Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Προηγμένα Συστήματα Πληροφορικής»

Κατεύθυνση

«Προηγμένες Τεχνολογίες Ανάπτυξης Λογισμικού»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Διαδικτυακή Εφαρμογή Διασύνδεσης με Τράπεζα με την χρήση API** **Web Application to connect to a bank with the use of an API**
Ονοματεπώνυμο Φοιτητή	Βαρβατές Γεώργιος
Πατρώνυμο	Κομνηνός
Αριθμός Μητρώου	ΜΠΣΠ 14008
Επιβλέπων	Δουληγέρης Χρήστος, Καθηγητής
Ημερομηνία Παράδοσης	Οκτώβριος 2018

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Χρήστος Δουληγέρης Καθηγητής	Δημήτρης Βέργαδος Αναπληρωτής Καθηγητής	Παναγιώτης Κοτζανικολάου Επίκουρος Καθηγητής

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή μου κο Δουληγέρη Χρήστο που με καθοδήγησε και με εμπιστεύτηκε για την υλοποίηση αυτής της μεταπτυχιακής εργασίας.

Ευχαριστώ επίσης την οικογένεια μου για την ψυχολογική και οικονομική βοήθεια που μου παρείχε καθ' όλη τη διάρκεια των σπουδών μου. Τέλος θα ήθελα να ευχαριστήσω την Περιστέρα Κουτρούλη για την υποστήριξη που μου έδειξε κατά την διάρκεια της συγγραφής αυτής της εργασίας.

Πίνακας Περιεχομένων

Περίληψη

Η παρούσα μεταπτυχιακή διατριβή ασχολείται με την ανάλυση, τον σχεδιασμό και την υλοποίηση μιας εφαρμογής η οποία θα εκμεταλλεύεται την πλατφόρμα rAPIdLINK για να συνδέεται με την Τράπεζα Πειραιώς. Ο χρήστης αυτής της εφαρμογής θα μπορεί με την χρήση των κωδικών που έχει για το e-banking της Τράπεζας Πειραιώς, να συνδεθεί και να έχει πρόσβαση σε λογαριασμούς, κινήσεις, κάρτες καθώς επίσης και σε λεπτομέρειες που αφορούν τα σημεία εξυπηρέτησης της τράπεζας. Επίσης του επιτρέπει να πραγματοποιήσει μεταφορές από δικούς του λογαριασμούς, σε λογαριασμούς άλλων τραπεζών.

Παράλληλα γίνεται παρουσίαση της εξέλιξης του τραπεζικού συστήματος παγκοσμίως. Τέλος όσων αφορά την Ελλάδα, παρουσιάζονται οι υπηρεσίες που προσφέρουν οι συστημικές τράπεζες.

Abstract

This master thesis deals with the analysis, the design and the development of an application that makes use of the rAPIdLINK platform in order to connect with Piraeus Bank. The user of this web application can use his credentials for Piraeus Bank e-banking in order to connect and to have access to his accounts, transactions, cards as long as to details that have to do with the service points of the bank. Furthermore, he can transfer money from his accounts to other accounts no matter the bank.

In the meantime we present the evolution of the banking system worldwide. Finally as far as Greece is concerned , we present all the services that each one of the systemic banks offers.

Πίνακας Εικόνων

Κεφάλαιο 1

1.1 Σκοπός και στόχοι της εργασίας

Στον 21ο αιώνα μπορούμε να πούμε με βεβαιότητα ότι έχουμε ξεφύγει από τις παραδοσιακές μεθόδους επικοινωνίας με τις τράπεζες. Με την είσοδο του διαδικτύου και των κινητών συσκευών η χρήση online banking γίνεται όλο και πιο συχνή. Παράλληλα και οι ίδιες οι τράπεζες βλέποντας αυτή την στροφή των πελατών τους , τους προσφέρουν ολοένα και περισσότερες ηλεκτρονικές υπηρεσίες.

Στόχοι της εργασίας αυτής είναι οι εξής:

- να μελετήσει τις online υπηρεσίες που παρέχουν οι συστημικές τράπεζες στην Ελλάδα.
- η υλοποίηση μιας διαδικτυακής εφαρμογής η οποία θα επικοινωνεί μέσω API με μία τράπεζα και πιο συγκεκριμένα με την τράπεζα Πειραιώς.

Η εργασία χωρίζεται στα παρακάτω κεφάλαια:

- Κεφάλαιο 1: Εισαγωγή και σύντομα ιστορικά στοιχεία. Στο κεφάλαιο αυτό περιγράφεται συνοπτικά η εξέλιξη της τράπεζας από την αρχαιότητα μέχρι σήμερα, καθώς επίσης και η εξέλιξη της ηλεκτρονικής τραπεζικής μέσα στα χρόνια.
- Κεφάλαιο 2: Σε αυτό το κεφάλαιο αναλύεται το online banking. Παρουσιάζουμε τα διάφορα είδη online banking, τον τρόπο λειτουργίας του και τις δυνατότητες που παρέχει στον χρήστη. Τέλος δίνουμε αναλυτικά τα μειονεκτήματα και τα πλεονεκτήματα του online banking τόσο για τον πελάτη όσο και για την τράπεζα.
- Κεφάλαιο 3: Παρουσίαση των συστημάτων ηλεκτρονικής τραπεζικής, των 4 συστημικών τραπεζών (Alpha Bank, Εθνική Τράπεζα, Eurobank, Τράπεζα Πειραιώς). Αναλύουμε όλες τις δυνατότητες, ευκολίες και όλα τα εργαλεία που προσφέρουν μέσω αυτών των συστημάτων οι τράπεζες στους πελάτες τους.
- Κεφάλαιο 4: Παρουσίαση της πλατφόρμας rAPIdlink, η οποία παρέχει την επικοινωνία του πελάτη με την τράπεζα Πειραιώς μέσω API καθώς και των εργαλείων που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής.
- Κεφάλαιο 5: Παρουσίαση εφαρμογής που διασυνδέει τον πελάτη με την Τράπεζα Πειραιώς μέσω API.

1.2 Από την αρχαιότητα στον 20ο αιώνα. Η εξέλιξη της Τράπεζας.

Η ιστορία της τραπεζικής ξεκινά από την αρχαιότητα, γύρω στο 2000 π.Χ. στην Ασσυρία και την Σουμερία, με εμπόρους που έδιναν δάνεια από σιτηρά στους αγρότες και τους πωλητές που μετέφεραν προϊόντα μεταξύ των πόλεων να αποτελούν ουσιαστικά τις πρώτες τράπεζες.

Παράλληλα όμως είναι αλληλένδετη και με την ιστορία του χρήματος. Οι τράπεζες, με διάφορες μορφές υπήρχαν από την πρώτη στιγμή που κόπηκαν τα πρώτα νομίσματα. Οι αυτοκρατορίες έψαχναν ένα τρόπο να φορολογούν τον λαό και η μέθοδος του ενός γουρουνιού τον χρόνο, σύμφωνα με την οποία οι πολίτες ήταν αναγκασμένοι να δώσουν ένα υγιές γουρούνι στο κράτος ως φόρο, σταμάτησε να είναι βολική από ένα σημείο κι έπειτα. Επίσης, οι αυτοκρατορίες ήθελαν να πληρώνουν για τα διάφορα αγαθά και υπηρεσίες με κάτι που μπορεί να ανταλλαχθεί πιο άμεσα και το οποίο έχει μια αντικειμενική αξία. Έτσι δημιουργήθηκαν τα πρώτα νομίσματα. Αυτά τα νομίσματα όμως έπρεπε κάπου να φυλαχθούν. Τα πρώτα μέρη φύλαξης νομισμάτων ήταν οι ναοί για δύο λόγους. Πρώτον, δεν μπορούσε ο καθένας να έχει ασφαλές χρηματοκιβώτιο και δεύτερον γιατί οι ναοί θεωρούνταν τα πιο ασφαλή μέρη της πόλης. Υπάρχουν καταγεγραμμένα αρχεία από την Ελλάδα, την Ρώμη, την Αίγυπτο και την Αρχαία Βαβυλωνία που αναφέρουν πως οι ναοί δάνειζαν λεφτά με αντάλλαγμα να τα κρατήσουν ασφαλή. Δεν είναι τυχαίο άλλωστε πως σε περίοδο πολέμων οι ναοί ήταν τα πρώτα μέρη που λεηλατούσαν οι αντίπαλοι.

Η πρώτη τράπεζα με τη μορφή που την ξέρουμε σήμερα, δηλαδή ξεχωριστό κτήριο αποσπασμένο από άλλες ιδιότητες και με αποκλειστικό σκοπό την πραγματοποίηση χρηματικών συναλλαγών , δημιουργήθηκε από τους Ρωμαίους. Ο Ιούλιος Καίσαρας σε ένα από τα περίφημα διατάγματά του που άλλαξαν το Ρωμαϊκό Δίκαιο, μας δίνει το πρώτο παράδειγμα κατάσχεσης γης από την τράπεζα λόγω της αδυναμίας αποπληρωμής δανείου. Η Ρωμαϊκή Αυτοκρατορία έπεσε, αλλά το τραπεζικό σύστημα διασώθηκε κάνοντας τους ηγεμόνες της Ευρώπης να συνειδητοποιήσουν την δύναμη που θα τους χάριζαν τα τραπεζικά ιδρύματα.

Τους επόμενους αιώνες το τραπεζικό σύστημα συνέχισε να εξελίσσεται και τον 18ο αιώνα στην Αγγλία οι αποδείξεις κατάθεσης για σταθερά ποσά αποτέλεσαν τα πρώτα τραπεζογραμμάτια. Έτσι σιγά - σιγά με την ανταλλαγή των τραπεζογραμματίων, οι τράπεζες άρχισαν να αποκτούν τον ρόλο που κατέχουν σήμερα στην σύγχρονη

οικονομία. Ειδικότερα στην περίοδο της Αναγέννησης, εμφανίστηκαν οι πρώτες οργανωμένες τράπεζες(στην Ιταλία και στη Φλάνδρα), καλύπτοντας ένα ευρύτερο φάσμα εργασιών, προετοιμάζοντας τη σύγχρονη έννοια της τράπεζας ως πιστωτικό ίδρυμα που απευθύνεται (και εξυπηρετεί) καταθέτες και επιχειρήσεις. [1][2]

1.3 20ος αιώνας και Διαδίκτυο

Έτσι φτάνουμε στον 20ο αιώνα που πραγματοποιήθηκε η μεγαλύτερη επανάσταση στο σύγχρονο τραπεζικό σύστημα με την άφιξη του Διαδικτύου και των κινητών συσκευών. Στα τέλη της δεκαετίας του 80, με τον όρο Online Banking (επιγραμμική τραπεζική) αναφερόμασταν στην χρήση ενός τερματικού, ενός πληκτρολογίου και μιας οθόνης με σκοπό να έχουμε πρόσβαση σε κάποιον τραπεζικό λογαριασμό χρησιμοποιώντας τηλεφωνική γραμμή. Τώρα ο ίδιος όρος περιλαμβάνει κάθε ηλεκτρονικό σύστημα πληρωμών που επιτρέπει στους πελάτες ή στο πιστωτικό ίδρυμα να πραγματοποιήσει οικονομικές συναλλαγές μέσω μιας εφαρμογής. Μερικές αξιοσημείωτες περιπτώσεις της εξέλιξης του online banking είναι οι παρακάτω[3]:

- 1981: New York City Banks Test At-Home Banking .Η Νέα Υόρκη είναι η πρώτη πόλη που παρέχει απομακρυσμένες υπηρεσίες, αφού 4 από τις τράπεζές της, επιτρέπουν στους πελάτες του να έχουν πρόσβαση στα τραπεζικά στοιχεία τους από τα σπίτια τους.
- 1983: Η πρώτη online banking υπηρεσία από τράπεζα της Ευρώπης και συγκεκριμένα από την τράπεζα της Σκωτίας. Το σύστημα βασιζόταν στο Βρετανικό Prestel viewlink system (εικόνα 1) και χρησιμοποιούσε έναν υπολογιστή ή ένα πληκτρολόγιο συνδεδεμένο με το τηλεφωνικό σύστημα και μια τηλεόραση.

Εικόνα 1- Prestel Viewlink System

- 1994: Το 1994 ξεκίνησαν πιλοτικά οι online banking υπηρεσίες και στη Γαλλία. Το 1998 ξεκίνησε και επίσημα αυτές οι υπηρεσίες με την χρήση

Minitel (εικόνα 2) τερματικών συστημάτων, τα οποία μοιράστηκαν στους πολίτες δωρεάν

Εικόνα 2- Minitel

- 1996: Ιδρύεται η NetBank: Η πιο επιτυχημένη μέχρι σήμερα τράπεζα που παρείχε τις υπηρεσίες μόνο ηλεκτρονικά. Έκλεισε το 2007.
- 1999: Ιδρύεται η Bank of Internet USA. Αυτή είναι η μεγαλύτερη, ηλικιακά, ηλεκτρονική τράπεζα της Αμερικής. Λειτούργησε το 2000 και έκανε φανερό στους πελάτης της την χρησιμότητα και τα προτερήματα της ηλεκτρονικής τραπεζικής.

