

# PDN System

## Class Diagram

---

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

**Αριθμός Εγγράφου: 012**

### Ιστορικό αναθεώρησης

Revision	Date	Description
A01	28/04/2024	Original Version
A02	30/05/2024	Τελική έκδοση με πρόσθεση περιγραφή των μεθόδων

## 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

## Contents

Ιστορικό αναθεώρησης.....	1
Σύνθεση Ομάδας.....	2
ServerController.....	5
ServerThread.....	5
Denomination_info.....	5
CashInventoryUI.....	6
CashierWindow.....	6
Login_NDA_UI.....	8
NDA_Menu_Button.....	8
NDA_UI.....	8
Window.....	9
PickupUI.....	9
SystemTrayIcon.....	10
ClickableLabel.....	10
Application.....	11
Εργαλεία Που Χρησιμοποιήθηκαν.....	12



## ServerController

Περιγραφή: Κληρονομεί από το QObject και χρησιμεύει ως γέφυρα επικοινωνίας μέσα σε μια εφαρμογή PyQt. Χειρίζεται κυρίως σήματα που σχετίζονται με ενημερώσεις ονομασιών, διευκολύνοντας την επικοινωνία μεταξύ νημάτων μεταξύ ενός νήματος δικτύου και των κύριων στοιχείων του GUI.

### + update\_denomination\_signal() : void

Εκπέμπει ένα σήμα για την ενημέρωση των πληροφοριών της ονομαστικής αξίας.

## ServerThread

Περιγραφή: Μια υποκλάση του QThread που χειρίζεται τις λειτουργίες δικτύου σε ξεχωριστό νήμα για να αποφεύγεται το πάγωμα του GUI. Ακούει για εισερχόμενες συνδέσεις και επεξεργάζεται δεδομένα που λαμβάνονται μέσω μιας υποδοχής δικτύου. Η κλάση είναι υπεύθυνη για τη λήψη δεδομένων συναλλαγών από τους πελάτες, την ενημέρωση του GUI μέσω σημάτων και τη διαχείριση του δικού της κύκλου ζωής για να διασφαλίζεται ο καθαρός τερματισμός.

### + \_\_init\_\_(host, port, controller, parent=None) : void

Αρχικοποιεί το νήμα του διακομιστή με τον κεντρικό υπολογιστή, τη θύρα και τον ελεγκτή.

### + run() : void

Εκτελεί τον διακομιστή για να αποδεχθεί συνδέσεις και να επεξεργαστεί εισερχόμενα δεδομένα.

### + stop() : void

Διακόπτει τον διακομιστή θέτοντας τη σημαία εκτέλεσης σε False και κλείνοντας την υποδοχή.

## Denomination\_info

Περιγραφή: Ένα προσαρμοσμένο QWidget που εμφανίζει πληροφορίες σχετικά με μια συγκεκριμένη ονομαστική αξία νομίσματος. Αυτό το widget χρησιμοποιείται για την παρακολούθηση και την εμφάνιση αλλαγών στην καταμέτρηση των ονομαστικών αξιών νομισμάτων καθώς πραγματοποιούνται συναλλαγές. Περιλαμβάνει λειτουργικότητα για τη δυναμική ενημέρωση της εμφάνισής του με βάση την κατάσταση του συστήματος διαχείρισης νομισμάτων, όπως η εμφάνιση των τρεχουσών μετρήσεων, ο χειρισμός των κατωφλίων και η ένδειξη της κατάστασης μέσω αλλαγών χρώματος.

### + \_\_init\_\_(denomination="", is\_BNR=False, is\_CashBox=0, is\_Loader=0) : void

Αρχικοποιεί το widget πληροφοριών ονομαστικής αξίας.

### + init\_ui\_Denominations() : void

Αρχικοποιεί τα στοιχεία UI για την εμφάνιση των πληροφοριών ονομαστικής αξίας.

### + init\_ui\_CashBox\_Loader() : void

Αρχικοποιεί τα στοιχεία UI για την εμφάνιση των πληροφοριών του CashBox ή του Loader.

### + current\_notes\_coins() : int

Επιστρέφει την τρέχουσα ποσότητα των σημειώσεων ή νομισμάτων.

