

# PDN System

## Use Cases

---

**Github repository:** <https://github.com/loannisKaldiris/PayPoint/tree/main>

**Αριθμός Εγγράφου: 005**

### Ιστορικό αναθεώρησης

Revision	Date	Description
A01	11/05/2024	Original Version
A02	30/05/2024	Προσθήκη καινούργιο κώδικα και τελευταία έκδοση του document

## Σύνθεση Ομάδας

Ονοματεπώνυμο		ΑΜ	Έτος	Email
Μέλος 1 <sup>ο</sup>	Καλδίρης Ιωάννης	1080428	5ο	up1080428@ac.upatras.gr
Μέλος 2 <sup>ο</sup>	Παπαδόπουλος Περικλής	1084540	4ο	up1084540@ac.upatras.gr
Μέλος 3 <sup>ο</sup>	Γιαννόπουλος Χαράλαμπος	1064037	7ο	up1064037@ac.upatras.gr
Μέλος 4 <sup>ο</sup>	Γιαννέλος Στάθης	1048394	8ο	up1048394@ac.upatras.gr

## Εισαγωγή

Στον παρόν τεχνικό κείμενο σας παρέχουμε μία προεπισκόπηση του κώδικα που γράψαμε για την υλοποίηση του PayPoint software. Για να μεταβεί κάποιος κατευθείαν στο GitHub repo, δηλαδή εκεί που έχουμε τον κώδικα μας, αρκεί απλά κάποιος να πατήσει το παρακάτω κουμπί:

GitHub Repo =>

[https://github.com/loannisKaldiris/PayPoint/  
tree/main](https://github.com/loannisKaldiris/PayPoint/tree/main)

## Οδηγίες Εκτέλεσης

Αυτός ο οδηγός παρέχει βήμα προς βήμα οδηγίες για το πώς να γίνει setup και να εκτελεστούν τα use case που χρησιμοποιούν την βιβλιοθήκη PyQt6 για τη υλοποίηση της εφαρμογής PayPoint. Πριν από την εκτέλεση του σεναρίου, είναι απαραίτητο να εγκατασταθούν όλα τα απαραίτητα πακέτα Python και να ρυθμιστεί σωστά το περιβάλλον.

### Προαπαιτούμενα

1. **Εγκατάσταση της Python:** Αρχικά θα πρέπει να βεβαιωθούμε ότι η Python είναι εγκατεστημένη στο υπολογιστικό σύστημα σας. Η εφαρμογή αναπτύχθηκε στο περιβάλλον Python 3.12.3 (tags/v3.12.3:f665f9, Apr 9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win 32, παρόλα αυτά λόγω της συμβατότητας της βιβλιοθήκης PyQt6 είναι δυνατόν να τρέξει σε έκδοσης Python από Python 3.6 και μεταγενέστερες εκδόσεις. Μπορείτε να κατεβάσετε και να εγκαταστήσετε την Python από τον [επίσημο ιστότοπο](#).
2. **IDE περιβάλλον:** Αν και δεν είναι απολύτως απαραίτητο, η ύπαρξη ενός IDE όπως το VSCode, το Sublime Text ή το PyCharm συμβουλεύεται. Το παρόν σύστημα αναπτύχθηκε με την βοήθεια του IDE PyCharm.
3. **Εγκατάσταση XAMPP:** Το XAMPP θα παρέχει το τοπικό περιβάλλον του server που είναι απαραίτητο για τη φιλοξενία της βάσης δεδομένων MySQL.

### Οδηγός για την εγκατάσταση του XAMPP

Βήμα 1: Εγκατάσταση του XAMPP

Λήψη του XAMPP: Επισκεφθείτε αυτήν την [ιστοσελίδα](#) και κατεβάστε το πρόγραμμα εγκατάστασης του XAMPP για το λειτουργικό σας σύστημα.

Εκτελέστε το πρόγραμμα εγκατάστασης που κατεβάσατε.

Ακολουθήστε τις οδηγίες που εμφανίζονται στην οθόνη. Συνιστάται να αποδεχτείτε τις προεπιλεγμένες ρυθμίσεις, εκτός αν έχετε συγκεκριμένες απαιτήσεις.

Βεβαιωθείτε ότι έχετε επιλέξει τις MySQL και phpMyAdmin ως στοιχεία προς εγκατάσταση, καθώς αυτά απαιτούνται για τη βάση δεδομένων σας.

Εκκινήστε το XAMPP:

- Εκκινήστε τον πίνακα ελέγχου του XAMPP.
- Εκκινήστε τις ενότητες Apache και MySQL. Θα πρέπει να δείτε την κατάστασή τους να γίνεται πράσινη στον πίνακα ελέγχου.

Αποκτήστε πρόσβαση στο phpMyAdmin:

- Ανοίξτε ένα πρόγραμμα περιήγησης στο διαδίκτυο και μεταβείτε στη διεύθυνση <http://localhost/phpmyadmin/>.

Αυτή η διεπαφή χρησιμοποιείται για τη διαχείριση των βάσεων δεδομένων MySQL.

Δημιουργήστε μια βάση δεδομένων:

Στο phpMyAdmin, κάντε κλικ στην καρτέλα "Databases" (Βάσεις δεδομένων) και δημιουργήστε μια νέα βάση δεδομένων για την εφαρμογή σας. Στην δική μας περίπτωση είτε φορτώνετε την έτοιμη βάση είτε τρέχετε το αρχείο mysql\_connector.py

### Αρχικό Setup

Βήμα 1: Λήψη του πηγαίου κώδικα

Αρχικά, θα πρέπει να έχετε όλα τα σχετικά αρχεία. Συμβουλεύεται εφόσον παρέχεται το αποθετήριο, να τα κλωνοποιήσετε ή να τα κατεβάσετε από την πηγή. Για περισσότερες πληροφορίες για το πως μπορεί να γίνει αυτό κάντε κλικ [εδώ](#).