Κεφάλαιο 2

2.1 Τι είναι το online banking

Με τον όρο online banking ή αλλιώς internet banking εννοούμε ένα σύνολο από ηλεκτρονικές υπηρεσίες που επιτρέπουν στους πελάτες της τράπεζας ή άλλων χρηματοπιστωτικών ιδρυμάτων να εκτελέσουν ένα πλήθος από οικονομικές συναλλαγές μέσω της ιστοσελίδας του αντίστοιχου ιδρύματος ή τράπεζας.[4] Το online banking σύστημα αποτελεί κομμάτι του Core Banking συστήματος που εκτελείται από μια τράπεζα κι έρχεται σε αντίθεση με Branch Banking σύστημα το οποίο ήταν ο παραδοσιακός τρόπος με τον οποίο είχαν πρόσβαση οι πελάτες στις υπηρεσίες της τράπεζας. Με τον όρο Core Banking (Centralized Online Real-time Exchange) είναι μια τραπεζική υπηρεσία που παρέχεται από μια ομάδα δικτυακών τραπεζικών καταστημάτων όπου οι πελάτες μπορούν να έχουν πρόσβαση στον τραπεζικό τους λογαριασμό και να εκτελούν βασικές συναλλαγές από οποιοδήποτε από τα υποκαταστήματα μελών.[6]

Σήμερα υπάρχουν τράπεζες οι οποίες έχουν παρουσία και υπόσταση μόνο στο διαδίκτυο. Σύμφωνα με το toptenreviews.com [7] οι πιο γνωστές τράπεζες που έχουν μόνο διαδικτυακή υπόσταση παρουσιάζονται στον παρακάτω πίνακα

PRODUCT	PRICE	OVERALL RATING	INTEREST RATES	SERVICES	SECURITY & SUPPORT	BASIC CHECKING	PREMIUM CHECKING
Ally Bank	Check Price	9.3	9.5	8.8	10	0.10%	0.60%
First Internet Bank of Indiana	Check Price	8.1	7.3	9.5	8	0.00%	0.55%
Discover Bank	Check Price	8	8.3	7.3	8.8	$0.10 ON UP TO 100 TRANSACTIONS PER MONTH	$0.10 ON UP TO 100 TRANSACTIONS PER MONTH
Bank of Internet USA	Check Price	7.9	8.3	6.5	10	0.00%	1.25%
Bank5 Connect	Check Price	7.7	10	4.3	8	0.76%	0.76%

Εικόνα 3- Top 10 only online banks

2.2 Internet και διάδοση online banking

Αδιαμφησβήτητα σημαντικός σταθμός για την δημιουργία του online banking, και γενικότερα πολλών online υπηρεσιών που χρησιμοποιούμε στις μέρες μας, ήταν η δημιουργία του Internet.

Το σημερινό Internet αποτελεί εξέλιξη του ARPANET, ενός δικτύου που άρχισε να αναπτύσσεται πειραματικά στα τέλη της δεκαετίας του 60 στις ΗΠΑ. Στα πανεπιστήμια των ΗΠΑ οι ερευνητές ξεκινούν να πειραματίζονται με τη διασύνδεση απομακρυσμένων υπολογιστών μεταξύ τους. Το δίκτυο ARPANET γεννιέται το 1969 με πόρους του προγράμματος ARPA (Advanced Research Project Agency) του Υπουργείου Άμυνας, με σκοπό να συνδέσει το Υπουργείο με στρατιωτικούς ερευνητικούς οργανισμούς και να αποτελέσει ένα πείραμα για τη μελέτη της αξιόπιστης λειτουργίας των δικτύων. Στην αρχική του μορφή, το πρόγραμμα απέβλεπε στον πειραματισμό με μια νέα τεχνολογία γνωστή σαν μεταγωγή πακέτων (packet switching), σύμφωνα με την οποία τα προς μετάδοση δεδομένα κόβονται σε πακέτα και πολλοί χρήστες μπορούν να μοιραστούν την ίδια επικοινωνιακή γραμμή.

Στόχος ήταν η δημιουργία ενός διαδικτύου που θα εξασφάλιζε την επικοινωνία μεταξύ απομακρυσμένων δικτύων, έστω και αν κάποια από τα ενδιάμεσα συστήματα βρίσκονταν προσωρινά εκτός λειτουργίας. Κάθε πακέτο θα είχε την πληροφορία που χρειάζονταν για να φτάσει στον προορισμό του, όπου και θα γινόταν η επανασύνθεσή του σε δεδομένα τα οποία μπορούσε να χρησιμοποιήσει ο τελικός χρήστης.

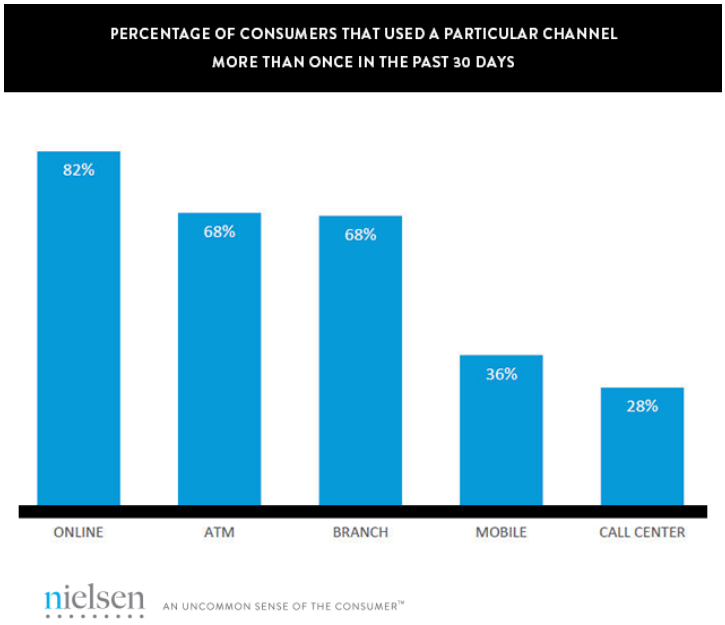
Το παραπάνω σύστημα θα επέτρεπε σε υπολογιστές να μοιράζονται δεδομένα και σε ερευνητές να υλοποιήσουν το ηλεκτρονικό ταχυδρομείο. Φυσικά η ύπαρξη του ARPANET από μόνο του δεν θα καθιστούσε δυνατή την ύπαρξη των άλλων υπηρεσιών. Το 1983, το πρωτόκολλο TCP/IP (δηλ. ο συνδυασμός των TCP και IP) αναγνωρίζεται ως πρότυπο από το Υπουργείο Άμυνας των ΗΠΑ. Παράλληλα το 1985 το National Science Foundation (NSF) δημιουργεί ένα δικό του γρήγορο δίκτυο, το NSFNET χρησιμοποιώντας το πρωτόκολλο TCP/IP, προκειμένου να συνδέσει πέντε κέντρα υπερ-υπολογιστών μεταξύ τους και με την υπόλοιπη επιστημονική κοινότητα. Χιλιάδες πανεπιστήμια και οργανισμοί δημιουργούν τα δικά τους δίκτυα και τα συνδέουν πάνω στο παγκόσμιο αυτό δίκτυο το οποίο αρχίζει να γίνεται γνωστό σαν INTERNET και να εξαπλώνεται με τρομερούς ρυθμούς σε ολόκληρο τον κόσμο. Το πιο καθοριστικό όμως βήμα έγινε τη δεκαετία του ΄90. Το 1993, το εργαστήριο CERN στην Ελβετία παρουσιάζει το World Wide Web (WWW) (Παγκόσμιο Ιστό) που αναπτύχθηκε από τον Tim Berners-Lee. Πρόκειται για ένα σύστημα διασύνδεσης πληροφοριών σε μορφή πολυμέσων (multimedia) που βρίσκονται αποθηκευμένες σε χιλιάδες υπολογιστές του Internet σε ολόκληρο τον κόσμο και παρουσίασής τους σε

ηλεκτρονικές σελίδες, στις οποίες μπορεί να περιηγηθεί κανείς χρησιμοποιώντας το ποντίκι.[8]

Μαζί με το Internet όμως εξελίσσονταν και τα τραπεζικά συστήματα. Το 1980 εγκαταστάθηκαν στην Αμερική τα πρώτα home banking συστήματα. Με το home banking οι τράπεζες εκμεταλλευόμενες το διαδίκτυο επέτρεπαν στους πελάτες τους να εκτελούν μέσω ηλεκτρονικού υπολογιστή τραπεζικές συναλλαγές.

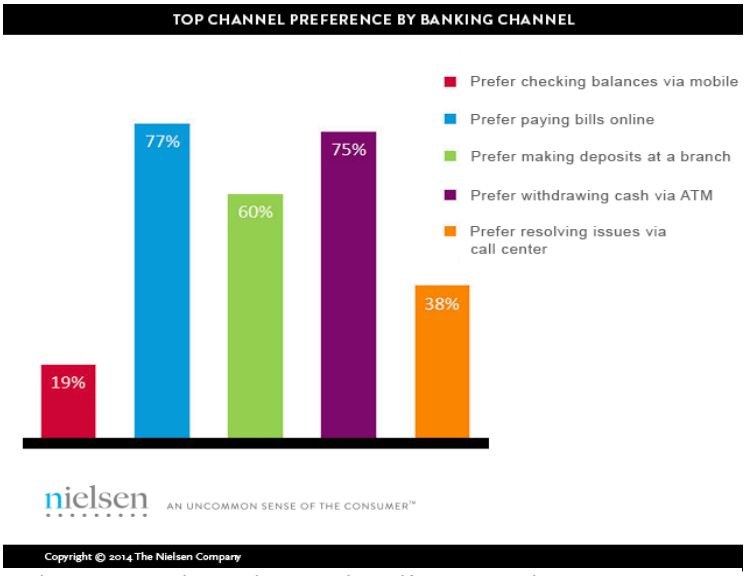
Με την διάδοση του internet και την έλευση των πρώτων φυλλομετρητών το home banking αντικαταστάθηκε από κάτι πιο εξελιγμένο, το online banking. Τα βασικά πλεονεκτήματα του online banking έναντι του home banking είναι πως πρώτον δεν απαιτεί από τον πελάτη να έχει κάποιο ειδικό λογισμικό ή συσκευή, παρά μόνο σύνδεση στο διαδίκτυο κι έναν browser εγκατεστημένο και δεύτερον οι τράπεζες δεν υποχρεούνταν να συντηρούν ιδιωτικά δίκτυα που σημαίνει μείωση του κόστους.

Η εξέλιξη και η ευρεία χρήση του mobile banking δεν είναι κάτι θεωρητικό αλλά βασίζεται σε έρευνες που αποδεικνύουν πως πλέον οι πελάτες των τραπεζών προτιμούν αυτό τον τρόπο συναλλαγής από τον παλιό παραδοσιακό που περιλάμβανε επίσκεψη στο υποκατάστημα της τράπεζας σου. Σύμφωνα με έρευνα[10] που εκπονήθηκε από την Nielsen το 2014, το 82% των Αμερικανών πολιτών προτιμούν το online banking αφού δήλωσαν ότι το χρησιμοποιήσαν τουλάχιστον μία φορά τον τελευταίο μήνα, σε σχέση με το 68% που δήλωσε ότι επισκέφθηκε φυσικό υποκατάστημα την ίδια περίοδο.



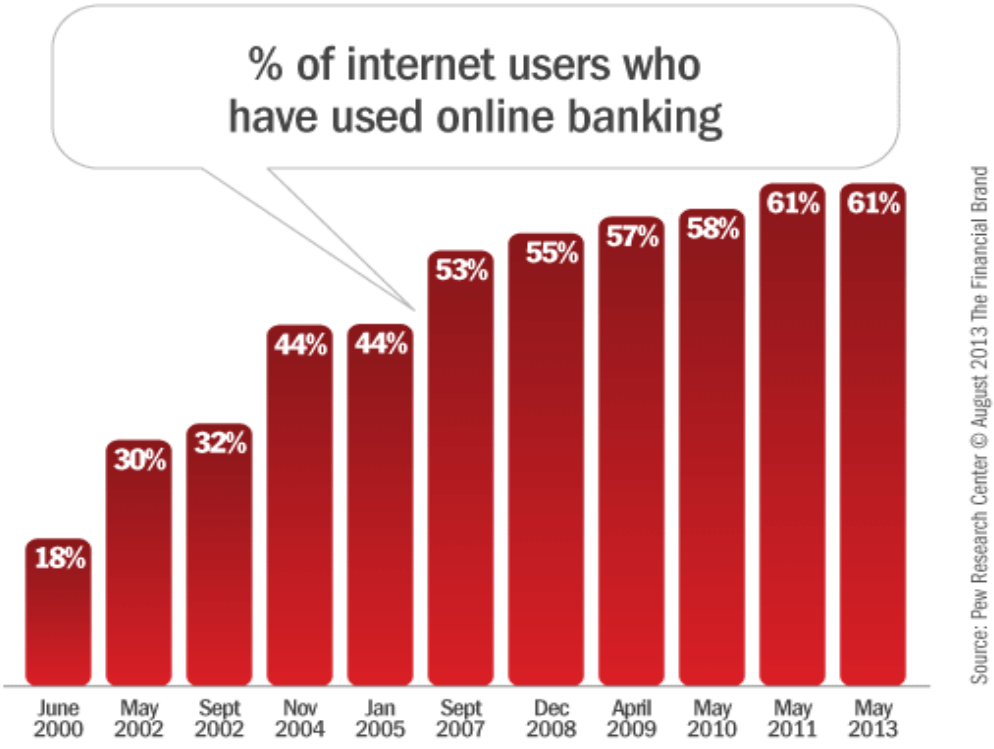
Εικόνα 4- Ποσοστό χρηστών που χρησιμοποιεί κάθε κανάλι επικοινωνίας με την τράπεζα

Επίσης η ίδια έρευνα παρουσίασε κι ένα άλλο σχετικό αποτέλεσμα. Το 77% των ερωτηθέντων απάντησε πως προτιμάει να πληρώνει τους λογαριασμούς του online. Αυτό αποδεικνύει την ολοκληρωτική στροφή των πελατών σε online υπηρεσίες έναντι των παραδοσιακών.