### **+ add\_cash(count) : void**

Προσθέτει μετρητά στην ονομαστική αξία και ενημερώνει τη βάση δεδομένων.

### **+ update\_status\_color() : void**

Ενημερώνει το χρώμα κατάστασης βάσει της τρέχουσας ποσότητας των σημειώσεων/νομισμάτων.

### **+ refresh\_display() : void**

Ανανεώνει την εμφάνιση με την τελευταία ποσότητα σημειώσεων/νομισμάτων.

## **CashInventoryUI**

Περιγραφή: Αυτό το QWidget λειτουργεί ως η κύρια διεπαφή για τη διαχείριση της απογραφής μετρητών. Ενσωματώνει διάφορα Denomination\_info widgets για την εμφάνιση της κατάστασης των διαφόρων νομισματικών ονομαστικών αξιών. Η κλάση παρέχει λειτουργικότητα για την εκκίνηση και τη διακοπή ενός νήματος διακομιστή για ενημερώσεις σε πραγματικό χρόνο, τον χειρισμό αλληλεπιδράσεων του χρήστη για τη διαχείριση του αποθέματος μετρητών και την ενημέρωση της οθόνης της με βάση σήματα που υποδεικνύουν αλλαγές στο σύστημα διαχείρισης μετρητών.

### **+ \_\_init\_\_() : void**

Αρχικοποιεί το UI αποθέματος μετρητών.

### **+ setupUI() : void**

Ρυθμίζει τα στοιχεία και τη διάταξη του UI.

### **+ center() : void**

Κεντράρει το παράθυρο στην οθόνη.

### **+ return\_to\_main\_menu() : void**

Επιστρέφει στο κύριο μενού.

### **+ update\_total\_cash\_inventory\_from\_display() : void**

Ενημερώνει το συνολικό απόθεμα μετρητών βάσει των εμφανιζόμενων τιμών.

### **+ start\_server() : void**

Ξεκινά το νήμα του διακομιστή.

### **+ stop\_server() : void**

Διακόπτει το νήμα του διακομιστή.

### **+ update\_denomination(denomination, amount) : void**

Ενημερώνει την ονομαστική αξία με το δεδομένο ποσό.

### **+ showEvent(event) : void**

Διαχειρίζεται το συμβάν εμφάνισης και ξεκινά το διακομιστή.

### **+ hideEvent(event) : void**

Διαχειρίζεται το συμβάν απόκρυψης και διακόπτει το διακομιστή.

## **CashierWindow**

Περιγραφή: Η κλάση CashierWindow είναι μια υποκλάση της QWidget που παρέχει μια εξειδικευμένη διεπαφή χρήστη για τη διαχείριση συναλλαγών σε ένα σύστημα ταμείου. Υποστηρίζει λειτουργίες όπως η εκκίνηση μιας συναλλαγής, η αποδοχή εισροών μετρητών, η διανομή ρέστων και η οριστικοποίηση

συναλλαγών. Η κλάση έχει σχεδιαστεί για να αλληλεπιδρά με ένα backend σύστημα για την καταγραφή των λεπτομερειών της συναλλαγής και τη διαχείριση της ροής μετρητών. Χρησιμοποιεί σήματα για το χειρισμό της αλληλεπίδρασης του χρήστη μέσω κουμπιών και την ενημέρωση των ετικετών της οθόνης με βάση την κατάσταση της συναλλαγής.

**+ \_\_init\_\_(total\_amount, mode=2) : void**

Αρχικοποιεί το παράθυρο ταμιά με το συνολικό ποσό και τη λειτουργία.

**+ initUI(mode) : void**

Ρυθμίζει τα στοιχεία και τη διάταξη του UI με βάση τη λειτουργία.

**+ setup\_labels() : void**

Ρυθμίζει τις ετικέτες του παραθύρου ταμιά.

**+ setup\_buttons(mode) : void**

Ρυθμίζει τα κουμπιά του παραθύρου ταμιά με βάση τη λειτουργία.

**+ center\_window() : void**

Κεντράρει το παράθυρο στην οθόνη.

**+ deposit\_update\_labels(cash\_in) : void**

Ενημερώνει τις ετικέτες κατά την κατάθεση μετρητών.