Βήμα 2: Εγκατάσταση των απαιτούμενων πακέτων

Ο κώδικας μας απαιτεί συγκεκριμένες βιβλιοθήκες Python για να εκτελεστεί σωστά, συμπεριλαμβανομένης της PyQt6. Για τον λόγο αυτό έχουμε φτιάξει το αρχείο με όνομα requirements.txt που περιέχει όλα τα πακέτα που πρέπει να εγκατασταθούν. Ακολουθήστε αυτά τα βήματα για να τα εγκαταστήσετε:

- Ανοίξτε μία γραμμή εντολών ή ένα τερματικό: Προηγηθείτε στο folder path που μόλις κλωνοποιήσετε και βρίσκεται το αρχείο requirements.txt.
- Εγκαταστήστε τις εξαρτήσεις: Εκτελέστε την ακόλουθη εντολή για να εγκαταστήσετε τα απαιτούμενα πακέτα:

```
\PycharmProjects\Software_Engineer_2024>pip install -r requirements.txt
```

### Εκτέλεση του σεναρίου

Βήμα 3: Προετοιμάστε το περιβάλλον

Τα use case γενικότερα χρησιμοποιούν διάφορα modules που μπορεί να μην συμβατά προς τα πίσω με προηγούμενες εκδόσεις της Python

Βήμα 4: Εκτέλεση σεναρίου

Με τις εξαρτήσεις εγκατεστημένες, μπορείτε τώρα να εκτελέσετε τα use case:

```
C:\Users\user>python your_script_name.py
```

Και θα πρέπει να αντικατασταθεί το your\_script\_name.py με το πραγματικό όνομα του εκάστοτε script.

Αλληλεπίδραση με την εφαρμογή:

Η εφαρμογή που έχει αναπτυχθεί με την βοήθεια της βιβλιοθήκης PyQt6 και τώρα θα πρέπει να είστε έτοιμοι σε θέση να συνδεθείτε με την βάση δεδομένων MySQL που φιλοξενείται στο XAMPP, επιτρέποντάς σας να εκτελείτε λειτουργίες όπως εισαγωγή, ανάκτηση, ενημέρωση και διαγραφή δεδομένων.

Βήμα 5: Αντιμετώπιση προβλημάτων

Βεβαιωθείτε ότι όλες οι υπηρεσίες λειτουργούν: Βεβαιωθείτε ότι οι υπηρεσίες Apache και MySQL στο XAMPP εκτελούνται πριν ξεκινήσετε την εφαρμογή σας.

Προβλήματα σύνδεσης με τη βάση δεδομένων: Ελέγξτε τις λεπτομέρειες σύνδεσης με τη βάση δεδομένων

στο σενάριο Python (όπως ports), συμπεριλαμβανομένων του κεντρικού υπολογιστή, του ονόματος χρήστη, του κωδικού πρόσβασης και του ονόματος της βάσης δεδομένων.

## Use Case – Accept Cash

Για να τρέξει το συγκεκριμένο use case θα πρέπει να τρέξουν τα συγκεκριμένα τρία αρχεία:

- Main.py, περιλαμβάνει το POS application που διαχειρίζεται ο Cashier
- Server.py, περιλαμβάνει τον server του PayPoint που δέχεται και διαχειρίζεται τα requests από το POS application και επικοινωνεί με το hardware
- Send\_input\_denomination.py, κάνει simulated την είσοδο που θα έβαζε ο χρήστης στο PayPod (θεωρητικά αν είχαμε το hardware θα έπρεπε να κάνουμε τα αντίστοιχα API calls για να μπορέσουμε να έχουμε άμεση επικοινωνία με το hardware, στην περίπτωση μας το software αναπτύχθηκε στα πλαίσια του μαθήματος Τεχνολογία Λογισμικού και έτσι δεν υπήρχε η δυνατότητα να διατεθεί το συγκεκριμένο κεφάλαιο)

## Use Case – Cash Inventory and Refill

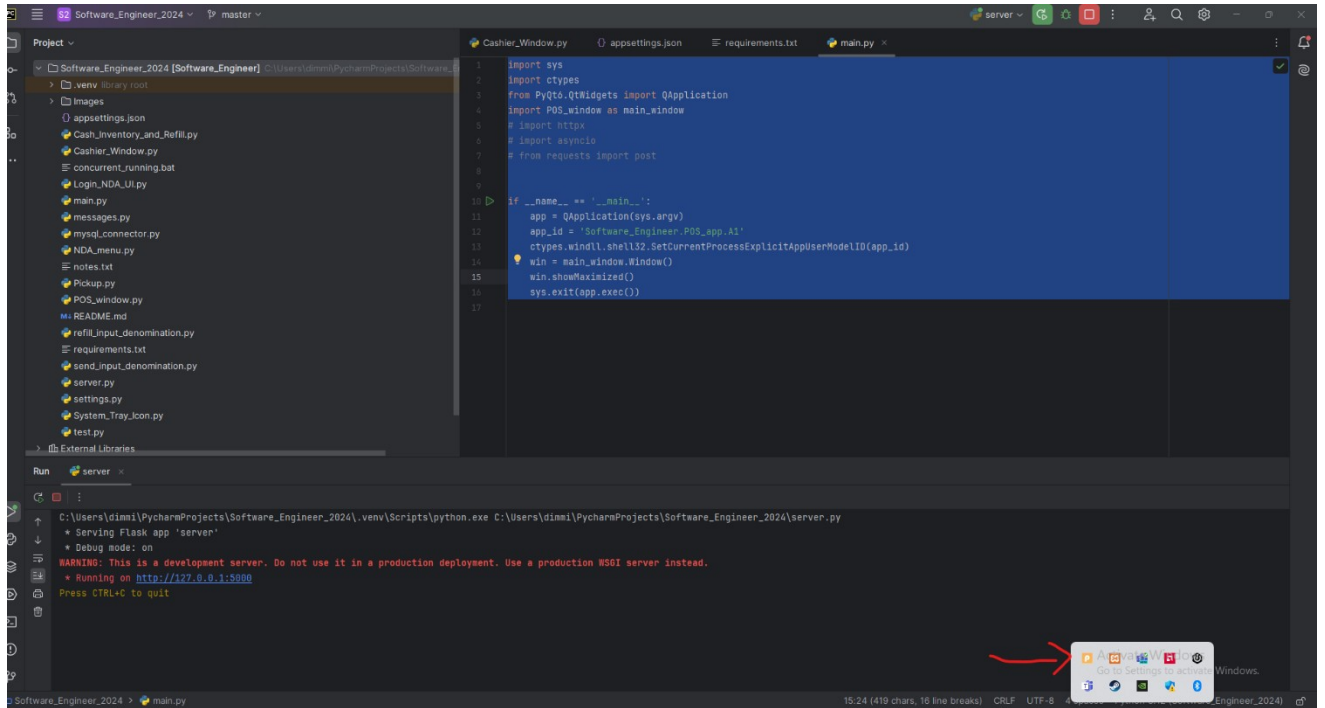
Για να τρέξει το συγκεκριμένο use case θα πρέπει να τρέξουν τα συγκεκριμένα δύο αρχεία:

- Server.py, περιλαμβάνει τον server του PayPoint που δέχεται και διαχειρίζεται τα requests από το POS application και επικοινωνεί με το hardware και επιπλέον δίνει την δυνατότητα στον χρήστη να συνδεθεί στο NDA menu.
- Send\_input\_denomination.py, κάνει simulated την είσοδο που θα έβαζε ο χρήστης στο PayPod (θεωρητικά αν είχαμε το hardware θα έπρεπε να κάνουμε τα αντίστοιχα API calls για να μπορέσουμε να έχουμε άμεση επικοινωνία με το hardware, στην περίπτωση μας το software αναπτύχθηκε στα πλαίσια του μαθήματος Τεχνολογία Λογισμικού και έτσι δεν υπήρχε η δυνατότητα να διατεθεί το συγκεκριμένο κεφάλαιο)

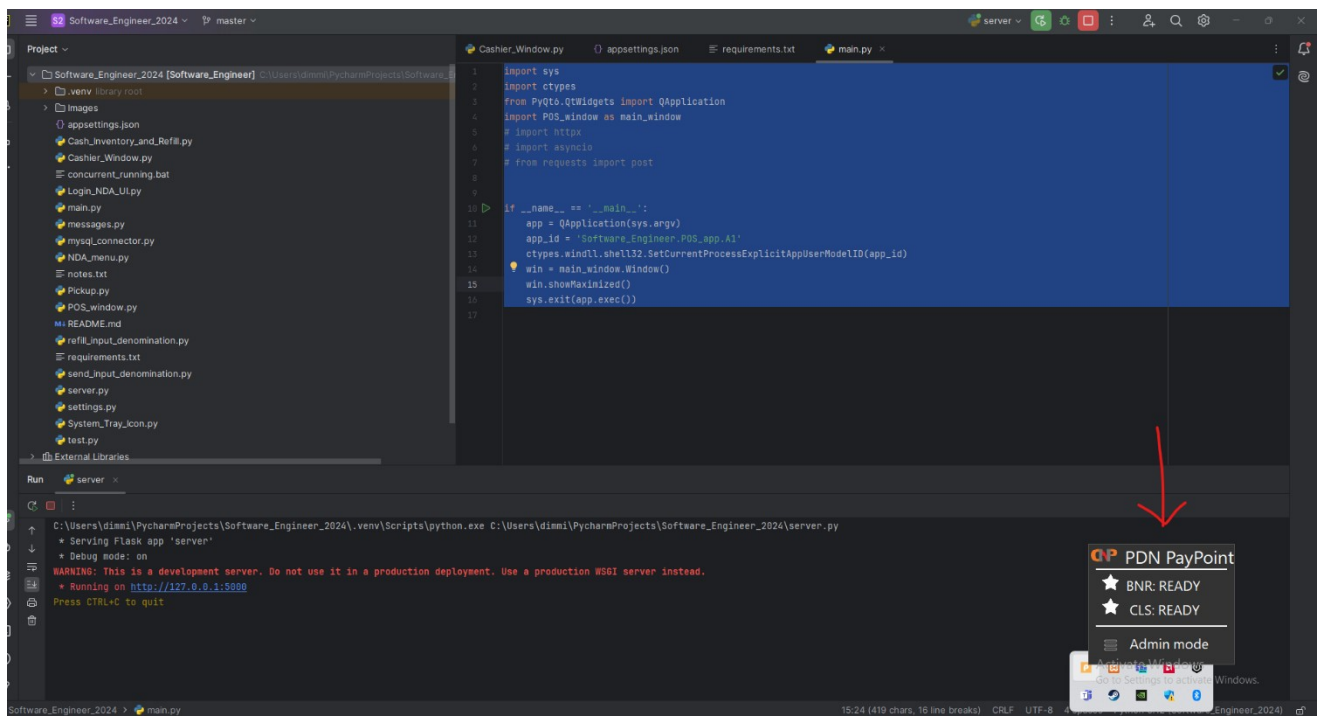
## Use Case – Cash Pickup

Για να τρέξει το συγκεκριμένο use case θα πρέπει να τρέξουν τα συγκεκριμένα ένα αρχείο:

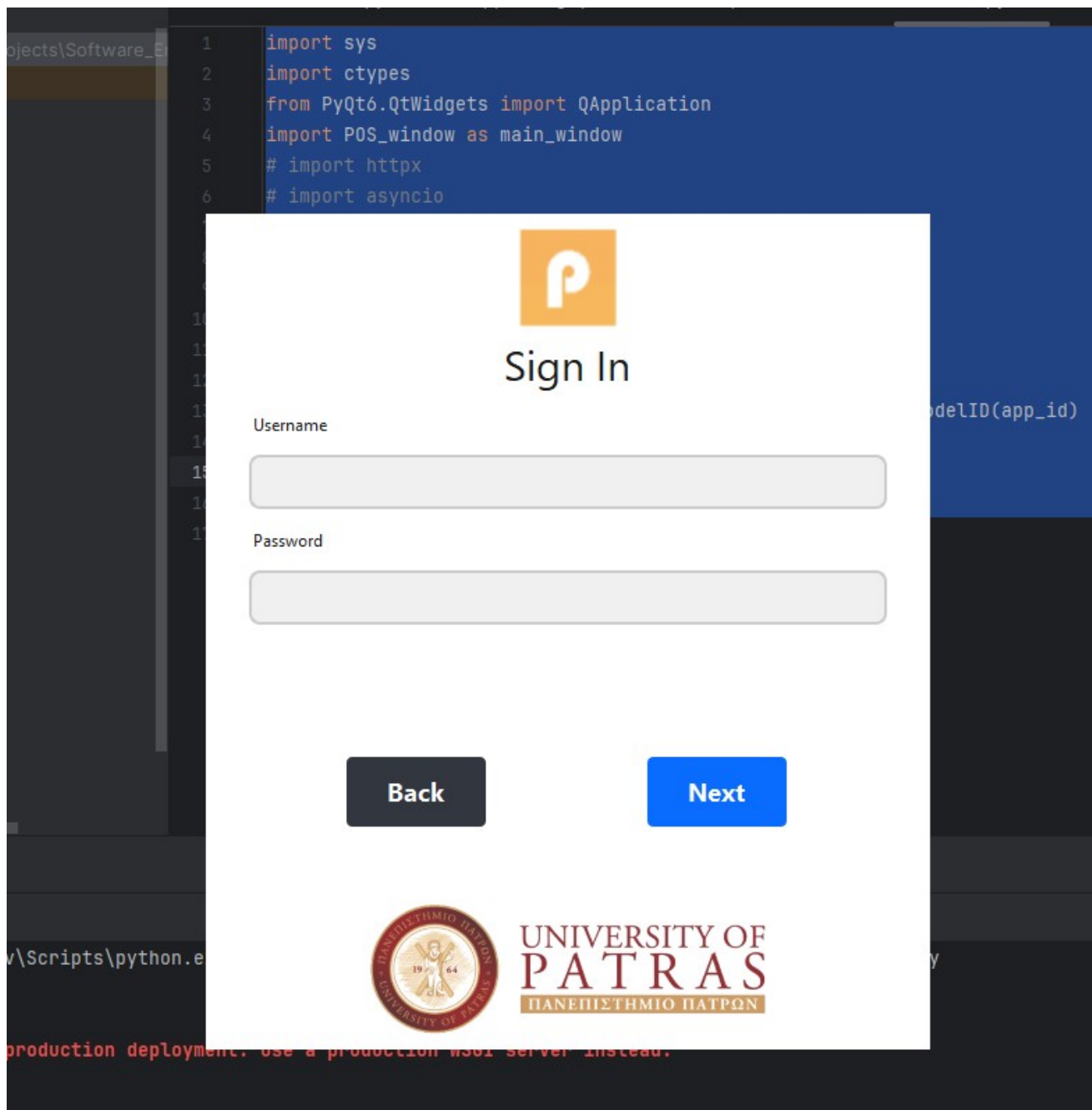
- Server.py, περιλαμβάνει τον server του PayPoint που δέχεται και διαχειρίζεται τα requests από το POS application και επικοινωνεί με το hardware και επιπλέον δίνει την δυνατότητα στον χρήστη να συνδεθεί στο NDA menu.