Εικόνα 5- Κορυφαία προτίμηση ανά κανάλι επικοινωνίας

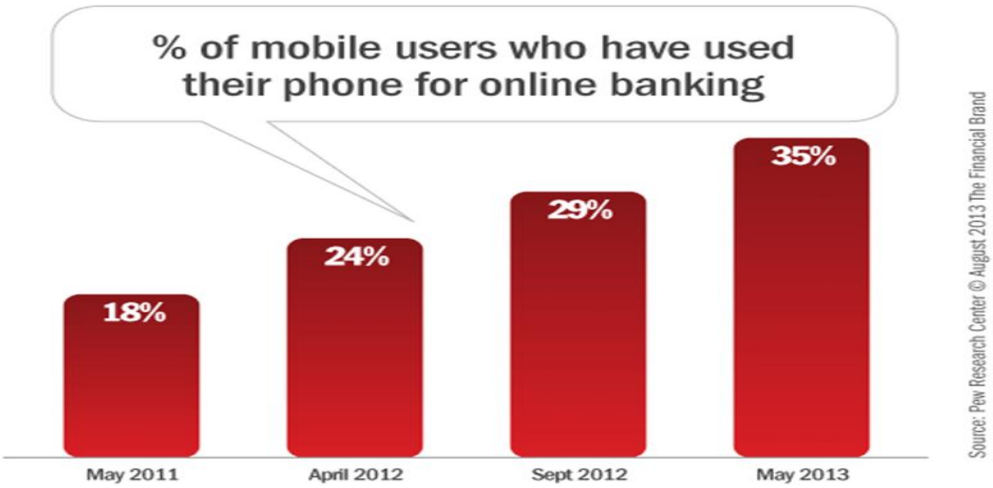
Την παραπάνω έρευνα έρχεται να επιβεβαιώσει μια άλλη έρευνα που διενεργήθηκε από το Pew Internet & American Life Project. Σύμφωνα με αυτή λοιπόν το ποσοστό των



Εικόνα 6-Ποσοστό χρηστών του Internet που κάνουν επίσης χρήση του online banking

χρηστών που κάνουν χρήση online banking έχει αυξηθεί από το 18% του Ιουνίου του 2000 στο 61% του Μαΐου του 2013.

Παρόμοια είναι και η αύξηση των χρηστών που χρησιμοποιούν το κινητό τους για να έχουν πρόσβαση σε online banking υπηρεσίες.[11]



Εικόνα 7- Ποσοστό χρηστών που χρησιμοποίησαν online banking υπηρεσίες μέσω κινητού

2.3 Είδη online banking

Το online banking μπορεί να κατηγοριοποιηθεί σε 4 μεγάλες κατηγορίες με βάση το μέσο που χρησιμοποιείται. Θα τις δούμε συνοπτικά καθώς επίσης και τις υποκατηγορίες για την κάθε κατηγορία.[14]

- Με την χρήση τηλεφωνικής συσκευής. Στην κατηγορία αυτή ανήκουν το phone banking και το mobile banking. Το phone banking είναι η παροχή τραπεζικών υπηρεσιών μέσω παραδοσιακής τηλεφωνικής γραμμής. Ο πελάτης μπορεί να αποκτήσει τις επιθυμητές πληροφορίες καλώντας έναν συγκεκριμένο αριθμό και επιβεβαιώνοντας την ταυτότητα του μέσω ενός τρόπου που έχει προ-συμφωνηθεί. Ο πελάτης μπορεί να αποκτήσει πληροφορίες για τα τραπεζικά του προϊόντα καθώς επίσης και να εκτελέσει κάποια πληρωμή. Σε αυτή την περίπτωση ένα μηχάνημα φαξ χρησιμοποιείται συνήθως σαν μέσο επικοινωνιακής εξόδου. Ένα πλεονέκτημα του phone banking είναι πως απαιτεί μόνο την ύπαρξη τηλεφωνικής συσκευής και γραμμής και είναι διαθέσιμο 24 ώρες την ημέρα, 7 ημέρες την εβδομάδα. Το mobile banking αποτελείται από τα SMS banking, GSM SIM Toolkit and WAP.Απαραίτητη προϋπόθεση είναι η ύπαρξη κινητής τηλεφωνικής συσκευής. Στο SMS banking

η ενημέρωση γίνεται μέσω μηνυμάτων SMS ενώ στα άλλα είδη γίνεται μέσω των τεχνολογιών που αναφέρονται στην ονομασία τους (GSM,WAP)

- Με την χρήση ηλεκτρονικών υπολογιστών. Πιο διαδεδομένο στις μέρες μας είναι το Internet Banking που πέρα από την ύπαρξη ηλεκτρονικού υπολογιστή, απαιτεί μόνο την ύπαρξη internet browser. Σε αυτή την κατηγορία επίσης ανήκουν τα home banking και mail banking.
- Με την χρήση μέσων πληρωμής, όπως είναι οι κάρτες, τα μηχανήματα POS και τα ηλεκτρονικά πορτοφόλια.
- Με την χρήση self-service zones. Πρόκειται για χώρους με τερματικά και συσκευές όπου ο πελάτης διεκπεραιώνει τις τραπεζικές του συναλλαγές χωρίς την ύπαρξη κάποιου υπαλλήλου.

2.4 Τυπικός τρόπος λειτουργίας online banking

Για να αποκτήσει πρόσβαση ένας πελάτης στα online συστήματα του ιδρύματος που επιθυμεί θα πρέπει να ακολουθήσει μια διαδικασία η οποία είναι παρόμοια στην πλειονότητα των περιπτώσεων. Αρχικά λοιπόν θα πρέπει να εγγραφεί σε αυτή την υπηρεσία ώστε να αποκτήσει τα απαραίτητα διαπιστευτήρια, δηλαδή όνομα χρήστη και κωδικό πρόσβασης. Συνήθως η απόκτηση αυτών των διαπιστευτηρίων προϋποθέτει και την καταβολή κάποιου μικρού σχετικά ποσού.

Στην συνέχεια ο πελάτης επισκέπτεται τον ασφαλή ιστότοπο του ιδρύματος ώστε να ολοκληρώσει την εγγραφή του και να έχει πρόσβαση πλέον στις ηλεκτρονικές υπηρεσίες. Ανάλογα με το πακέτο εγγραφής που διάλεξε ο πελάτης, διαφέρουν και οι προσφερόμενες δυνατότητες που του παρέχει το ίδρυμα.

2.5 Δυνατότητες online banking

Πλέον το online banking παρέχει μια πληθώρα δυνατοτήτων στους χρήστες του και δεν περιορίζεται στην απλή απεικόνιση υπολοίπων. Πιο αναλυτικά:

- **Πληροφόρηση για τους λογαριασμούς:** ο πελάτης μπορεί να ενημερωθεί για τους λογαριασμούς του, τα υπόλοιπα(λογιστικά-διαθέσιμα), τις κινήσεις που

έχει πραγματοποιήσει σε συγκεκριμένο ημερομηνιακό διάστημα καθώς επίσης και άλλες λεπτομέρειες για τον λογαριασμό(είδος, συνδικαιούχοι κτλ)

- **Άνοιγμα λογαριασμού online.** Ο πελάτης μπορεί να ανοίξει λογαριασμό στην τράπεζα χωρίς να χρειαστεί να μεταβεί σε κάποιο φυσικό κατάστημα.
- **Μεταφορές χρημάτων:** ο πελάτης μπορεί να μεταφέρει χρήματα σε δικούς του λογαριασμούς εντός της τράπεζας και σε άλλους λογαριασμούς εντός της τράπεζας χωρίς κάποιο κόστος.
- **Εκτέλεση εμβασμάτων:** ο πελάτης μπορεί να μεταφέρει χρήματα σε λογαριασμούς άλλων εγχώριων τραπεζών καθώς επίσης και σε λογαριασμούς τραπεζών ξένων χωρών με το ανάλογο κόστος , το οποίο είναι χαμηλότερο από το να τα έστελνε από τον γκισέ της τράπεζας.
- **Πληρωμές:** Ο πελάτης μπορεί να εκτελέσει μια ευρεία γκάμα πληρωμών στις οποίες περιλαμβάνεται η πληρωμή λογαριασμών δημόσιων οργανισμών, οφειλών σε δημόσιες υπηρεσίες, οφειλών προς ταμεία ασφάλισης, προς ασφαλιστικές εταιρίες, προς παρόχους κινητής και σταθερής τηλεφωνίας.
- **Μαζικές πληρωμές - Μισθοδοσία**
- **Διαχείριση πιστωτικών καρτών:**
    - Πληρωμή πιστωτικών καρτών ιδίου: Ο χρήστης επιλέγει τον τραπεζικό λογαριασμό χρέωσης και τον αριθμό της πιστωτικής κάρτας που επιθυμεί να πληρώσει. Ακολούθως πληκτρολογεί το ποσό που θέλει να μεταφέρει για την πληρωμή της πιστωτικής κάρτας και την ημερομηνία που επιθυμεί να γίνει η πληρωμή. Ο χρήστης έχει την πολυτέλεια και μεταχρονολογημένων πληρωμών, γεγονός που τον διευκολύνει να προγραμματίζει τις πληρωμές του.
    - Πληρωμή πιστωτικών καρτών τρίτου: Ο χρήστης επιλέγει τον τραπεζικό λογαριασμό χρέωσης, στη συνέχεια καλείται να πληκτρολογήσει τον αριθμό της πιστωτικής κάρτας. Ο χρήστης πρέπει να είναι ιδιαίτερα προσεκτικός στο σημείο αυτό, ώστε τα λεφτά να πιστωθούν στη σωστή πιστωτική κάρτα. Ακολούθως πληκτρολογεί το ποσό που θέλει να μεταφέρει για την πληρωμή της πιστωτικής κάρτας και την ημερομηνία που επιθυμεί να γίνει η πληρωμή.
    - Πληρωμή πιστωτικών καρτών άλλης τράπεζας: Οι πληρωμές πιστωτικών καρτών άλλης τράπεζας διεκπεραιώνεται μέσω του

διατραπεζικού συστήματος Dias transfer. Για την πληρωμή πιστωτικών καρτών άλλης τράπεζας, ο χρήστης επιλέγει τον τραπεζικό λογαριασμό χρέωσης στη συνέχεια επιλέγει την τράπεζα δικαιούχου, από να σύνθετο πεδίο που περιέχει όλες τις τράπεζες εσωτερικού. Έπειτα ο πελάτης καλείται να πληκτρολογήσει τον αριθμό της πιστωτικής κάρτας. Ακολούθως πληκτρολογεί το ποσό που θέλει να μεταφέρει για την πληρωμή της πιστωτικής κάρτας και την ημερομηνία που επιθυμεί να γίνει η πληρωμή.

- **Διαχείριση δανείων:** ο πελάτης μπορεί να ενημερώνεται για το υπόλοιπο των δανείων που έχει στη συγκεκριμένη τράπεζα καθώς επίσης και να πληρώνει τα δόσεις του.
- **Διαχείριση πάγιων εντολών:** ο πελάτης μπορεί να δημιουργήσει πάγιες εντολές που θα εκτελούνται στις ημερομηνίες που έχει ορίσει αυτός, εφόσον υπάρχει υπόλοιπο στον λογαριασμό του, και δεν θα χρειάζεται να επεμβαίνει αυτός.
- **Διαχείριση προθεσμιακών καταθέσεων:** Υπάρχει η δυνατότητα ο χρήστης της τράπεζας να κλείνει ή να ανανεώνει τα χρήματα του σε προθεσμιακές καταθέσεις ανάλογα με τους όρους που ισχύουν την κάθε περίοδο.
- **Αιτήσεις:** Ο χρήστης μπορεί ηλεκτρονικά να κάνει αιτήσεις για την έκδοση βιβλιαρίου επιταγών, πιστωτικών καρτών, καινούριων κωδικών
- **Εκτύπωση statements:** ο πελάτης μπορεί να εκτυπώσει συγκεντρωτικά τις κινήσεις ενός χρονικού διαστήματος ή μεμονωμένα μια κίνηση
- **Εργαλεία:** Πολλές τράπεζες παρέχουν στους χρήστες τους διάφπρα εργαλεία όπως είναι ο υπολογιστής δανείου όπου μπορούμε να υπολογίσουμε τις δόσεις του δανείου, ο υπολογιστής IBAN που μετατρέπει τον λογαριασμό σε IBAN, μετατροπέας νομίσματος και τρέχουσες τιμές συναλλάγματος.

2.6 Πλεονεκτήματα στο online banking

Τα πλεονεκτήματα του online banking οφείλουμε να τα μελετήσουμε τόσο από την πλευρά των πελατών, όσο κι από την πλευρά των τραπεζών.

2.6.1 Από την πλευρά των πελατών

- **<u>Συνεχής πρόσβαση στο σύστημα</u>**

Ο πελάτης μπορεί να εξυπηρετείται από την τράπεζα οποιαδήποτε στιγμή, δεδομένου ότι αποκτά πρόσβαση στις τραπεζικές υπηρεσίες 24 ώρες το 24ωρο, κάθε μέρα ανεξάρτητα αν είναι καθημερινή, σαββατοκύριακο ή αργία. Αντιθέτως μέσω ενός κανονικού καταστήματος η πρόσβαση του θα περιοριζόταν στις εργάσιμες ώρες και ημέρες.

