013 8:19:13 PM	1	3.0000	1	1	3.0000	1	2	6.0000	1
5/30/2013 8:19:20 PM	0		0	2	6.0000	1	2	6.0000	1
5/30/2013 8:19:29 PM	2	6.0000	1	1	3.0000	1	3	9.0000	1
5/30/2013 8:19:39 PM	2	6.0000	1	0		0	2	6.0000	1
Total	5	15.0000	3	4	12.0000	3	9	27.0000	4

Εικόνα 38: Report Sales

Κεφάλαιο 7

Συμπεράσματα

7.1 Ανασκόπηση της εφαρμογής 74
7.2 Αξιολόγηση – Συμπεράσματα 75

7.1 Ανασκόπηση της εφαρμογής

Σε αυτή την διπλωματική εργασία πετύχαμε την υλοποίηση μιας αποδοτικής εφαρμογής για POS καφετέριας μέσω της οποίας ο χρήστης θα μπορεί να διαχειρίζεται τις παραγγελίες και πληρωμές των παραγγελιών.

Ξεκινήσαμε με μια γενική μελέτη γύρω από το θέμα των καφετεριών και από εκεί εξάγαμε τις απαιτήσεις που πρέπει να ικανοποιούνται από την εφαρμογή.
Μετά μελετήθηκε σε βάθος ο τρόπος λειτουργίας των καφετεριών πάνω στο οποίο βασίστηκε σε τελική ανάλυση η εφαρμογή.

Έχοντας τα προηγούμενα ως υπόψη προχωρήσαμε σε μια εκτενή έρευνα γύρω από τις διάφορες διαθέσιμες τεχνολογίες που θα χρειαζόμαστε για να προχωρήσουμε στην ανάπτυξη του λογισμικού. Και μετά από ανάλυση τους καταλήξαμε στις πιο ιδανικές.

Προχωρώντας στην φάση της σχεδίασης έγινε μια εις βάθος διερεύνηση των τελικών προδιαγραφών που απαιτούνται από το σύστημα και εξάγαμε την δομή την οποία θα ακολουθούσε η εφαρμογή.

Γνωρίζοντας πλέον τη δομή προχωρήσαμε στην τελική φάση της σχεδίασης καθορίζοντας τις διάφορες λειτουργίες από τις οποίες θα απαρτίζετε η εφαρμογή ψάχνοντας συνεχώς για τους πιο αποδοτικούς τρόπους υλοποίησης τους. Τέλος προχωρήσαμε στην υλοποίηση της εφαρμογής.

7.2 Αξιολόγηση – Συμπεράσματα

Η εφαρμογή που υλοποιήθηκε αποτελεί μια αποδοτική εφαρμογή η οποία μπορεί να χρησιμοποιηθεί για να λειτουργήσει πλήρως κάποιος μια καφετέρια. Όπως είδαμε κτίσθηκε ανάλογα με τον τρόπο λειτουργίας μιας self service καφετέριας ώστε να πληροί τις απαιτήσεις της καφετέριας στην λειτουργικότητα και χρηστικότητα.

Σχετικά με την αρχιτεκτονική της εφαρμογής είδαμε πως τα Windows Forms μαζί με την ενσωματωμένη βάση δεδομένων έδωσαν το καλύτερο αποτέλεσμα για την δημιουργία της εφαρμογής, για τον αποδοτικό σχεδιασμό και εύχρηστο περιβάλλον που να κάνει την δουλεία του χρήστη πιο εύκολη. Άρα ο στόχος που τέθηκε για την δημιουργία μιας efficient και επεκτάσιμης εφαρμογής ικανοποιήθηκε.

Επίσης μέσω αυτής της εργασίας μελετήθηκαν οι βασικές λειτουργίες μιας εφαρμογής POS οι οποίες ορίσθηκαν στις προδιαγραφές του συστήματος μας και υλοποιήθηκαν με επιτυχία. Επίσης η απλότητα και οι συνεχείς οδηγίες που παρέχονται από το σύστημα καθιστούν την εφαρμογή εύχρηστη αφού δεν προαπαιτεί εξειδικευμένες γνώσεις για το χειρισμό της.