Εικόνα 1:NDA menu System Tray εικονίδιο

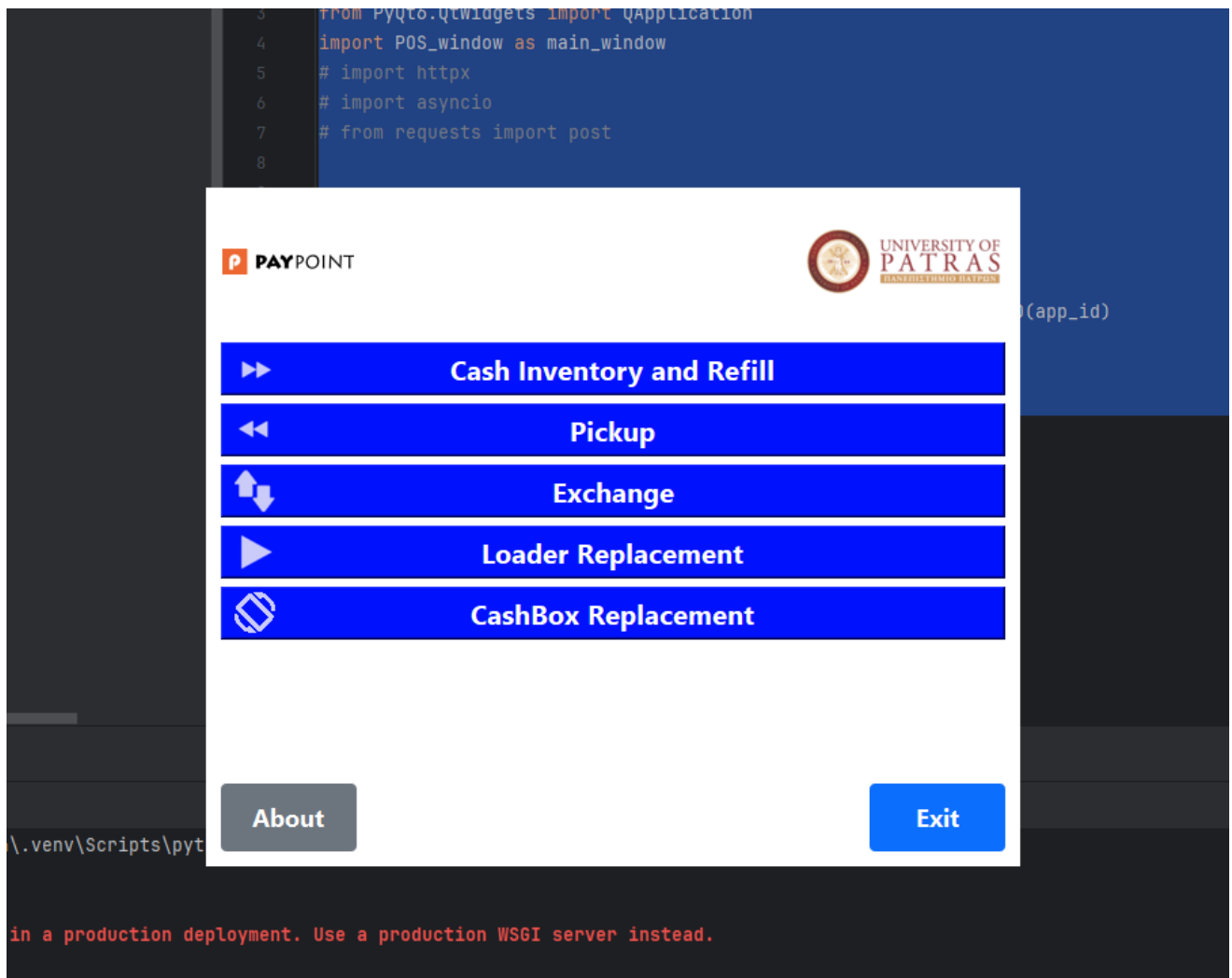


Εικόνα 2: Με δεξί κλικ μπορούμε να ανοίξουμε όλο το menu



Εικόνα 3: Συμπληρώνουμε τα login credentials





Εικόνα 4:NDA menu



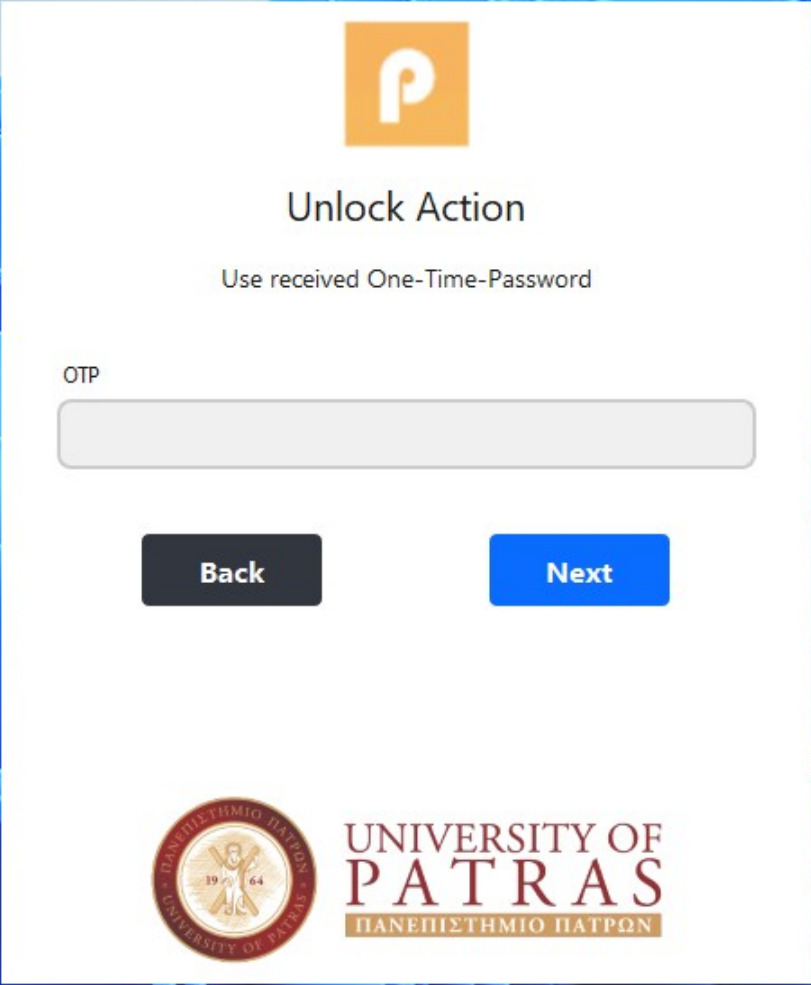
```
class Deposit_Denomination(QEvent):
    def __init__(self, cash_in):
        self.cash_in = cash_in

class Application(QApplication):
    def __init__(self, args):
        super().__init__(args)
        self.cashier_window = None
        self.tray_icon = SystemTrayIcon(QIcon("Images/PayPoint_logo.ico"), self)

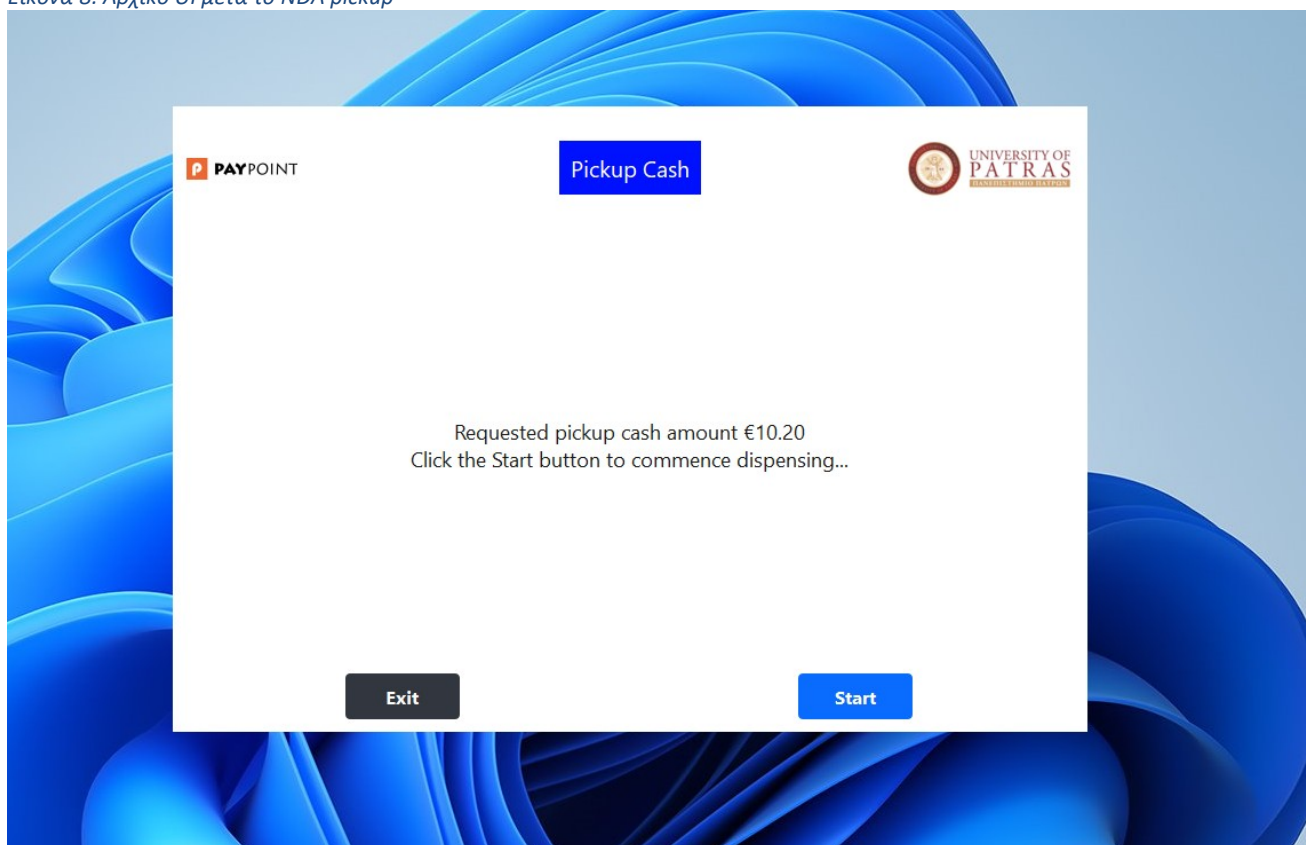
    def event(self, event):
        if event.type() == ShowCashierEvent.EVENT_TYPE:
            self.show_cashier(event.amount)
            return True
        elif event.type() == Deposit_Denomination.EVENT_TYPE:
            if self.cashier_window:
                self.cashier_window.deposit_update_labels(event.cash_in)
            return True
        return super().event(event)
```