- **Πρόσβαση από παντού**

  Ο χρήστης δεν έχει περιορισμό ως προς το σημείο πρόσβασης. Μπορεί να το κάνει από τον υπολογιστή του σπιτιού του, από το κινητό στον δρόμο, από το γραφείο. Το μόνο που χρειάζεται είναι μια συσκευή με πρόσβαση στο διαδίκτυο.

- **Εξοικονόμηση χρόνου**

  Με το να μην χρειάζεται η επίσκεψη σε φυσικό κατάστημα ο πελάτης γλιτώνει χρόνο από τις ουρές στις τράπεζες και από την κίνηση που θα συναντήσει στον δρόμο. Επίσης δεν είναι υποχρεωμένος να κάνει τις συναλλαγές του τις ώρες που είναι ανοιχτά τα υποκαταστήματα, οι οποίες συνήθως συμπίπτουν με τις ώρες εργασίας του, αλλά όποτε τον βολεύει εκείνον.

- **Μείωση χαρτιού και αρχείων**

Ο χρήστης έχει πρόσβαση σε λογαριασμούς και κινήσεις κάθε ώρα και στιγμή χωρίς να χρειάζεται να κρατάει εκτυπωμένο αρχείο, γλιτώνοντας με αυτό τον τρόπο χαρτί, χώρο καθώς επίσης και την πιθανότητα να χάσει τις πληροφορίες και να χρειαστεί να ξαναπάει στο κατάστημα.

- **Ασφάλεια**

Τα ηλεκτρονικά τραπεζικά συστήματα παρέχουν ιδιαίτερη ασφάλεια με διάφορες δικλίδες ασφαλείας και πολλαπλούς κωδικούς.

- **Ταχύτητα και άνεση**

Η ταχύτητα διεκπεραίωσης των συναλλαγών είναι μεγάλη αφού δεν υπάρχουν καθυστερήσεις

- **Πρόσβαση σε ευρύ φάσμα πληροφοριών και υπηρεσιών**

Τα sites των τραπεζών παρέχουν αρκετές πληροφορίες χρήσιμες για την ενημέρωση των πελατών πχ ημερολόγιο φορολογικών υποχρεώσεων μήνα, για χρηματοοικονομικά θέματα, για υπηρεσίες, χρήσιμες ηλεκτρονικές διευθύνσεις κλπ.

2.6.2 Από την πλευρά των τραπεζών

- **Επέκταση δικτύου εξυπηρέτησης**

Οι τράπεζες δεν περιορίζονται γεωγραφικά στα υφιστάμενα υποκαταστήματα , αλλά μπορούν να επεκταθούν παντού αφού στο διαδίκτυο δεν υπάρχουν σύνορα και όρια.

- **Παροχή καινοτόμων υπηρεσιών**

Οι τράπεζες εκμεταλλευόμενες τις δυνατότητες που τους παρέχουν το διαδίκτυο, οι γρήγορες ταχύτητες και οι σύγχρονοι browsers και ηλεκτρονικές συσκευές μπορούν να παρέχουν στους πελάτες τους πρωτοποριακές υπηρεσίες που σε άλλες συνθήκες δεν θα μπορούσαν να πραγματοποιηθούν.

- **Αύξηση αποδοτικότητας τραπεζών**

Μπορούν να εκτελεστούν περισσότερες συναλλαγές από ότι αν εκτελούνταν όλες στα καταστήματα. Επομένως αυξάνει και το κέρδος.

- **Μείωση κόστους**

Η ψηφιοποίηση των συναλλαγών μειώνει το κόστος των τραπεζών σε πρώτες ύλες .

2.7 Μειονεκτήματα και κίνδυνοι στο online banking

2.7.1 Από την πλευρά των πελατών

- **Ασφάλεια**

Παρότι στην προηγούμενη ενότητα αναφέραμε την ασφάλεια σαν πλεονέκτημα, μπορεί να λειτουργήσει και σαν μειονέκτημα. Αυτό συμβαίνει όχι τόσο εξαιτίας κάποιας αμέλειας των τραπεζών , αλλά γιατί οι ίδιοι οι πελάτες δεν λαμβάνουν την ίδια μέριμνα για την προστασία των δικών τους

μέσων. Μερικά παραδείγματα αμέλειας των χρηστών είναι να χρησιμοποιούν δημόσιους υπολογιστές για την πραγματοποίηση συναλλαγών και να ξεχνάνε να κάνουν logout ή να δημοσιοποιούν τους κωδικούς πρόσβασης. Παρακάτω θα γίνει εκτενέστερη ανάλυση των προβλημάτων ασφαλείας.

- **Απουσία διαπροσωπικής επαφής και έλλειψη επαρκούς υποστήριξης.**

2.7.2 Από την πλευρά των τραπεζών

- **Υψηλό κόστος αρχικής εγκατάστασης**

Οι τράπεζες χρειάζεται να ξοδέψουν ένα αρκετά μεγάλο ποσό σε hardware και software για την αρχική εγκατάσταση ενός e-banking συστήματος καθώς επίσης και στην εκπαίδευση του προσωπικού και την προώθηση της πλατφόρμας.

- **Λειτουργικοί κίνδυνοι**

Η τράπεζα οφείλει να είναι σε ετοιμότητα να αντιμετωπίσει λειτουργικούς κινδύνους όπως είναι οι τεχνικές δυσλειτουργίες, ανθρώπινα λάθη, λανθασμένες ή ανεπαρκείς επιχειρηματικές δομές

- **Κίνδυνοι φήμης**

Η τράπεζα πρέπει λοιπόν από την στιγμή που υιοθετήσει το ebanking, να είναι έτοιμη να ανταποκριθεί απόλυτα στις απαιτήσεις των πελατών της με ταχύτητα και συνέπεια. Οι πελάτες που ούτως ή άλλως είναι επιφυλακτικοί στα νέα κανάλια επικοινωνίας μπορεί να επηρεαστούν πολύ αρνητικά στην πρώτη δυσλειτουργία του συστήματος με αποτέλεσμα να κλονιστεί η εμπιστοσύνη τους στην τράπεζα.

2.8 Ασφάλεια στο online banking

Η τράπεζα και οι χρήστες έχουν να αντιμετωπίσουν πολλούς κινδύνους ασφαλείας. Τόσο η τράπεζα όσο και ο χρήστης οφείλουν να λάβουν τα μέτρα τους ώστε να αποφευχθούν τυχόν απώλειες κατά την εκτέλεση των τραπεζικών συναλλαγών. Οι κυριότεροι κίνδυνοι ασφαλείας είναι οι παρακάτω:

- Ιοί που εισβάλλουν στους υπολογιστές των χρηστών και των τραπεζών και έχουν σκοπό να τους βλάψουν και να υποκλέψουν στοιχεία.

- Sniffers που παρακολουθούν την κίνηση στο διαδίκτυο
- Trojan Horses
- Key loggers που πιθανόν να υποκλέψουν κωδικούς και αριθμούς καρτών που έχει πληκτρολογήσει ο χρήστης
- Phishing emails που σκοπό έχουν να κάνουν τον χρήστη να δώσει μόνος του τα στοιχεία του.

Από την μεριά του ο χρήστης αυτό που μπορεί να κάνει είναι να χρησιμοποιεί anti-virus προγράμματα και να προσέχει σε ποιον υπολογιστή συνδέεται και που κοινοποιεί τους κωδικούς του.

Η τράπεζα από την πλευρά της έχει πάρει τα εξής μέτρα

- Κρυπτογράφηση. Όλες οι συναλλαγές μεταξύ τράπεζας και πελάτη είναι κρυπτογραφημένες για να μην μπορεί κάποιος να υποκλέψει την συνομιλία και να κλέψει στοιχεία και κωδικούς
- Μοναδικά username και passwords για κάθε πελάτη
- Αριθμός αυθεντικότητας της συναλλαγής ή TAN. Ουσιαστικά πρόκειται για την ψηφιακή υπογραφή της συναλλαγής.
- Αυτόματη αποσύνδεση από το e-banking μετά από 10 λεπτά.
- Υποχρεωτική αλλαγή κωδικών
- Συσκευές παραγωγής token για την εκτέλεση συναλλαγών
- Two-factor authentication. Σε κάποιες περιπτώσεις δεν αρκεί μόνο ο κωδικός του χρήστη για να πραγματοποιηθεί μια συναλλαγή αλλά χρειάζεται κι ένας επιπλέον κωδικός ο οποίος αποστέλλεται συνήθως στο κινητό τηλέφωνο που έχει δηλώσει ο χρήστης.

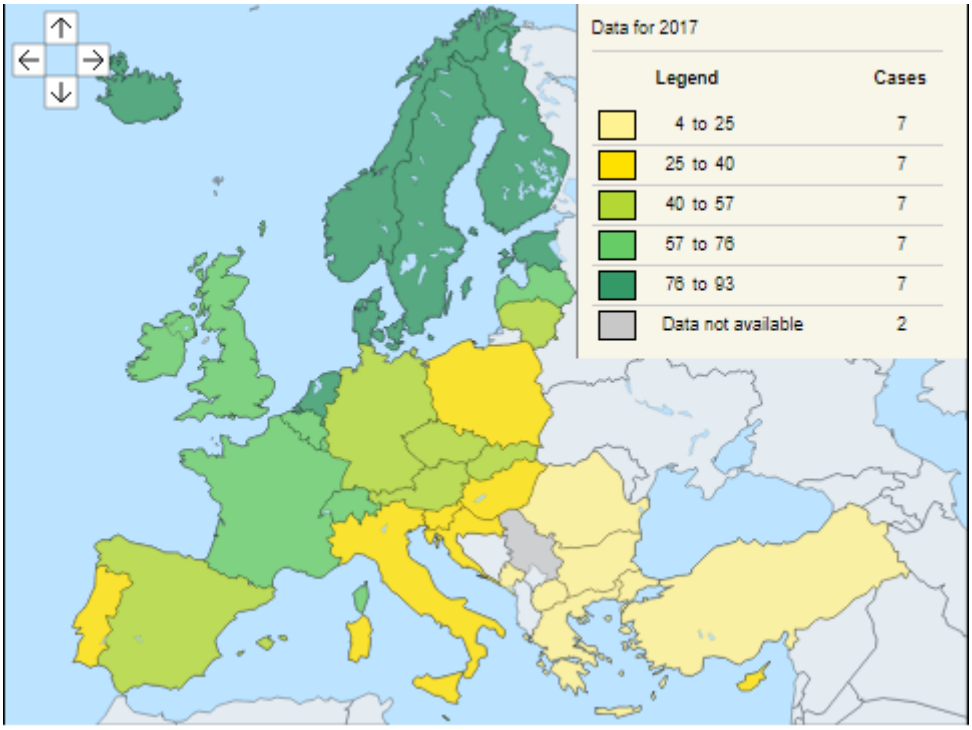
Κεφάλαιο 3

3.1 Το online banking στην Ελλάδα

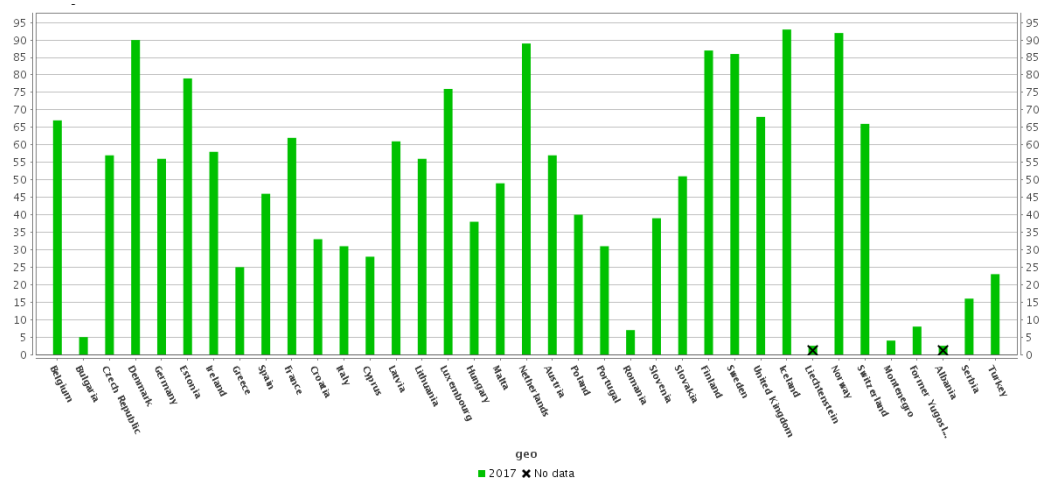
Το e-banking εμφανίστηκε στην Ελλάδα στα τέλη του 20ου αιώνα, γύρω στο 1997-1998 και το εισήγαγε πρώτη η Εγνατία Τράπεζα μέσω της εφαρμογής Web Teller. Μέσω της εφαρμογής αυτής οι πελάτες είχαν την δυνατότητα να εκτελέσουν τις τραπεζικές τους κινήσεις μέσω διαδικτύου. Παρόλα αυτά η επίσκεψη στην τράπεζα για την ολοκλήρωση των κινήσεων ήταν απαραίτητα. Το 2000 η τράπεζα Πειραιώς εισήγαγε την πρώτη ολοκληρωμένη πλατφόρμα τραπεζικών συναλλαγών με την ονομασία WinBank.

Με το πέρασμα των ετών όλες οι τράπεζες δημιούργησαν τα δικά τους συστήματα ώστε να εξυπηρετήσουν τους πελάτες τους. Πλέον οι πελάτες των τραπεζών μπορούν να εκτελέσουν μια πληθώρα συναλλαγών όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο.