**+ update\_and\_center\_label(label, text) : void**

Ενημερώνει και κεντράρει την ετικέτα με το κείμενο.

**+ start\_transaction() : void**

Ξεκινά μια νέα συναλλαγή.

**+ cancel\_transaction() : void**

Ακυρώνει τη συναλλαγή και ενημερώνει τις ετικέτες.

**+ start\_dispensing(cancel\_operation=0) : void**

Ξεκινά τη διανομή μετρητών ή την ακύρωση της συναλλαγής.

**+ stop\_transaction() : void**

Σταματά τη συναλλαγή και ενημερώνει το αρχείο καταγραφής.

**+ end\_transaction() : void**

Ολοκληρώνει τη συναλλαγή και εμφανίζει μήνυμα ευχαριστίας.

**+ finalize\_transaction() : void**

Ολοκληρώνει και καταγράφει τη συναλλαγή.

**+ reset\_for\_new\_transaction(amount) : void**

Επαναφέρει το παράθυρο για μια νέα συναλλαγή με το καθορισμένο ποσό.

## Login\_NDA\_UI

Περιγραφή: Η κλάση Login\_NDA\_UI είναι μια υποκλάση της QWidget που έχει σχεδιαστεί για να χειρίζεται τον έλεγχο ταυτότητας χρήστη για ένα σύστημα διαχείρισης συμφωνιών μη αποκάλυψης (NDA) μέσα σε μια εφαρμογή. Παρέχει μια γραφική διεπαφή χρήστη (GUI) για τους χρήστες να εισάγουν τα διαπιστευτήριά τους (όνομα χρήστη και κωδικό πρόσβασης) και επικυρώνει αυτά τα διαπιστευτήρια σε σχέση με μια βάση δεδομένων. Η κλάση είναι δομημένη ώστε να διευκολύνει την ασφαλή πρόσβαση, με χαρακτηριστικά για το χειρισμό σφαλμάτων και την ανατροφοδότηση του χρήστη σχετικά με τη διαδικασία σύνδεσης.

### + \_\_init\_\_() : void

Αρχικοποιεί το παράθυρο σύνδεσης NDA με όλες τις απαραίτητες ρυθμίσεις και στοιχεία διεπαφής χρήστη.

### + setupUI() : void

Ρυθμίζει τα στοιχεία και τη διάταξη του UI, περιλαμβάνοντας ετικέτες, πεδία εισαγωγής και κουμπιά.

### + create\_connection() :

#### mysql.connector.connection\_cext.CMySQLConnection

Δημιουργεί σύνδεση με τη βάση δεδομένων χρησιμοποιώντας τις ρυθμίσεις από το αρχείο JSON.

### + validateCredentials() : void

Επεξεργάζεται την επικύρωση των διαπιστευτηρίων χρήστη, ελέγχοντας το όνομα χρήστη και τον κωδικό πρόσβασης στη βάση δεδομένων.

### + closeEvent(event) : void

Κλείνει το παράθυρο και κρύβει το UI κατά το συμβάν κλεισίματος.

## NDA\_Menu\_Button

Περιγραφή: Η κλάση NDA\_Menu\_Button είναι μια υποκλάση της QPushButton ειδικά σχεδιασμένη για να βελτιώνει την οπτική παρουσίαση των κουμπιών στο σύστημα διαχείρισης NDA. Αυτή η κλάση κουμπιών υποστηρίζει προσαρμοσμένες εικόνες (pixmap) παράλληλα με το κείμενο των κουμπιών, ενισχύοντας τη χρηστικότητα και την αισθητική της διεπαφής.

### + \_\_init\_\_(parent=None) : void

Αρχικοποιεί το κουμπί του μενού NDA με το καθορισμένο γονικό στοιχείο.

### + sizeHint() : QSize

Επιστρέφει το προτεινόμενο μέγεθος του κουμπιού με βάση το pixmap, αν υπάρχει.

### + setPixmap(pixmap) : void

Ορίζει το pixmap για το κουμπί και ανανεώνει το κουμπί για να εμφανίσει το pixmap.