Βιβλιογραφία

Classic Cafes, The Coming Of The cafes, April 10, 2013, http://www.classiccafes.co.uk/History.html

Humboldt Bay Coffee Company, About Us, Where Did Coffee Come From?, April 11,2013,http://web.archive.org/web/20070915014128/http://www.humboldtcoffee.com/History.htm

Massachusetts Travel journal, America's First Coffeehouse, April 15th, 2013, http://masstraveljournal.com/page/massachusetts-firsts/americas-first-coffeehouse

Ray Oldenburg, "The Great Good Place: Cafes, Coffee Shops, Community Centers, General Stores, Bars, Hangouts, and How They Get You through the Day". New York: Paragon Books, 1989.

Petzold, Charles (2002). "Programming Microsoft Windows with C#". Microsoft Press.

C# Tutorial and Source Code, December 1, 2012, http://csharp.net-informations.com/

C# Programming Guide, December 3, 2012,http://msdn.microsoft.com/en-us/library/67ef8sbd.aspx

H. van Vliet, "Software Engineering: Principles and Practice", Third edition, John Wiley & Sons, 2008.

P. Stevens, R. Pooley, "Using UML Software Engineering with Objects and Components", second edition, Addison-Wesley, 2006.

Msdn, Windows Forms, December 10, 2012, http://msdn.microsoft.com/en-us/library/dd30h2yb.aspx

Msdn, Automating windows forms, December 12, 2012, http://msdn.microsoft.com/en-us/library/ms996405.aspx

Code Project, Beginning C# - Chapter 13: Using Windows Form Controls, December 12, 2012, http://www.codeproject.com/Articles/1465/Beginning-C-Chapter-13-Using-Windows-Form-Controls

Παράρτημα Α – Απόσπασμα Κώδικα

Add Product.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;

namespace CaffeBarProject
{
    public partial class AddProduct : Form
    {

        private CaffeBarDatabaseEntities cbe = new CaffeBarDatabaseEntities();

        private Byte[] byteBLOBData;

        public AddProduct()
        {
            InitializeComponent();

            cboCategory.DataSource = cbe.ProductTypes;
            cboCategory.DisplayMember = "Description";
            cboCategory.ValueMember = "ProductType1";


        }

        private void btnUploadImg_Click(object sender, EventArgs e)
        {
            DialogResult result = openFileDialog1.ShowDialog();

            if (result == DialogResult.OK)
            {
                FileStream fsBLOBFile = new FileStream(openFileDialog1.FileName,
FileMode.Open, FileAccess.Read);

                byteBLOBData = new Byte[fsBLOBFile.Length];

                fsBLOBFile.Read(byteBLOBData, 0, byteBLOBData.Length);
                fsBLOBFile.Close();

                MemoryStream stmBLOBData = new MemoryStream(byteBLOBData);

                pbImage.Image = Image.FromStream(stmBLOBData);

            }

        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            Product product = new Product();

            product.Description = txtDescription.Text;

            product.Price = decimal.Parse(txtPrice.Text);
```

```
            product.Image = byteBLOBData;

            product.ProductType = (int)cboCategory.SelectedValue;

            cbe.AddToProducts(product);

            cbe.SaveChanges();