Παρά την εξέλιξη όμως του ηλεκτρονικού τραπεζικού συστήματος οι Έλληνες πελάτες των τραπεζών φαίνονται δύσπιστοι στο να εμπιστευτούν το καινούριο σύστημα και στο να προσαρμοστούν σε αυτό.

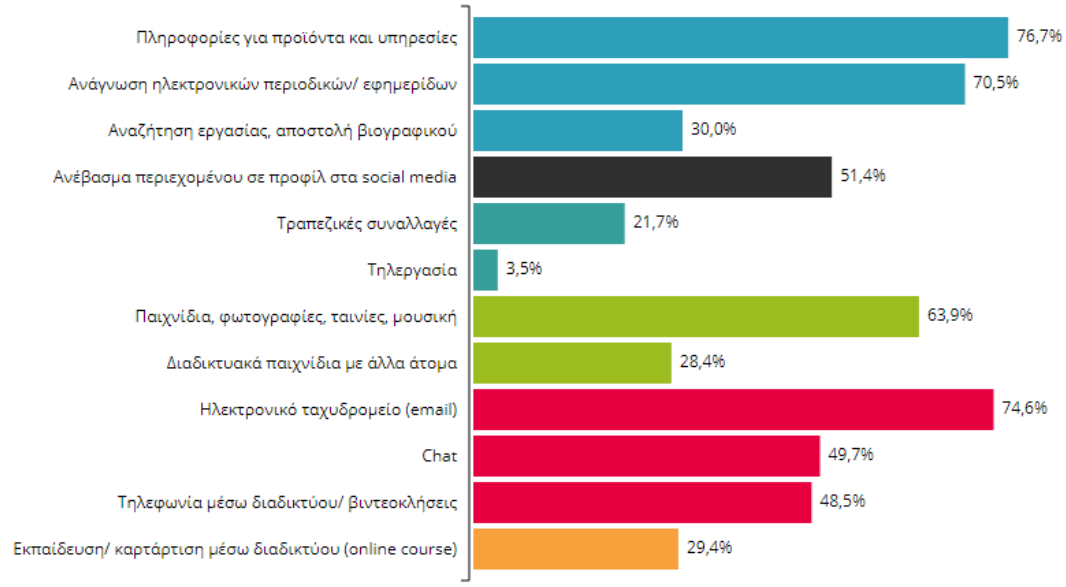


Εικόνα 7- Ποσοστά χρήσης e-banking στην Ευρώπη



Εικόνα 8- Χρήση e-banking στην Ευρώπη

Όπως φαίνεται στα διαγράμματα εικόνων 9 και 10 [12] σύμφωνα με την έρευνα της Eurostat η Ελλάδα έχει τα χαμηλότερα ποσοστά χρήσης online banking ανάμεσα στις χώρες της Ευρώπης. Το αποτέλεσμα αυτό βέβαια δεν προκαλεί καμιά έκπληξη αν δούμε και την έρευνα[13] από το Παρατηρητήριο για την Κοινωνία της Πληροφορίας όπου βλέπουμε πως μόνο το 21.7% των Έλληνων χρηστών του διαδικτύου χρημοποιεί το internet για την εκτέλεση τραπεζικών συναλλαγών.



Εικόνα 9-Λόγοι χρήσης internet στην Ελλάδα

3.2 Alpha Bank

Ο Όμιλος Alpha Bank είναι ένας από τους μεγαλύτερους Ομίλους του χρηματοοικονομικού τομέα στην Ελλάδα. Προσφέρει ευρύ φάσμα υψηλής ποιότητας χρηματοοικονομικών προϊόντων και υπηρεσιών, συμπεριλαμβανομένων της λιανικής τραπεζικής, της τραπεζικής μεσαίων και μεγάλων επιχειρήσεων, της διαχειρίσεως κεφαλαίων και private banking, της διανομής ασφαλιστικών προϊόντων, της επενδυτικής τραπεζικής, των χρηματιστηριακών εργασιών και της διαχειρίσεως ακίνητης περιουσίας.

Μητρική Εταιρία και βασική Τράπεζα του Ομίλου είναι η Alpha Bank, η οποία ιδρύθηκε το 1879 από τον Ιωάννη Φ. Κωστόπουλο. Η Alpha Bank, Τράπεζα εμπιστοσύνης και σταθερό σημείο αναφοράς στο ελληνικό τραπεζικό σύστημα διαθέτει έναν από τους υψηλότερους δείκτες κεφαλαιακής επάρκειας στην Ευρώπη.

Σημαντικοί σταθμοί των τελευταίων ετών στη μακρά και επιτυχή διαδρομή του Ομίλου, είναι:

- Η ολοκλήρωση της εξαγοράς των εργασιών Λιανικής Τραπεζικής της Citibank, την 30.9.2014.
- Η εξαγορά του συνόλου των προνομιούχων μετοχών του Ελληνικού Δημοσίου από την Τράπεζα, η οποία πρώτη από τις συστημικές τράπεζες προέβη στην αποπληρωμή της συμμετοχής του.
- Η επιτυχής ολοκλήρωση της αυξήσεως του Μετοχικού Κεφαλαίου της Τραπέζης ύψους Ευρώ 1,2 δισ., την 31.3.2014.
- Η ολοκλήρωση της νομικής συγχωνεύσεως δι' απορροφήσεως της Εμπορικής Τραπέζης, την 28.6.2013.[21]

Οι υπηρεσίες online banking που υποστηρίζει η Alpha Bank είναι οι εξής[20]:

3.2.1 Ενημέρωση

Λογαριασμοί	AWB
Συνολικό υπόλοιπο καταθετικών λογαριασμών ευρώ	✓
Υπόλοιπα λογαριασμών ευρώ, ξένο νόμισμα	✓
Κινήσεις και πρόσθετη πληροφόρηση λογαριασμών	✓
Πληροφορίες εισερχομένων εντολών	✓ **
Κατάσταση επιταγών	✓
Ηλεκτρονικά αντίγραφα λογαριασμών (Alpha e-statements)	✓

Εικόνα 10- Alpha Ενημέρωση

Κάρτες	AWB
Συνολικό τρέχον υπόλοιπο καρτών	✓
Υπόλοιπο κάρτας εκδόσεως Τραπέζης	✓
Τελευταίες κινήσεις κάρτας εκδόσεως Τραπέζης*	✓
Παλαιότερα Αντίγραφα κάρτας εκδόσεως Τραπέζης	✓
Bonus πόντοι καρτών εκδόσεως Τραπέζης	✓
Ιστορικό μεταφορών Bonus καρτών	✓
Ηλεκτρονικά αντίγραφα καρτών (Alpha e-statements)	✓

Εικόνα 11- Alpha Ενημέρωση

Δάνεια	AWB
Συνολικό υπόλοιπο δανείων	✓
Υπόλοιπο στεγαστικού δανείου	✓
Κινήσεις στεγαστικού δανείου (10 τελευταίες)	✓
Υπόλοιπο Alpha Ανοικτού Δανείου και Alpha 700	✓
Κινήσεις Alpha Ανοικτού Δανείου και Alpha 700*	✓
Ηλεκτρονικά αντίγραφα δανείων (Alpha e-statements)	✓

Εικόνα 13- Alpha Ενημέρωση

Επενδύσεις	AWB
Χαρτοφυλάκιο μετοχών Alpha Finance	✓
Χαρτοφυλάκια Alpha Αμοιβαίων Κεφαλαίων	✓
Χαρτοφυλάκιο Alpha Gold Personal Banking	✓
Χαρτοφυλάκιο Alpha Private Banking	✓
Καταθέσεις προσθεσμίας	✓

Πληροφορίες γενικού ενδιαφέροντος	AWB
Υπολογισμός αφορολογήτου σύμφωνα με το εισόδημα	✓
Συναλλαγές που υπολογίζονται στο αφορολόγητο	✓

Εικόνα 12-Alpha Ενημέρωση

3.2.2 Μεταφορές

Στην Alpha Bank - Άμεσα (1)	AWB
Μεταφορά μεταξύ λογ/σμών καταθέσεων του προφίλ της συνδρομής	✓
Μεταφορά σε λογ/σμό τρίτου (ή σε μη προδηλωμένο λογαριασμό)	✓
Μεταφορά σε προδηλωμένους λογαριασμούς Alpha Bank από κάρτα ή δάνειο	✓

Εικόνα 14- Alpha Μεταφορές

Στην Alpha Bank - σε μελλοντική ημερομηνία	AWB
Μεταφορά μεταξύ λογ/σμών καταθέσεων	✓
Μεταφοράς σε λογ/σμό καταθέσεων τρίτου(2)	✓
Ορισμός επαναλαμβανόμενων μεταφορών μεταξύ λογαριασμών	✓

Εικόνα 15- Alpha Μεταφορές

Σε άλλη τράπεζα	AWB
Εμβασμα σε λογ/σμό τραπέζης εσωτερικού (2)	✓
Εμβασμα σε λογ/σμό τραπέζης εξωτερικού (2)	✓

Εικόνα 16- Alpha Μεταφορές

3.2.3 Πληρωμές

Κάρτες - Δάνεια	AWB
Κάρτες εκδόσεως Τραπέζης	✓
Κάρτες εκδόσεως άλλων Τραπεζών (DIASTRANSFER)	✓
Φόρτιση προπληρωμένης κάρτας	✓
Δάνειο Alpha Επιλογή/Alpha 700	✓
Λοιπά καταναλωτικά/προσωπικά δάνεια	✓

Εικόνα 17- Alpha Πληρωμές

Δημόσιο/ Δ.Ε.Κ.Ο.

Τηλεφωνία (Σταθερή, Κινητή & Internet)

Ασφαλιστικές Εταιρίες

Λοιπές Εταιρίες

Δωρέες υπέρ κοινωνικού σκοπού

Εικόνα 18- Alpha Πληρωμές

3.2.4 Online Προϊόντα -Αιτήσεις

Online προϊόντα:

- άνοιγμα του νέου αποταμιευτικού λογαριασμού "Alpha Έξυπνη Αποταμίευση",
- άνοιγμα και διαχείριση (μεταβολή συνδεδεμένου λογαριασμού και προεξόφληση) της Alpha online Προθεσμιακής και της Alpha online Προθεσμιακής με Bonus,
- έκδοση χρεωστικής κάρτας (Visa, Mastercard, American Express),
- έκδοση, φόρτιση ή αποφόρτιση προπληρωμένης κάρτας.

Online διαχείριση καρτών:

- επανέκδοση PIN της χρεωστικής ή πιστωτικής σας κάρτας,
- δήλωση απώλειας / κλοπής της κάρτας σας για άμεση ακύρωσή της,
- ενεργοποίηση της νέας σας κάρτας,
- μεταβολή του ορίου αναλήψεως των καρτών σας,
- μεταβολή του ορίου αγορών με την χρεωστική σας κάρτα,
- μεταβολή των στοιχείων αυτόματης εξοφλήσεως,
- διαχείριση των συνδεδεμένων τραπεζικών λογαριασμών με κάθε κάρτα σας,
- επανέκδοση της κάρτας σας λόγω φθοράς.

Online διαχείριση ηλεκτρονικών εργαλείων:

- ενεργοποίηση και διαχείριση των Υπηρεσιών Πρόσθετου Κωδικού Ασφαλείας,
- ενεργοποίηση των υπηρεσιών Alpha e-statements και Alpha alerts,
- ενεργοποίηση και διαχείριση της υπηρεσίας Alpha Quick Transfer.

Άλλες online αιτήσεις:

- έκδοση βιβλιαρίου επιταγών με 20% έκπτωση από το Κατάστημα,
- σύνδεση του Χαρτοφυλακίου μετοχών σας,
- προσθήκη προϊόντων στο προφίλ σας.

3.2.5 Διαχείριση –Ρυθμίσεις

Οι χρήστες μπορούν να ορίζουν:

- Τα στοιχεία του προφίλ σας (Μυστικό Κωδικό και Κωδικό Συνδρομητή)
- Τις "Φιλικές Ονομασίες", τα δικαιώματα χρήσεως και τα χρώματα ανά προϊόν
- Τα ανώτατα όρια μεταφορών σε μη προδηλωμένους λογαριασμούς Alpha Bank και άλλων τραπεζών μέσω των υπηρεσιών Alpha e-Banking.

Οι χρήστες μπορούν να επιλέγουν:

- Τα προϊόντα που θα εμφανίζονται στην οθόνη της "Επισκόπησης"
- Τις πρόσθετες υπηρεσίες (Υπηρεσίες Πρόσθετου Κωδικού Ασφαλείας, Alpha e-statements, Alpha alerts, Alpha Secure Web) που μπορείτε να ενεργοποιήσετε στη συνδρομή σας
- Το φόντο των υπηρεσιών Alpha Web Banking και Alpha Mobile Banking.

Οι χρήστες μπορούν να διαχειρίζονται:

- Τα online προϊόντα σας και τις αιτήσεις τους
- Τις πρόσθετες υπηρεσίες που έχετε ενεργοποιήσει στο προφίλ σας
- Τις συναλλαγές (μεταφορές, πληρωμές) που εκτελείτε συχνά, αποθηκεύοντάς τες στο προφίλ σας για εύκολη πρόσβαση από την "Επισκόπηση".

3.3 Εθνική Τράπεζα της Ελλαδός

Η Εθνική Τράπεζα ιδρύθηκε το 1841 και αποτέλεσε την πρώτη τράπεζα του νεοελληνικού κράτους, με καθοριστική συνεισφορά στην οικονομική ζωή του τόπου σε αυτά τα 177 χρόνια ιστορίας της. Σήμερα η Εθνική ηγείται ενός από τους μεγαλύτερους και ισχυρότερους Ομίλους χρηματοοικονομικών υπηρεσιών στην Ελλάδα, με δυναμική συμβολή στη στήριξη της Ελληνικής οικονομίας και τον αναπτυξιακό και κοινωνικό μετασχηματισμό της χώρας.