C:\Users\dimmi\PycharmProjects\Software\_Engineer\_2024\.venv\Scripts\python.exe C:\Users\dimmi\PycharmProjects\Software\_Engineer\_2024\refill\_input\_denomination.py  
Enter the denomination or 'c' to close: 1  
Received from server: Data received and processed  
Enter the denomination or 'c' to close: 10

Εικόνα 7: Μέσω του script `refill_input_denomination` μπορούμε να κάνουμε `refill` και κάνουμε ουσιαστικά `simulate` την είσοδο του χρήστη



Εικόνα 8: Αρχικό UI μετά το NDA pickup



Εικόνα 9: Εφόσον βρισκόμαστε ακόμα σε *production environment*, δεν έχει ενεργοποιηθεί ακόμα το OTP οπότε πατάμε NEXT και πάμε στο *Pickup Cash* και μπορούμε να ξεκινήσουμε το *Pickup*

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

				denomination	count	min_threshold	float_threshold	max_threshold	full_threshold
<input type="checkbox"/>	Edit	Copy	Delete	1 ¢	52	80	100	730	810
<input type="checkbox"/>	Edit	Copy	Delete	1 €	104	105	50	270	300
<input type="checkbox"/>	Edit	Copy	Delete	10 ¢	55	10	60	495	550
<input type="checkbox"/>	Edit	Copy	Delete	10 €	15	5	30	27	30
<input type="checkbox"/>	Edit	Copy	Delete	100 €	8	0	0	0	0
<input type="checkbox"/>	Edit	Copy	Delete	2 ¢	52	70	100	645	715
<input type="checkbox"/>	Edit	Copy	Delete	2 €	102	105	50	200	225
<input type="checkbox"/>	Edit	Copy	Delete	20 ¢	45	40	60	315	355
<input type="checkbox"/>	Edit	Copy	Delete	20 €	12	4	20	40	60
<input type="checkbox"/>	Edit	Copy	Delete	200 €	1	0	0	0	0
<input type="checkbox"/>	Edit	Copy	Delete	5 ¢	53	50	80	420	475
<input type="checkbox"/>	Edit	Copy	Delete	5 €	21	24	20	50	60
<input type="checkbox"/>	Edit	Copy	Delete	50 ¢	52	100	50	220	245
<input type="checkbox"/>	Edit	Copy	Delete	50 €	11	1	12	12	30
<input type="checkbox"/>	Edit	Copy	Delete	500 €	2	0	0	0	0

☐ Check all | With selected: Edit Copy Delete Export

Εικόνα 10: Simulated inventory table

Software Engineer v0.1

Total amount to pay

Amount 1

Amount 2

Amount 3

Amount 4

Amount 5

Amount 6

Customer

Clear

Pay

1

2

3

4

5

6

7

8

9


-

0

.