            MessageBox.Show("Record Saved");
        }

    }
}
```

CaffeBarPOS.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.Objects;
using System.Drawing.Printing;
using System.IO;

namespace CaffeBarProject
{
    public partial class CaffeBarPOS : Form
    {


        private BindingList<Product> products = new BindingList<Product>();

        private CaffeBarDatabaseEntities cbe = new CaffeBarDatabaseEntities();

        private decimal transactionTotal;

        public decimal TransactionTotal
        {
            get { return transactionTotal; }
            set
            {
               transactionTotal = value;
               txtTotal.Text = String.Format("{0:c}", transactionTotal);

            }
        }



        public CaffeBarPOS()
        {

            InitializeComponent();

            lstProductsChosen.DataSource = products;
            lstProductsChosen.DisplayMember = "Description";
```

```
            CreateTabbedPannel();

            AddProductsToTabbedPanel();

        }


        private void AddProductsToTabbedPanel()
        {
            int i = 1;

            foreach (TabPage tp in tabControl1.TabPages)
            {
                ObjectQuery<Product> filteredProduct = new
ObjectQuery<Product>("SELECT VALUE P FROM Products AS P WHERE P.ProductType = " +
i.ToString(), cbe);

                FlowLayoutPanel flp = new FlowLayoutPanel();

                flp.Dock = DockStyle.Fill;

                foreach (Product tprod in filteredProduct)
                {
                    Button b = new Button();
                    b.Size = new Size(100, 100);
                    b.Text = tprod.Description;
                    b.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
                    b.Font = new Font(b.Font, FontStyle.Bold);
                    byte[] data = (byte[])tprod.Image;
                    MemoryStream ms = new MemoryStream(data);
                    b.BackgroundImage = Image.FromStream(ms);
                    b.BackgroundImageLayout = ImageLayout.Stretch;
                    b.Tag = tprod;
                    b.Click += new EventHandler(UpdateProductList);
                    flp.Controls.Add(b);



                }

                tp.Controls.Add(flp);
                i++;
            }

        }

        void UpdateProductList(object sender, EventArgs e)
        {
            Button b = (Button)sender;

            Product p = (Product)b.Tag;

            products.Add(p);

            UpdateCustomerPanel(p);

            TransactionTotal = TransactionTotal + (decimal)p.Price;

            lstProductsChosen.SelectedIndex = lstProductsChosen.Items.Count - 1;

        }

        private void UpdateCustomerPanel(Product product)
        {

            string currentDescription = product.Description;
```

```
            string currentPrice = String.Format("{0:c}", product.Price);
            string currentDescriptionPadded = currentDescription.PadRight(30);

            txtInfoPanel.Text = currentDescriptionPadded + currentPrice;
        }

        private void CreateTabbedPannel()
        {

            foreach (ProductType pt in cbe.ProductTypes)
            {
                tabControl1.TabPages.Add(pt.ProductType1.ToString(), pt.Description);

            }
        }


        private void button1_Click(object sender, EventArgs e)
        {
            Product p = new Product() { Description = "Product A", Price = 1.99M };
            products.Add(p);
        }

        private void FormatListItem(object sender, ListControlConvertEventArgs e)
        {
            string currentDescription = ((Product)e.ListItem).Description;
            string currentPrice = String.Format("{0:c}", ((Product)e.ListItem).Price);
            string currentDescriptionPadded = currentDescription.PadRight(30);

            e.Value = currentDescriptionPadded + currentPrice;
        }

        private void DeleteItem(object sender, EventArgs e)
        {
            Product selectedProduct = (Product)lstProductsChosen.SelectedItem;
            products.Remove(selectedProduct);
            if (TransactionTotal>0)
             TransactionTotal = TransactionTotal - (decimal)selectedProduct.Price;
        }

        private void OpenPayment(object sender, EventArgs e)
        {
            Payment payment = new Payment();

            payment.PaymentMade += new Payment.PaymentMadeEvent(paymentSuccess);
            payment.PaymentAmount = TransactionTotal;

            payment.ShowDialog();

            txtInfoPanel.Text = "Next Customer";
            products.Clear();
            TransactionTotal = 0;

        }

        private void PrintReceipt()
        {
            PrintDialog printDialog = new PrintDialog();

            PrintDocument printDocument = new PrintDocument();

            printDialog.Document = printDocument;

            printDocument.PrintPage += new
PrintPageEventHandler(printDocument_PrintPage);
```

```
            DialogResult result = printDialog.ShowDialog();

            if (result == DialogResult.OK)
            {
                printDocument.Print();
            }

        }

        void printDocument_PrintPage(object sender, PrintPageEventArgs e)
        {
            Graphics graphic = e.Graphics;

            Font font = new Font("Courier New", 12);

            float fontHeight = font.GetHeight();

            int startX = 10;
            int startY = 10;
            int offset = 40;

            graphic.DrawString("Welcome to the CaffeBar", new Font("Courier New", 18),
new SolidBrush(Color.Black), startX, startY);

            foreach (Product product in products)
            {
                string productDescription = product.Description.PadRight(30);
                string productTotal = String.Format("{0:c}", product.Price);
                string productLine = productDescription + productTotal;

                graphic.DrawString(productLine, font, new SolidBrush(Color.Black),
startX, startY + offset);

                offset = offset + (int)fontHeight + 5;


            }

            offset = offset + 20;

            graphic.DrawString("Total to pay".PadRight(30) + String.Format("{0:c}",
TransactionTotal), font, new SolidBrush(Color.Black), startX, startY + offset);


        }



        void paymentSuccess(object sender, PaymentMadeEventArgs e)
        {
            Transaction transaction = new Transaction();
            transaction.TransactionDate = DateTime.Now;


            if (e.PaymentSucess == true)
            {
                //save the current transaction

                foreach (Product product in products)
                {
                    transaction.TransactionItems.Add(new TransactionItem() { ProductID
= product.ProductID });

                }

                cbe.AddToTransactions(transaction);
```

```
                cbe.SaveChanges();
                PrintReceipt();

            }
        }


        public byte[] byteImages { get; set; }
    }
}
```