Η Εθνική προσφέρει ευρύ φάσμα χρηματοοικονομικών προϊόντων και υπηρεσιών που ανταποκρίνονται στις συνεχώς μεταβαλλόμενες ανάγκες επιχειρήσεων και ιδιωτών.

Δίκαια μπορεί να χαρακτηριστεί ως η Τράπεζα της Ελληνικής Οικογένειας, καθώς ελέγχει το ¼ της λιανικής τραπεζικής. Το ηγετικό μερίδιο καταθέσεων ταμιευτηρίου αντανακλά την εμπιστοσύνη του αποταμιευτικού κοινού που αποτελεί και την κινητήρια δύναμή της.

Με 542 Μονάδες και 1.466 ATM, διαθέτει το ευρύτατο δίκτυο εξυπηρέτησης, καλύπτοντας ολόκληρη τη γεωγραφική έκταση της Ελλάδας, ενώ παράλληλα αναπτύσσει σύγχρονα ηλεκτρονικά κανάλια, όπως οι υπηρεσίες Mobile και Internet Banking. Σήμερα, το Δίκτυο της Τράπεζας στο εξωτερικό περιλαμβάνει 93 μονάδες, ενώ ο Όμιλος συνολικά απασχολεί 11.500 εργαζόμενους (στοιχεία 31.03.2018).[19]

Οι υπηρεσίες online banking που υποστηρίζει η Εθνική Τράπεζα της Ελλαδός είναι οι εξής [16][17][18]:

3.3.1 Ενημέρωση

INTERNET BANKING	Φυσικά Πρόσωπα	Νομικά Πρόσωπα
ΠΛΗΡΟΦΟΡΗΣΗ (1), (2), (3)		
ΚΑΡΤΕΣ		
Στοιχεία πιστωτικών καρτών έκδοσης Εθνικής Τράπεζας (πιστωτικό υπόλοιπο, πιστωτικό όριο, τρέχουσα συνολική οφειλή, ποσό μελλοντικών δόσεων κ.ά.)	√	
Στοιχεία κάρτας "Prepaid Visa" έκδοσης Εθνικής Τράπεζας (διαθέσιμο υπόλοιπο κάρτας, κίνηση κάρτας -ημερομηνία, περιγραφή συναλλαγής, ποσό κ.ο.κ.)	√	
Κίνηση πιστωτικών καρτών έκδοσης Εθνικής Τράπεζας (ημερομηνία, περιγραφή συναλλαγής, ποσό)	√	
i-Statements: ΕΘΝΟdeposit Λήψη ηλεκτρονικού αντιγράφου κίνησης καρτών ΕΘΝΟdeposit (ημερήσια, δεκαήμερη, & μηνιαία περιοδικότητα, για τους τελευταίους 36 μήνες).		√
Πληροφόρηση σε εμπόρους για κίνηση καρτών πελατών τους (έκδοσης Εθνικής Τράπεζας) μέσω των P.O.S.		√
Υπηρεσία i-bank Alert, με email ή γραπτό μήνυμα SMS σε κινητό τηλέφωνο. Διαθέσιμη για λογαριασμούς, προπληρωμένες ή πιστωτικές κάρτες έκδοσης Εθνικής Τράπεζας.	√	
Ενημέρωση κινήσεων κάρτας ΕΘΝΟdeposit: · Κινήσεις ΕΘΝΟDeposit ανά Κάρτα · Κινήσεις ΕΘΝΟDeposit ανά Λογαριασμό		√
Στοιχεία κάρτας ΕΘΝΟCASHPLUS & DEBIT MASTERCARD (συνδεδεμένοι λογαριασμοί, κινήσεις κάρτας, κ.ά.)	√	

Εικόνα 19- Εθνική Τράπεζα Ενημέρωση

ΚΑΤΑΘΕΣΕΙΣ		
Αναλυτικά στοιχεία καταθετικών λογαριασμών σε Ευρώ και Ξένο Νόμισμα (δικαιούχοι, αριθμός λογαριασμού/IBAN, νόμισμα, υπόλοιπο λογιστικό /διαθέσιμο, συχνότητα εκτοκισμού κ.λπ.)	√	√
Αναλυτικά στοιχεία προθεσμιακών λογαριασμών σε Ευρώ και Ξένο Νόμισμα (δικαιούχοι, ημερομηνία ανοίγματος /λήξης, επιτόκιο, αρχικό ποσό προθεσμίας, λογιστικό /διαθέσιμο υπόλοιπο, τόκοι μέχρι τη λήξη κ.λπ.)	√	√
Υπόλοιπα λογαριασμών	√	√
Κίνηση λογαριασμών	√	√
i-Statements: Λήψη ηλεκτρονικού αντιγράφου μηνιαίας κίνησης τρεχούμενων λογ/σμών & λογ/σμών όψεως Ιδιωτών, για τους τελευταίους 12 μήνες, με επιλογή έτους & μήνα.	√	√
Άμεση - σε πραγματικό χρόνο - ενημέρωση των τρεχουσών ημερήσιων συναλλαγών που πραγματοποιούνται στο λογαριασμό της επιχείρησης μέσω των ταμείων της Τράπεζας.		√
Εμφάνιση δικαιούχων σε πιστούμενους λογαριασμούς Εθνικής Τράπεζας	√	√
ΧΡΗΜΑΤΟΔΟΤΗΣΕΙΣ (Ν.Π.) - ΔΑΝΕΙΑ (Φ.Π.)		
Υπόλοιπα Χρηματοδοτικών Λογαριασμών (νόμισμα /είδος χρηματοδότησης /ημερομηνία)		√
Κίνηση Χρηματοδοτικού-Δανειακού Λογαριασμού		√
Δοσολόγιο Χρηματοδοτικού-Δανειακού Λογαριασμού (Μελλοντικές δόσεις κεφαλαίου χρηματοδοτήσεων)		√
Ανάλυση Επιτοκιακής Επιβάρυνσης		√
Δάνεια Εμπόρων (κίνηση ταμείου, διεκπεραίωση δόσεων δανείων πελατών)		√
Υπόλοιπα Στεγαστικού, Καταναλωτικού, Ανοικτού Δανείου Φ.Π.	√	
Κίνηση Στεγαστικού, Καταναλωτικού, Ανοικτού Δανείου Φ.Π.	√	
Δοσολόγιο Στεγαστικού, Καταναλωτικού, Ανοικτού Δανείου Φ.Π.	√	

Εικόνα 20- Εθνική Τράπεζα Ενημέρωση

ΕΠΕΝΔΥΣΕΙΣ		
Αγορά / Πώληση μετοχών	√	√
Τύχη Εντολής (αγοράς /πώλησης μετοχής)	√	√
Μεταβολή εντολών αγοράς / πώλησης μετοχών	√	√
Ερώτηση Χαρτοφυλακίου μετοχών	√	√
Κίνηση Χαρτοφυλακίου μετοχών	√	√
Ενημέρωση χαρτοφυλακίου Αμοιβαίων Κεφαλαίων	√	√

Εικόνα 21- Εθνική Τράπεζα Ενημλερωση

3.3.2 Μεταφορές

ΕΜΒΑΣΜΑΤΑ		
Έμβασμα σε λογαριασμό - ιδίου ή τρίτου - Εθνικής Τράπεζας [2]	√	√
Μαζικές πιστώσεις σε λογαριασμούς Εθνικής Τράπεζας.	√	√
Έμβασμα σε λογαριασμό άλλης τράπεζας εσωτερικού [2] -με επιλογή τρόπου καταβολής εξόδων (SHA,OUR ή BEN-κατά περίπτωση)	√	√
Μαζικά εμβάσματα σε λογαριασμούς τραπεζών εσωτερικού Εξυπηρέτηση Μισθοδοσίας (μόνο για Ν. Π.) (Αρχείο Εμβασμάτων - μέχρι 200 πιστώσεις ανά συναλλαγή. Δυνατότητα αποστολής μεγαλύτερου αριθμού εγγραφών μέσω ΕΘΝΟ6les)	√	√
Έμβασμα σε λογαριασμό άλλης τράπεζας εξωτερικού ***Εξυπηρετείται:*** ***α. έως 2000€ ανά δίμηνο*** ***β. με χρήση ελεύθερων κεφαλαίων από το εξωτερικό*** ***γ. με εισαγωγή αριθμού αιτήματος έγκρισης συναλλαγής-σύστημα ΕΚΑΕΣ***	√	√
Μαζικά εμβάσματα σε λογαριασμούς τραπεζών Ε.Ε. ή υπεράκτιων κτήσεων χωρών της. ***(δεν εξυπηρετείται βάσει των α/18.7.2015 ΠΝΠ (ΦΕΚ Α΄ 84)***	√	√

Εικόνα 22-Εθνική Τράπεζα Μεταφορές

3.3.3 Πληρωμές

Η Εθνική Τράπεζα υποστηρίζει τις πληρωμές των παρακάτω υπηρεσιών:

Εικόνα 23- Εθνική Τράπεζα Πληρωμές


ΠΛΗΡΩΜΕΣ ΔΗΜΟΣΙΟΥ
ΔΗΜΟΙ
ΠΛΗΡΩΜΕΣ ΠΙΣΤΩΤΙΚΩΝ ΚΑΡΤΩΝ
ΠΛΗΡΩΜΕΣ ΕΤΑΙΡΕΙΩΝ ΤΗΛΕΠΙΚΟΙΝΩΝΙΑΣ
ΠΛΗΡΩΜΕΣ ΑΣΦΑΛΙΣΤΙΚΩΝ ΤΑΜΕΙΩΝ
ΠΛΗΡΩΜΕΣ ΑΣΦΑΛΕΙΩΝ
ΠΛΗΡΩΜΕΣ ΕΤΑΙΡΕΙΩΝ ΕΝΕΡΓΕΙΑΣ
ΠΛΗΡΩΜΕΣ ΕΤΑΙΡΕΙΩΝ ΥΔΡΕΥΣΗΣ
ΠΛΗΡΩΜΕΣ ΕΤΑΙΡΕΙΩΝ ΕΚΠΑΙΔΕΥΣΗΣ
ΠΛΗΡΩΜΕΣ ΧΡΗΜΑΤΟΠΙΣΤΩΤΙΚΩΝ ΟΡΓΑΝΙΣΜΩΝ
ΔΩΡΕΕΣ
ΠΛΗΡΩΜΕΣ ΥΠΗΡΕΣΙΩΝ ΑΣΦΑΛΕΙΑΣ
ΠΛΗΡΩΜΕΣ ΛΟΙΠΩΝ ΕΤΑΙΡΕΙΩΝ
ΝΑΥΤΙΛΙΑ - ΜΕΤΑΦΟΡΕΣ
ΕΠΙΜΕΛΗΤΗΡΙΑ

3.3.4 Online Προϊόντα –Αιτήσεις

ΕΠΕΝΔΥΣΕΙΣ		
Αγορά Μετοχής (Επιλογές: LIMIT/MARKET/ATO/ATC)	√	√
Πώληση Μετοχής (Επιλογές: LIMIT/MARKET/ATO/ATC)	√	√
Τύχη Εντολής (αγοράς / πώλησης μετοχής)	√	√
Ερώτηση Χαρτοφυλακίου μετοχών	√	√
Κίνηση Χαρτοφυλακίου μετοχών	√	√
Ενημέρωση χαρτοφυλακίου Αμοιβαίων Κεφαλαίων	√	√
Προεγγραφή σε δημόσιες εγγραφές	√	√
Κατανομή Αιτήσεων Προεγγραφής	√	√
ΠΑΓΙΕΣ ΕΝΤΟΛΕΣ		
Εισαγωγή Πάγιας Εντολής (Επιλογές: Εκτέλεση σε μελλοντική ημερομηνία, περιοδικότητα, πλήθος πληρωμών)	√	√
Διαγραφή Πάγιας Εντολής	√	√
Πληροφορίες Πάγιων Εντολών	√	√
Ιστορικό Πάγιας Εντολής	√	√

Εικόνα 25- Εθνική Τράπεζα Online Προϊόντα- Αιτήσεις

ΕΘΝΟFILES: ΛΗΨΗ & ΑΠΟΣΤΟΛΗ ΗΛΕΚΤΡΟΝΙΚΩΝ ΑΡΧΕΙΩΝ		
Μαζικές συναλλαγές, με δυνατότητα προσαρμογής στις απαιτήσεις της επιχείρησης, μέσω του Internet Banking. Πρόκειται για υπηρεσία ανταλλαγής ηλεκτρονικών αρχείων προκαθορισμένης δομής μεταξύ της Τράπεζας και της επιχειρηματικής πελατείας της. Αφορά στις περιπτώσεις που τα διακινούμενα αρχεία περιέχουν μεγάλο αριθμό εγγραφών ή /και ειδικές προδιαγραφές ή /και κάποιο είδος συναλλαγής που δεν είναι διαθέσιμο στο μενού συναλλαγών του Internet Banking. Διαθέσιμες συναλλαγές: Αποστολή Αρχείου, Λήψη Αρχείου, Ιστορικό Κινήσεων		√
Πληροφόρηση σχετικά με : * συναλλακτική συμπεριφορά των εκχωρημένων οφειλετών & λειτουργία των λογ/σμών. * λογιστική παρακολούθηση του Factoring. - Αναζήτηση σχετικών εντύπων * Επικοινωνία με τους υπεύθυνους παρακολούθησης των λογαριασμών Factoring, * Αποστολή αλληλογραφίας μέσω ηλεκτρονικού ταχυδρομείου.		√
ΜΑΖΙΚΕΣ ΧΡΕΩΣΕΙΣ		
Μαζικές εισπράξεις από οφειλέτες με άμεση χρέωση λογαριασμών τους μέσω Internet Banking (κατόπιν πάγιας εντολής σε Κατάστημα της Τράπεζας)		√
Κατάσταση παγίων εντολών οφειλετών		√
Πληροφόρηση παγίων εντολών χρέωσης (για τον οφειλέτη)	√	√
ΑΙΤΗΣΕΙΣ		
Παραγγελία Μπλόκ Επιταγών	√	√
Ιστορικό Παραγγελιών	√	√
Αίτηση Έκδοσης Virtual MasterCard	√	
Αίτηση Έκδοσης Πιστωτικής Κάρτας	√	
Αίτηση Έκδοσης Χρεωστικής Κάρτας	√	
Αίτηση Έκδοσης Νέας Visa Prepaid	√	
Αίτηση χορήγησης κωδικού Phone Banking (υιοθέτηση υφιστάμενου UserID σε νέο εξοπλισμό ασφαλείας)	√	
Αίτηση χορήγησης συσκευής i-code (για κατοίκους εξωτερικού)	√	
Αίτηση αντικατάστασης συσκευής i-code (για κατοίκους εξωτερικού)	√	