### + paintEvent(event) : void

Ζωγραφίζει το pixmap στο κουμπί κατά τη διάρκεια του γεγονότος ζωγραφικής.

## NDA\_UI

Περιγραφή: Η κλάση NDA\_UI επεκτείνει την QWidget και χρησιμεύει ως η κύρια διεπαφή για λειτουργίες που σχετίζονται με το NDA εντός μιας εφαρμογής. Αυτή η κλάση παρέχει μια δομημένη διάταξη που



περιέχει πολλαπλά λειτουργικά κουμπιά που οδηγούν σε διαφορετικά μέρη της εφαρμογής, όπως η διαχείριση του αποθέματος μετρητών και της αναπλήρωσης, οι λειτουργίες παραλαβής και άλλα.

### **+ \_\_init\_\_() : void**

Αρχικοποιεί το παράθυρο του NDA UI με όλες τις απαραίτητες ρυθμίσεις και στοιχεία διεπαφής χρήστη.

### **+ setupUI() : void**

Ρυθμίζει τα στοιχεία και τη διάταξη του UI, περιλαμβάνοντας ετικέτες, κουμπιά και εικόνες.

### **+ showCashInventory() : void**

Δημιουργεί και εμφανίζει το CashInventoryUI, κρύβοντας το NDA UI.

### **+ showPickup() : void**

Δημιουργεί και εμφανίζει το PickupUI, κρύβοντας το NDA UI.

## **Window**

Περιγραφή: Η κλάση Window επεκτείνει την QMainWindow για τη δημιουργία ενός κύριου παραθύρου εφαρμογής για μια διεπαφή επεξεργασίας πληρωμών. Αυτή η κλάση είναι υπεύθυνη για τη διαχείριση και την εμφάνιση διαφόρων στοιχείων διεπαφής χρήστη, όπως ετικέτες, κουμπιά και πεδία εισαγωγής για ένα προσομοιωμένο τερματικό πληρωμής.

### **+ \_\_init\_\_(parent=None) : void**

Αρχικοποιεί το κύριο παράθυρο της εφαρμογής με όλες τις απαραίτητες ρυθμίσεις και στοιχεία διεπαφής χρήστη.

### **+ setupUI() : void**

Ρυθμίζει τα στοιχεία και τη διάταξη του UI, περιλαμβάνοντας εικόνες, κουμπιά και πεδία εισαγωγής.

### **+ calculate\_total\_amount(sub\_amount, total\_amount) : void**

Υπολογίζει το συνολικό ποσό από τα δεδομένα εισόδου και ενημερώνει το πεδίο εισαγωγής του συνολικού ποσού.

### **+ clear\_text() : void**

Καθαρίζει το κείμενο από όλα τα πεδία εισαγωγής όταν πατηθεί το κουμπί εκκαθάρισης.

### **+ setFocusText(button\_value) : void**

Προσθέτει την τιμή του κουμπιού στο τελευταίο πεδίο εισαγωγής που είχε την εστίαση.

### **+ on\_focusChanged(widget) : void**

Ενημερώνει την εστίαση στο τελευταίο πεδίο εισαγωγής που είχε την εστίαση.

### **+ send\_POST\_request() : void**

Στέλνει αίτημα POST με το ποσό πληρωμής στη δεδομένη διεύθυνση URL και χειρίζεται την απάντηση.

## **PickupUI**

Περιγραφή: Η κλάση PickupUI επεκτείνει την QWidget και έχει σχεδιαστεί για να διαχειρίζεται τη διεπαφή χρήστη για μια ενέργεια παραλαβής σε ένα σύστημα διαχείρισης μετρητών ή πληρωμών. Αυτή η διεπαφή περιλαμβάνει λειτουργίες για την εισαγωγή ενός κωδικού πρόσβασης μίας χρήσης (OTP), την έναρξη της διαδικασίας έκδοσης μετρητών και την εμφάνιση της κατάστασης της συναλλαγής.

### **+ \_\_init\_\_(parent=None) : void**

Αρχικοποιεί το παράθυρο του Pickup UI με όλες τις απαραίτητες ρυθμίσεις και στοιχεία διεπαφής χρήστη.