Form1.cs(Main menu)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace CaffeBarProject
{
    public partial class Menu : Form
    {
        public Menu()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            AddProduct addProduct = new AddProduct();
            addProduct.Show();

        }

        private void button2_Click(object sender, EventArgs e)
        {
            ViewProducts viewProducts = new ViewProducts();
            viewProducts.Show();
        }

        private void btnOpenPOS_Click(object sender, EventArgs e)
        {
            CaffeBarPOS caffeBarPOS = new CaffeBarPOS();

            caffeBarPOS.Show();
        }

        private void btnDisplaySales_Click(object sender, EventArgs e)
        {
            DisplaySales displaySales = new DisplaySales();
            displaySales.Show();

        }

       }
}
```

Payment.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace CaffeBarProject
{
    public partial class Payment : Form
    {
        public delegate void PaymentMadeEvent(object sender, PaymentMadeEventArgs e);

        public event PaymentMadeEvent PaymentMade;

        private decimal paymentAmount;

        public decimal PaymentAmount
        {
            get { return paymentAmount; }
            set {
                    paymentAmount = value;
                    txtAmountToPay.Text = String.Format("{0:c}", paymentAmount);
                }
        }

        public Payment()
        {
            InitializeComponent();
        }

        private void PaymentHasBeenMade(object sender, EventArgs e)
        {
            decimal total = 0;

            try
            {
                total = decimal.Parse(txtAmountToPay.Text.TrimStart('$')) -
decimal.Parse(txtPaymentAmount.Text);
            }
            catch
            {
                MessageBox.Show("Error, please enter a valid amount");
                return;
            }

            if (total > 0)
            {
                txtAmountToPay.Text = total.ToString();
            }
            else
            {
                MessageBox.Show("Please give " + String.Format("{0:c}", -total));
                PaymentMade(this, new PaymentMadeEventArgs() { PaymentSucess = true
});
            }



        }
```

```
    }

    public class PaymentMadeEventArgs: EventArgs
    {
        private bool paymentSucess;

public bool PaymentSucess
{
  get { return paymentSucess; }
  set { paymentSucess = value; }
}

    }
}
```

ViewProducts.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.Objects;

namespace CaffeBarProject
{
    public partial class ViewProducts : Form
    {

        private CaffeBarDatabaseEntities cbe = new CaffeBarDatabaseEntities();

        public ViewProducts()
        {
            InitializeComponent();

            dataGridView1.DataSource = cbe.Products;

            dataGridView1.Columns["ProductType"].Visible = false;
            dataGridView1.Columns["ProductType1"].Visible = false;
            dataGridView1.Columns["TransactionItems"].Visible = false;

            cboFilter.DataSource = cbe.ProductTypes;
            cboFilter.DisplayMember = "Description";
            cboFilter.ValueMember = "ProductType1";

        }

        private void FilterList(object sender, EventArgs e)
        {
            ObjectQuery<Product> filteredProducts = new ObjectQuery<Product>(
                "SELECT VALUE product FROM Products AS product WHERE
product.ProductType = " + cboFilter.SelectedValue, cbe);

            dataGridView1.DataSource = filteredProducts;

        }
    }
}
```

DisplaySales.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace CaffeBarProject
{
    public partial class DisplaySales : Form
    {
        public DisplaySales()
        {
            InitializeComponent();
            generateGraph();
        }

        private void generateGraph()
        {
            using (CaffeBarDatabaseEntities cbe = new CaffeBarDatabaseEntities())
            {
                var query = from product in cbe.TransactionItems
                            group product by product.Product.Description into g
                            select new {ProductID = g.Key, TotalUnitsSold = g.Count()
};

                chart1.DataSource = query;

                chart1.Series["Series1"].XValueMember = "ProductID";
                chart1.Series["Series1"].YValueMembers = "TotalUnitsSold";

                chart1.Series["Series1"].Name = "Products";

                chart1.DataBind();

                chart1.Show();

            }

        }

    }
}
```