Εικόνα 24-Εθνική Τράπεζα Online Προϊόντα- Αιτήσεις

3.4 Eurobank

Ο όμιλος Eurobank είναι ένας δυναμικός χρηματοοικονομικός οργανισμός με παρουσία σε οκτώ χώρες, σύνολο ενεργητικού €60,8 δισ. και ανθρώπινο δυναμικό 13.744 εργαζόμενους. Ξεκίνησε τη διαδρομή του το 1990 και έπειτα από μία δυναμική πορεία ανάπτυξης και συγχωνεύσεων, πρωτοστατεί τα τελευταία χρόνια στις εξελίξεις και τη διαμόρφωση του τραπεζικού περιβάλλοντος.

Με συνολικό δίκτυο 900 σημείων εξυπηρέτησης στην Ελλάδα και στο εξωτερικό, ο Όμιλος παρέχει ένα ολοκληρωμένο φάσμα χρηματοοικονομικών προϊόντων και υπηρεσιών σε ιδιώτες και επιχειρήσεις.

Στην Ελλάδα, η Eurobank είναι ένας από τους τέσσερις πυλώνες του τραπεζικού συστήματος. Με δύο διακριτά δίκτυα λιανικής, το Δίκτυο Eurobank και το Δίκτυο Νέο Ταχυδρομικό Ταμιευτήριο, εξειδικευμένα κέντρα εξυπηρέτησης επιχειρήσεων, ανεξάρτητο δίκτυο private banking και βραβευμένα ηλεκτρονικά δίκτυα, η φιλοσοφία της Eurobank εστιάζει στην παροχή υπηρεσιών υψηλής ποιότητας στους πελάτες της.

Ο Όμιλος επίσης διατηρεί στρατηγική θέση στη Βουλγαρία, τη Ρουμανία και τη Σερβία, διακρίνεται στον τομέα διαχείρισης περιουσίας στην Κύπρο, το Λουξεμβούργο και το Λονδίνο, ενώ έχει παρουσία και στην Ουκρανία.

Πέρα από την επιχειρηματική του δραστηριότητα, ο Όμιλος αναπτύσσει δράσεις κοινωνικής υπευθυνότητας, ανταποκρινόμενος στις ανάγκες των κοινωνιών όπου δραστηριοποιείται. Στο πλαίσιο αυτό υλοποιεί ενέργειες και πρωτοβουλίες που στηρίζουν τους τομείς της Παιδείας, της Κοινωνίας, του Πολιτισμού, της Καινοτομίας και της Νεανικής Επιχειρηματικότητας, σε συνεργασία με αναγνωρισμένους φορείς και οργανώσεις, ενώ παράλληλα δραστηριοποιείται έντονα σε θέματα προστασίας του Περιβάλλοντος, τόσο σε εθνικό όσο και σε διεθνές επίπεδο[23].

Οι υπηρεσίες online banking που υποστηρίζει η Eurobank είναι οι εξής[22]:

3.4.1 Ενημέρωση

Οι χρήστες του e-banking της Eurobank μπορούν να ενημερωθούν για :

a) Λογαριασμούς και προθεσμιακές καταθέσεις
   - Υπόλοιπο, δεσμευμένο και λογιστικό.

- Κίνηση λογαριασμού.
- Αριθμός λογαριασμού και IBAN.
- Δικαιούχοι λογαριασμού.
- Πιστωτικοί και χρεωστικοί τόκοι.
- Κατάστημα διαχείρισης.

b) Κάρτες
- Υπόλοιπο πιστωτικής ή προπληρωμένης κάρτας.
- Κίνηση πιστωτικής κάρτας.
- Κίνηση προπληρωμένης κάρτας.

c) Δάνεια και υπεραναλήψεις
- Υπόλοιπο δανείου.
- Κίνηση δανείου.
- Συνδεδεμένοι λογαριασμοί.
- Όρια και δεσμεύσεις.

d) Ηλεκτρονικές Συναλλαγές
- Ημερομηνία
- Λογαριασμοί χρέωσης ή πίστωσης
- Αιτιολογία

e) Ασφαλιστικά προϊόντα
- Είδος ασφαλιστικού προϊόντος.
- Ημερομηνία έναρξης και λήξης του συμβολαίου.
- Καταβληθέντα ποσά ασφάλισης.
- Καταβληθέντα ποσά επένδυσης.

f) Επενδυτικά προϊόντα
- Συνολική θέση.
- Κινήσεις παραγώγων (μόνο με e-Banking).
- Κατάσταση ημερήσιων εντολών μετοχών.
- Κατάσταση αμοιβαίων κεφαλαίων (μόνο με e-Banking).
- Συγκριτικά γραφήματα επενδυτικών προϊόντων (μόνο με e-Banking).
- Ενημέρωση επίσης για το ημερήσιο κλείσιμο των αμοιβαίων κεφαλαίων.

g) Επιταγές και εγγυητικές επιστολές

- Πληροφορίες κατάθεσης πολλαπλών επιταγών Eurobank και άλλων τραπεζών.
- Πληροφορίες και κατάσταση μεμονωμένων επιταγών (και σε ενέχυρο).
- Πληροφορίες για εγγυητικές επιστολές.

h) Θυρίδες
- Αριθμός συμβολαίου
- Ημερομηνία ανοίγματος
- Κάτοχος

3.4.2 Μεταφορές

- Μεταφορές χρημάτων σε λογαριασμούς Eurobank ή άλλων τραπεζών.
- Εμβάσματα σε τράπεζες του εξωτερικού.
- Πληρωμές καρτών ή οφειλών.
- Εντολές αγοράς ή πώλησης μετοχών και αμοιβαίων κεφαλαίων.
- Μεταφορές χρημάτων μέσω της υπηρεσίας PaF Payments.
- Πληρωμές μέσω της υπηρεσίας MyBank.

3.4.3 Πληρωμές

a) Δάνεια και κάρτες
- Πληρωμή δόσης δανείου Eurobank.
- Πληρωμή πιστωτικής κάρτας Eurobank, δική σας ή άλλου.
- Πληρωμή πιστωτικής κάρτας άλλης τράπεζας.
- Φόρτιση ή επαναφόρτιση προπληρωμένης κάρτας σας στην Eurobank, δικής σας ή άλλου.

b) Οφειλές Δημοσίου
- Πληρωμή οφειλών δημοσίου, όπως φόρος εισοδήματος, ΦΠΑ, ΕΝΦΙΑ κ.λπ.
- Πληρωμή ασφαλιστικών ταμείων.
- Πληρωμή συνδρομής σε επιμελητήρια.
- Έκδοση e-παραβόλου (μόνο με e-Banking).

c) Λογαριασμοί ΔΕΚΟ
- Πληρωμή λογαριασμών ενέργειας.
- Πληρωμή λογαριασμών σταθερής ή κινητής τηλεφωνίας.
- Πληρωμή λογαριασμών ύδρευσης και δήμων.

d) Άλλες πληρωμές
   - Πληρωμή ασφαλίστρων σε ιδιωτικές ασφαλιστικές εταιρείες.
   - Πληρωμή λογαριασμών εταιρειών φύλαξης.
   - Πληρωμή άλλων λογαριασμών (π.χ. ιδιωτικά σχολεία, e-Pass Αττικής Οδού).

3.4.4 Online προϊόντα και αιτήσεις

- Προπληρωμένη κάρτα e-Prepaid VISA
- Λογαριασμός Live
- Προθεσμιακή Για Όλους Live – Απλή
- Προθεσμιακή Για Όλους Live Με €πιστροφή
- Ασφάλεια αυτοκινήτου Safe Drive Live
- Ενεργοποίηση ειδοποιήσεων
- Ανάκτηση κωδικών e-Banking

3.5 Τράπεζα Πειραιώς

Η Τράπεζα Πειραιώς ιδρύθηκε το 1916 και έχει έδρα την Αθήνα. Για πολλές δεκαετίες λειτούργησε ως ιδιωτική Τράπεζα και το 1975 πέρασε υπό κρατικό έλεγχο, όπου και παρέμεινε μέχρι το 1991. Από τον Δεκέμβριο του 1991 που ιδιωτικοποιήθηκε έχει παρουσιάσει μεγάλη ανάπτυξη εργασιών, μεγεθών και δραστηριοτήτων. Ο Όμιλος της Τράπεζας Πειραιώς απασχολεί συνολικά 24.495 εργαζομένους, ενώ το σύνολο του δικτύου καταστημάτων αριθμεί 1.653 μονάδες, με παρουσία σε 10 χώρες συμπεριλαμβανομένης της Ελλάδας. Στην Ελλάδα με 30% μερίδιο αγοράς στα δάνεια και 29% στις καταθέσεις είναι η μεγαλύτερη τράπεζα της χώρας. Διεθνώς έχει παρουσία σε περιοχές της Νοτιοδυτικής Ευρώπης και την Ανατολική Μεσόγειο, καθώς και στα οικονομικά κέντρα του Λονδίνου της Νέας Υόρκης και της Φρανκφούρτης.

Σήμερα, μετά τις εξαγορές της «υγιούς» ATEbank, της Γενικής Τράπεζας, των εγχώριων τραπεζικών δραστηριοτήτων των Τράπεζας Κύπρου, Cyprus Popular Bank και Ελληνικής Τράπεζας αλλά και την εξαγορά της Millennium Bank Ελλάδας, το συνολικό ενεργητικό του Ομίλου Πειραιώς φτάνει τα €93 δισ, οι χορηγήσεις μετά από προβλέψεις τα €62 δισ και οι καταθέσεις πελατών τα €55 δισ (στοιχεία Σεπτεμβρίου 2013). [25]

Η Τράπεζα Πειραιώς προσφέρει υπηρεσίες στους εξής τομείς: τραπεζικές υπηρεσίες απευθείας προς τον καταναλωτή, μικρές και μεσαίου μεγέθους επιχειρήσεις, αγροτική πίστη, αγορές κεφαλαίου, μισθώσεις (leasing), κτηματομεσιτικές υπηρεσίες, χρηματοδοτήσεις του ναυτιλιακού τομέα, πράσινη τραπεζική, ηλεκτρονική τραπεζική.

Οι υπηρεσίες online banking που υποστηρίζει η Τράπεζα Πειραιώς είναι οι εξής[24]:

3.5.1 Λογαριασμούς

- Υπόλοιπο - Κινήσεις
- Διαχείριση Alerts για λογαριασμούς
- Πληροφορίες - Στοιχεία Λογαριασμού
- Αντίγραφα Λογαριασμών
- Αποστολή κινήσεων λογαριασμών με email
- Άνοιγμα νέου λογαριασμού
- Αποστολή IBAN Λογαριασμού

3.5.2 Προθεσμιακές Καταθέσεις

- Πληροφορίες - Στοιχεία Προθεσμιακής
- Αίτηση Νέας Προθεσμιακής
- Διαχείριση Προθεσμιακής

3.5.3 Πιστωτικές Κάρτες

- Υπόλοιπο - Κινήσεις
- Διαχείριση Alerts
- Πληρωμή Πιστωτικής Κάρτας Τράπεζας Πειραιώς
- Πληρωμή Πιστωτικής Κάρτας Άλλης Ελληνικής Τράπεζας
- Ανάθεση/Διαχείριση/Ακύρωση Πάγιας Εντολής Πληρωμής
- Μεταφορά υπολοίπου
- Έκδοση Νέας Κάρτας Τράπεζας Πειραιώς
- Αλλαγή PIN ή Επανέκδοση PIN
- Ηλεκτρονικό Αρχείο Μηνιαίων Λογαριασμών (e-statements)
- Εμφάνιση Μηνιαίων Λογαριασμών
- Αποστολή κινήσεων λογαριασμών με email
- Προσωρινή Απενεργοποίηση/Ενεργοποίηση καρτών

3.5.4 Χρεωστικές Κάρτες

- Υπόλοιπο –Κινήσεις
- Μεταβολή Ημερησίου Ορίου Χρήσης
- Σύνδεση/Αποσύνδεση Λογαριασμού
- Αναλυτικά Στοιχεία Χρεωστικής
- Αίτηση Αλλαγής Κάρτας Λόγω Φθοράς