### **+ setupUI() : void**

Ρυθμίζει τα στοιχεία και τη διάταξη του UI, περιλαμβάνοντας ετικέτες, πεδία εισαγωγής και κουμπιά.

### **+ transition\_to\_state\_waiting\_dispense() : void**

Μεταβαίνει στην κατάσταση αναμονής διανομής μετρητών, κρύβοντας το τρέχον περιεχόμενο και αναδιαμορφώνοντας τη διάταξη.

### **+ rebuild\_waiting\_dispense\_layout() : void**

Αναδημιουργεί τη διάταξη για την κατάσταση αναμονής διανομής μετρητών.

### **+ final\_rebuild\_waiting\_dispense\_layout() : void**

Αναδημιουργεί τη διάταξη για την τελική κατάσταση διανομής μετρητών και ενημερώνει τη βάση δεδομένων με τα διανεμόμενα ποσά.

### **+ transition\_to\_state\_init\_dispense() : void**

Μεταβαίνει στην κατάσταση διανομής μετρητών μετά την πίεση του κουμπιού έναρξης.

### **+ hide\_layout\_contents(layout) : void**

Κρύβει όλα τα στοιχεία και υπο-διατάξεις εντός μιας δεδομένης διάταξης και αφαιρεί τους διαχωριστές.

### **+ resetUI() : void**

Επαναφέρει τα στοιχεία UI στις αρχικές τους καταστάσεις για εισαγωγή OTP.

### **+ onBackClicked() : void**

Επαναφέρει το UI στην αρχική του κατάσταση πριν την απόκρυψη και εκπέμπει το σήμα όταν πατηθεί το κουμπί πίσω.

## **SystemTrayIcon**

Περιγραφή: Η κλάση SystemTrayIcon επεκτείνει την QSystemTrayIcon και έχει σχεδιαστεί για να παρέχει μια διεπαφή εικονιδίου στο δίσκο συστήματος για μια εφαρμογή λογισμικού, ειδικά για ένα σύστημα που έχει ρυθμιστεί για να υποστηρίξει τη λειτουργία διαχείρισης NDA (Non-Disclosure Agreement).

### **+ \_\_init\_\_(icon, parent=None) : void**

Αρχικοποιεί ένα νέο SystemTrayIcon με το δεδομένο εικονίδιο και το γονικό στοιχείο.

### **+ setupUI() : void**

Ρυθμίζει το UI του εικονιδίου συστήματος, προσθέτοντας το μενού και τις ενέργειες.

### **+ NDA\_mode() : void**

Εμφανίζει το παράθυρο σύνδεσης του Admin όταν γίνεται κλικ στο NDA label.

## **ClickableLabel**

Η κλάση ClickableLabel είναι μια προσαρμοσμένη κλάση QLabel που επιτρέπει την ανίχνευση κλικ από τον χρήστη. Περιέχει ένα σήμα clicked, το οποίο εκπέμπεται όταν το label πιέζεται. Αυτή η λειτουργικότητα μπορεί να χρησιμοποιηθεί για να προσθέσει διαδραστικότητα σε ένα QLabel, καθιστώντας το χρήσιμο για περιπτώσεις όπου απαιτείται η χρήση ενός κειμένου ή εικόνας ως κουμπί.

### **+ \_\_init\_\_(parent=None) : void**

Αρχικοποιεί ένα νέο ClickableLabel με το δεδομένο γονικό στοιχείο.

### **+ mousePressEvent(event) : void**

Εκπέμπει το σήμα clicked όταν το label πιέζεται.

## Application

Περιγραφή: Η κλάση Application επεκτείνει την QApplication για τη διαχείριση μιας εφαρμογής GUI που ενσωματώνεται με έναν διακομιστή Flask. Χειρίζεται την επικοινωνία μεταξύ διεργασιών μεταξύ ενός διακομιστή Flask και της διεπαφής PyQt, συγκεκριμένα τη διαχείριση συναλλαγών μετρητών μέσω ενός παραθύρου ταμείου.

## **Εργαλεία Που Χρησιμοποιήθηκαν**

### *MS Word*

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

### *MS Excel*

Για την κατασκευή των Gantt charts.

### *MS Visio*

Για την κατασκευή των Pert Chart, των class και των robustness diagrams