C

->



UNIVERSITY OF  
PATRAS  
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

Εικόνα 11: Third party Application, το οποίο εμφανίζεται αφού τρέξουμε το main.py

Software Engineer VU.1

Total amount to pay

1.0

Amount 1

Amount 2

Amount 3

Amount 4

1

Amount 5

Amount 6

Customer

Clear

Pay

1

2

3

6

9

.

->

Total amount: 1.00€

Already Paid: 0.00€

Balance: 1.00€

Waiting for cash deposit...

Stop

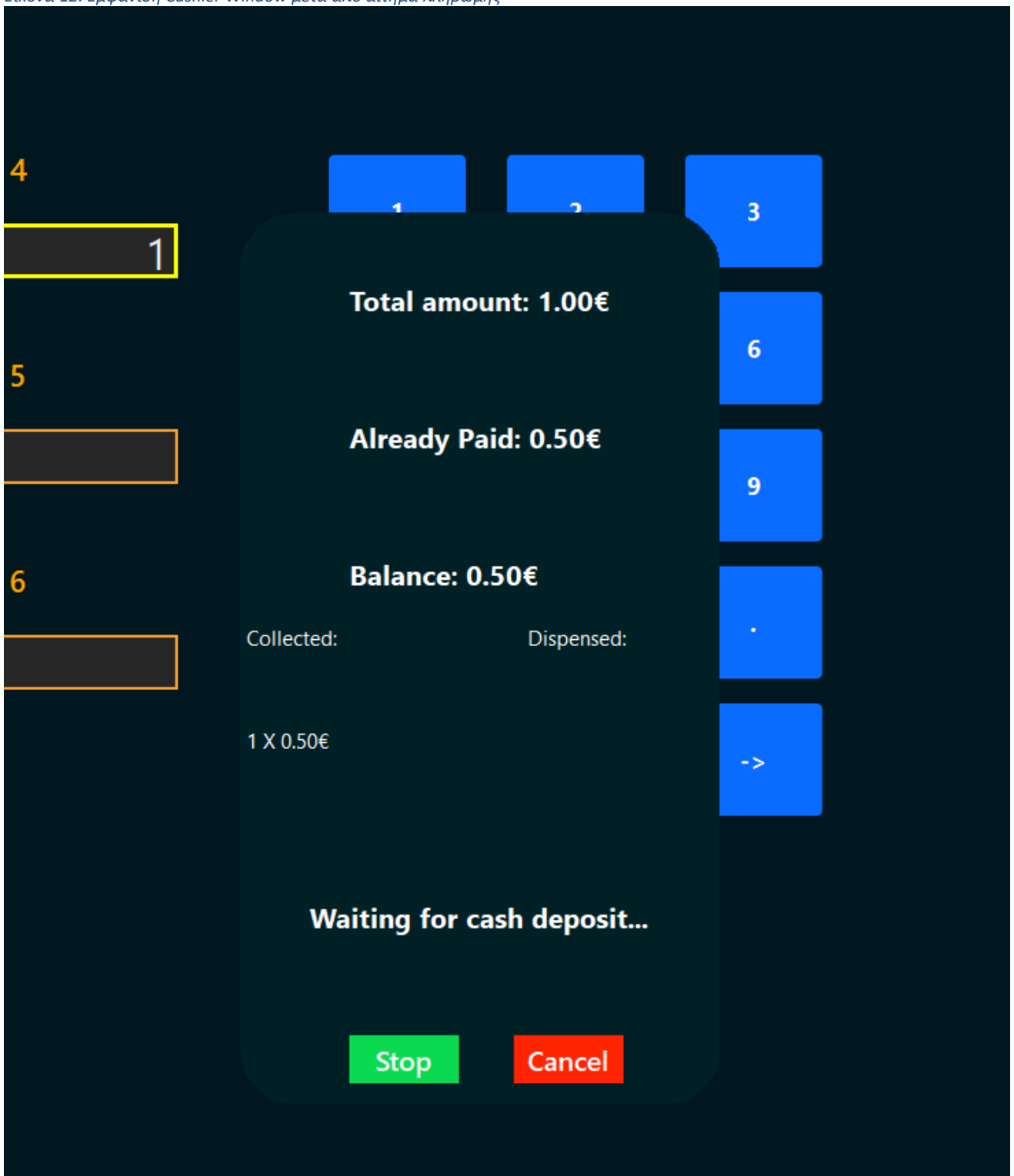
Cancel



UNIVERSITY OF  
PATRAS  
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ



Εικόνα 12: Εμφάνιση Cashier Window μετά από αίτημα πληρωμής



Εικόνα 13: Cashier Window όταν εισέρχονται denomination

The screenshot displays a dark-themed interface for a cashier window. A large, rounded rectangular overlay in the center contains transaction information. To the right of this overlay is a vertical column of blue buttons with white text, serving as a numeric keypad. The transaction details shown are:

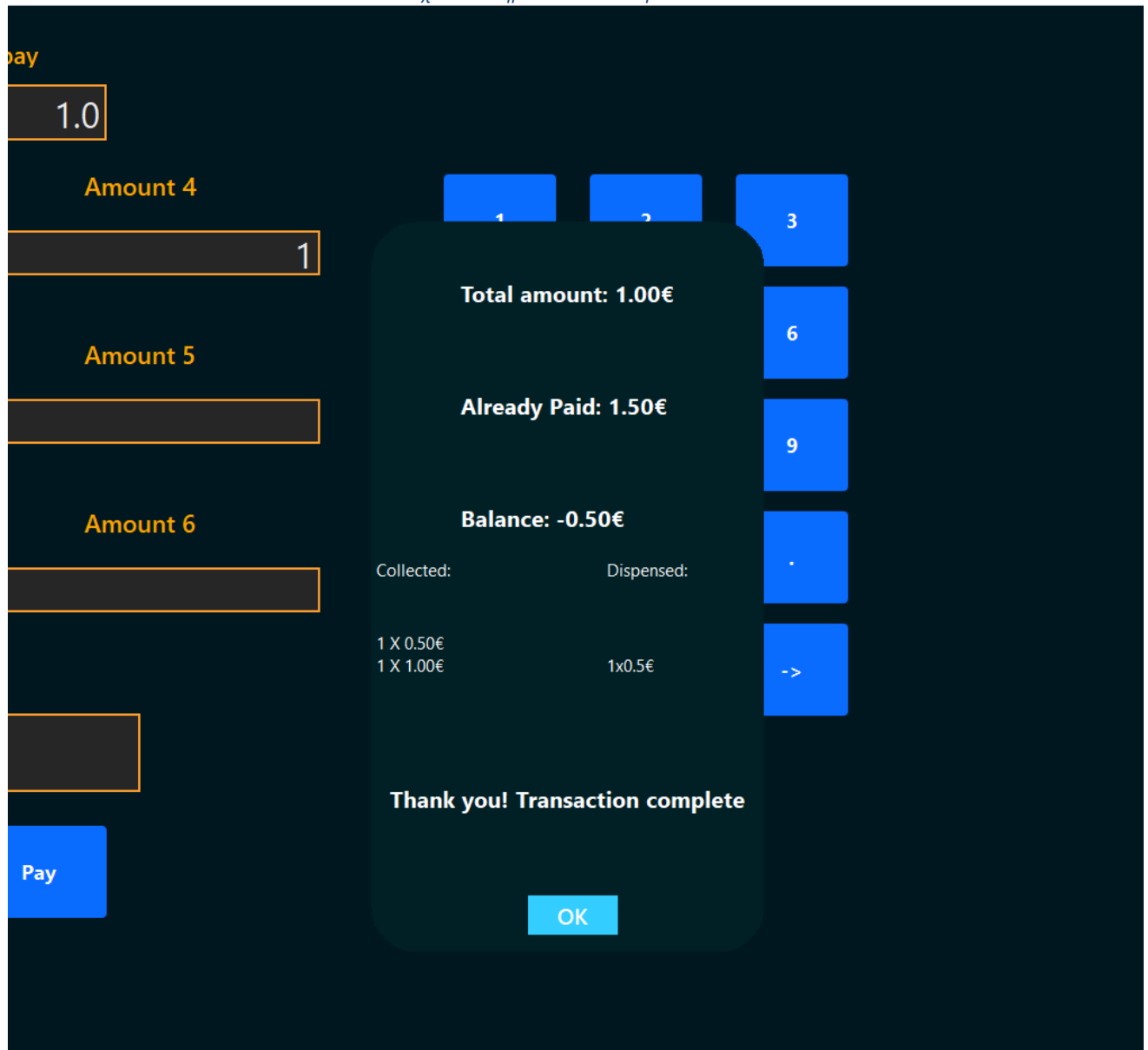
- Total amount: 1.00€**
- Already Paid: 1.50€**
- Balance: -0.50€**

Below these totals, there are two columns of text:

- Collected:**
  - 1 X 0.50€
  - 1 X 1.00€
- Dispensed:**
  - 1x0.5€

At the bottom of the central overlay, the text **Cashback for change** is displayed. The numeric keypad on the right includes buttons for digits 1, 2, 3, 6, 9, a decimal point, and a right arrow (->).

Εικόνα 14: Cashier Window όταν το transaction έχει ολοκληρωθεί και δίνει ρέστα



Εικόνα 15: Τελικό UI που βλέπουμε συνολικά τις πληροφορίες για το transaction

SELECT \* FROM `transactionlog`

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	TransactionId	TransactionType	Timestamp	UserId	TotalInsertedAmount	Description	TotalAmount	DispensedAmount	Status
<input type="checkbox"/> Edit Copy Delete	1	Dispense	2024-05-13 10:50:22	3	5.00	Dispensing change	1.00	4.00	Completed
<input type="checkbox"/> Edit Copy Delete	2	Dispense	2024-05-13 10:53:51	3	5.00	Dispensing change	1.00	4.00	Completed
<input type="checkbox"/> Edit Copy Delete	3	Finalize	2024-05-13 10:54:04	3	5.00	Transaction finalized	1.00	4.00	Completed
<input type="checkbox"/> Edit Copy Delete	4	Stop	2024-05-13 10:55:33	3	0.10	Transaction stopped	1.00	0.00	Stopped
<input type="checkbox"/> Edit Copy Delete	5	Finalize	2024-05-13 10:55:41	3	0.10	Transaction finalized	1.00	0.00	Completed
<input type="checkbox"/> Edit Copy Delete	6	Cancel	2024-05-13 10:55:55	3	0.10	Transaction cancelled	1.00	0.10	Cancelled
<input type="checkbox"/> Edit Copy Delete	7	Finalize	2024-05-13 10:56:02	3	0.10	Transaction finalized	1.00	0.10	Completed
<input type="checkbox"/> Edit Copy Delete	8	Finalize	2024-05-13 10:56:25	3	1.00	Transaction finalized	1.00	0.00	Completed
<input type="checkbox"/> Edit Copy Delete	9	Dispense	2024-05-14 12:04:35	3	1.50	Dispensing change	1.00	0.50	Completed
<input type="checkbox"/> Edit Copy Delete	10	Finalize	2024-05-14 12:04:50	3	1.50	Transaction finalized	1.00	0.50	Completed

Check all | With selected: Edit Copy Delete Export

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Query results operations

Εικόνα 16: Transaction log table όπου αποθηκεύουμε όλες τις πληροφορίες για όλα τα transaction

## Εργαλεία που χρησιμοποιήθηκαν

### *MS Word*

Για την επιμέρους συγγραφή των τεχνικών κειμένων και Pages για την τελική μορφοποίηση τους.

### *Visual Paradigm*

Χρησιμοποιήθηκε για την παραγωγή του Use Case Diagrams