- Αλλαγή PIN ή Επανέκδοση PIN
- Προσωρινή Απενεργοποίηση/Ενεργοποίηση καρτών

3.5.5 Προπληρωμένες και επαναφορτιζόμενες κάρτες

- Έκδοση/Επανέκδοση Νέας Κάρτας
- Φόρτιση Κάρτας μου ή Τρίτου
- Εκφόρτιση Κάρτας ή Ανάληψη Μετρητών
- Υπόλοιπο - Κινήσεις
- Διαχείριση Alerts
- Μεταβολή Ημερησίου Ορίου Συναλλαγών
- Ακύρωση Κάρτας
- Προσωρινή Απενεργοποίηση/Ενεργοποίηση καρτών

a) Δάνεια

- e-Loan by winbank (με πρόγραμμα επιβράβευσης yellow)
- Αναλυτικά Στοιχεία Δανείου
- Πλάνο αποπληρωμής Δανείου
- Πληρωμή Δόσης Δανείου
- Ανάθεση/Διαχείριση/Ακύρωση Πάγιας Εντολής Πληρωμής

3.5.6 Επιταγές

- Πληροφόρηση για Επιταγές που έχω εκδώσει
- Αίτηση Ανάκλησης/Ακύρωση Επιταγής
- Αίτηση Έκδοσης νέου Καρνέ
- Ιστορικό Αρχείο Επιταγών (Συνολική απεικόνιση)

3.5.7 Χρηματιστήριο

- Ενημέρωση για την αποτίμηση του χαρτοφυλακίου μου
- Διαχείριση Alerts (εκτέλεσης αγοραπωλησίας μετοχών, καθημερινή αποτίμηση χαρτοφυλακίου)
- Αγορά/Πώληση Μετοχών
- Αίτηση συμμετοχής σε δημόσιες εγγραφές
- Αντίγραφα Κινήσεων Χαρτοφυλακίου Μετοχών

3.5.8 Πληρωμές

- ΔΕΚΟ, Δημ. Υπηρεσίες, τηλεπικοινωνίες
- Ιστορικό Πληρωμών

3.5.9 Λεφτά στο Λεπτό

- Αποστολή/Καταχώριση εντολής για παραλαβή μετρητών χωρίς κάρτα
- Λήψη μετρητών χωρίς τη χρήση κάρτας
- Ιστορικό συναλλαγών ΛσΛ

3.5.10 Άλλες υπηρεσίες

- Έκδοση και Πληρωμή e-Παραβόλου
- Ανανέωση αερόχρονου για καρτοκινητή
- winbank Επαναφόρτιση e-PASS Αττικής Οδού
- Έκδοση εργοσήμου
- Εγγραφή στο πρόγραμμα Επιβράβευσης yellow
- Αγορά Πακέτου Πληρωμής Λογαριασμών «ΕξόφΛΥΣΗ»
- Δαπάνες Μείωσης φόρου

Κεφάλαιο 4

4.1 Εφαρμογή Διασύνδεσης με πλατφόρμα rAPId LINK

Η εφαρμογή που παρουσιάζεται στην παρούσα εργασία έχει σκοπό να παρέχει στους χρήστες ένα εναλλακτικό μέσο επικοινωνίας με τις τράπεζες, διαφορετικό από το παραδοσιακό e-banking. Για το λόγο αυτό θα χρησιμοποιήσουμε ένα σύνολο από REST APIs για να επιτύχουμε την επικοινωνία με την τράπεζα. Στα πλαίσια αυτής της εργασίας θα παρουσιαστεί η διασύνδεση με την Τράπεζα Πειραιώς μέσω της πλατφόρμας που παρέχει και η οποία ονομάζεται rAPId LINK.

Στο παρόν κεφάλαιο θα γίνει παρουσίαση της πλατφόρμας και των τεχνολογιών που χρησιμοποιήθηκαν για την ολοκλήρωση της εφαρμογής.

4.2 Παρουσίαση πλατφόρμας rAPId LINK

Εικόνα 26-rAPId LINK logo

Η πλατφόρμα rAPId LINK που υλοποιήθηκε από την Τράπεζα Πειραιώς, καθιστά εφικτή την άμεση πρόσβαση των πελατών της στα τραπεζικά δεδομένα τους και την εκτέλεση τραπεζικών συναλλαγών, μέσω εφαρμογών λογισμικού τρίτων συνεργαζόμενων εταιριών.

Η πλατφόρμα δίνει τη δυνατότητα σε εταιρίες λογισμικού που διαθέτουν εφαρμογές σε πελάτες τους (για παράδειγμα συστήματα ERP), να διασυνδέσουν αυτές σε «πραγματικό χρόνο» με τα συστήματα της Τράπεζας, διατηρώντας πάντοτε τα υψηλότερα επίπεδα ασφάλειας για τον πελάτη. Η διαθέσιμη λειτουργικότητα της νέας πλατφόρμας συγκεντρώνεται στις παρακάτω βασικές κατηγορίες:

- Προσωπικές Πληροφορίες

- Χαρτοφυλάκιο Προϊόντων
- Καταθετικοί Λογαριασμοί
- Πιστωτικές/Προπληρωμένες/Χρεωστικές Κάρτες
- Εκτελέσεις Μεταφορών / Εμβασμάτων
- Εκτελέσεις Μαζικών Εμβασμάτων / Μισθοδοσιών
- Πληρωμές Λογαριασμών (ΔΕΗ, ΟΤΕ, ΕΥΔΑΠ, κλπ)
- Γενικές Πληροφορίες Δικτύου Καταστημάτων και ATM, APS

και θα εμπλουτίζεται διαρκώς με νέες υπηρεσίες και συναλλαγές. Οι λειτουργίες αυτές είναι πάντοτε κάτω από τον ασφαλή και απόλυτο έλεγχο του ίδιου του πελάτη.

Η πλατφόρμα βρίσκεται σε παραγωγική λειτουργία και ήδη μεγάλες ελληνικές εταιρείες παραγωγής λογισμικού, όπως οι Entersoft, EpsilonNet, SoftOne, Singular Logic, Unisoft και άλλες, έχουν υλοποιήσει τη διασύνδεση με τα APIs της πλατφόρμας και έχουν ενσωματώσει τις διαθέσιμες λειτουργικότητες στα συστήματά τους. Μπορούν έτσι να προσφέρουν στους πελάτες τους που είναι και πελάτες της Τράπεζας Πειραιώς τη νέα εμπειρία τραπεζικών συναλλαγών

Σχετικά με την αρχιτεκτονική της πλατφόρμας, είναι βασισμένη στην αρχιτεκτονική Open Banking. Το Open Banking είναι όρος σχετικός με οικονομικές υπηρεσίες και αποτελεί κομμάτι του FinTech. Ο όρος αυτός περιλαμβάνει :

- Την χρήση Open APIs που επιτρέπουν σε τρίτες εταιρίες να αναπτύξουν εφαρμογές και υπηρεσίες γύρω από τα χρηματοπιστωτικά ιδρύματα
- Μεγαλύτερη διαφάνεια για τους κατόχους λογαριασμών ως προς τις πληροφορίες στις οποίες έχουν πρόσβαση και οι οποίες εκτείνονται από Open Data μέχρι πιο προσωπικά δεδομένα.
- Την χρήση open source τεχνολογιών για να επιτευχθούν τα παραπάνω.

Η αφορμή που αποτέλεσε την αιτία για να αναπτυχθεί το Open Banking ήταν ο κανονισμός Payments Services Directive (PSD2), ο οποίος υποχρέωνε τις τράπεζες να δώσουν πρόσβαση στους πελάτες τους στα δεδομένα τους μέσω τρίτων παρόχων.

4.2.1 Παρουσίαση API

Η πλατφόρμα rAPId LINK δίνει πρόσβαση στις εξής κατηγορίες APIs

- API Accounts

- API Cards
- API Customers
- API General Info
- API Transactions

Ας δούμε αναλυτικά ποια services υποστηρίζει κάθε API.

4.2.1.1 API Accounts

- GET /

Επιστρέφει όλους τους λογαριασμούς και τα υπόλοιπα του χρήστη

- GET /{Iban}/info

Επιστρέφει πληροφορίες για έναν λογαριασμό, όχι αναγκαστικά του χρήστη με βάση το IBAN του

- GET /{accountId}/details

Επιστρέφει πληροφορίες για τον λογαριασμό του χρήστη με βάση το μοναδικό id του λογαριασμού

- GET /{accountId}/transactions/{input_filter}

Επιστρέφει τις κινήσεις που πραγματοποιήθηκαν σε αυτόν τον λογαριασμό. Τα φίλτρα περιλαμβάνουν ημερομηνία από (fromDate) , ημερομηνία έως(toDate), πλήθος εγγραφών ανά κλήση(pageSize) και από ποια εγγραφή και μετά να επιστρέψει το service(fromRow) .

4.2.1.2 API General Info

- GET /toIban/{accountNumber}

Μετατρέπει το IBAN ενός λογαριασμού της Τράπεζας Πειραιώς, σε αριθμό λογαριασμού της τράπεζας

- GET /verifyIban/{Iban}

Επιβεβαιώνει την εγκυρότητα ενός IBAN

- GET /countries

Επιστρέφει λίστα με χώρες

- GET /pointsOfPresence

Επιστρέφει λίστα με πληροφορίες για όλα τα σημεία που έχει παρουσία η Τράπεζα Πειραιώς, όπως ATM, υποκαταστήματα κτλ.

- GET /pointsOfPresence/{PointTypeName}

Επιστρέφει λίστα με πληροφορίες για όλα τα σημεία ενός συγκεκριμένου τύπου πχ όλα τα ATM.

4.2.1.3 API Cards

- GET /

Επιστρέφει όλες τις κάρτες του χρήστη (χρεωστικές, πιστωτικές, προπληρωμένες).

- GET /credit

Επιστρέφει τις πιστωτικές κάρτες του χρήστης.

- GET /debit

Επιστρέφει τις χρεωστικές κάρτες του χρήστη.

- GET /prepaid

Επιστρέφει τις προπληρωμένες κάρτες του χρήστη.

- GET /{cardId}/details

Επιστρέφει πληροφορίες για συγκεκριμένη κάρτα.

- GET /{cardId}/transactions/{input_filter}

Επιστρέφει τις κινήσεις για μια συγκεκριμένη κάρτα.

- GET /credit/{cardId}/details

Επιστρέφει πληροφορίες για μια συγκεκριμένη πιστωτική κάρτα

- GET /credit/{cardId}/transactions/{input_filter}

Επιστρέφει τις κινήσεις για μια συγκεκριμένη πιστωτική κάρτα

- GET /debit/{cardId}/details

Επιστρέφει πληροφορίες για μια συγκεκριμένη χρεωστική κάρτα

- GET /debit/{cardId}/transactions/{input_filter}

Επιστρέφει τις κινήσεις για μια συγκεκριμένη χρεωστική κάρτα

- GET /prepaid/{cardId}/details

Επιστρέφει πληροφορίες για μια συγκεκριμένη προπληρωμένη κάρτα

- GET /prepaid/{cardId}/transactions/{input_filter}

Επιστρέφει τις κινήσεις για μια συγκεκριμένη προπληρωμένη κάρτα.

4.2.1.4 API Transactions

- GET /payroll/{PaymentCode}/details

Επιστρέφει πληροφορίες για μια συγκεκριμένη μισθοδοσία με βάση τον κωδικό πληρωμής.

- GET /payroll/history/{input_filter}

Επιστρέφει ιστορικό προηγούμενων πληρωμών μισθοδοσίας.

- GET /bulkPayment/{PaymentCode}/{PaymentDate}/details

Επιστρέφει πληροφορίες για μια συγκεκριμένη μαζική πληρωμή με βάση τον κωδικό πληρωμής.

- GET /bulkPayment/history/{input_filter}

Επιστρέφει ιστορικό προηγούμενων μαζικών πληρωμών.

- POST /payroll

Εκτέλεση πληρωμής μισθοδοσίας. Οι εγγραφές μπορούν να είναι είτε σε αρχείο είτε σε json

- POST /payroll/validate

Επιστρέφει αν είναι έγκυρη η μισθοδοσία ώστε να εκτελεστεί , τα έξοδα που θα προκύψουν καθώς επίσης κι ένα κλειδί για να εκτελεστεί η κίνηση

- POST /payroll/execute/{SessionKey}

Εκτελεί την μισθοδοσία με τον κλειδί που επέστρεψε το προηγούμενο service

- POST /bulkPayment

Εκτέλεση μαζικών πληρωμών. Οι εγγραφές μπορούν να είναι είτε σε αρχείο είτε σε json

- POST /bulkPayment/validate

Επιστρέφει αν είναι έγκυρη η μαζική πληρωμή ώστε να εκτελεστεί , τα έξοδα που θα προκύψουν καθώς επίσης κι ένα κλειδί για να εκτελεστεί η κίνηση

- POST /bulkPayment/execute/{SessionKey}

Εκτελεί την μαζική πληρωμή με τον κλειδί που επέστρεψε το προηγούμενο service.

- POST /transferToIban

Εκτέλεση πληρωμής

- POST /transferToIban/validate

Επιστρέφει αν είναι έγκυρη η πληρωμή ώστε να εκτελεστεί , τα έξοδα που θα προκύψουν καθώς επίσης κι ένα κλειδί για να εκτελεστεί η κίνηση

- POST /transferToIban/execute/{SessionKey}

Εκτελεί την πληρωμή με τον κλειδί που επέστρεψε το προηγούμενο service.

4.2.1.5 API Customers

- GET /info

Επιστρέφει πληροφορίες για τον χρήστη που έχει συνδεθεί.

4.2.2 Αυθεντικοποίηση – Authorization(OAuth 2)

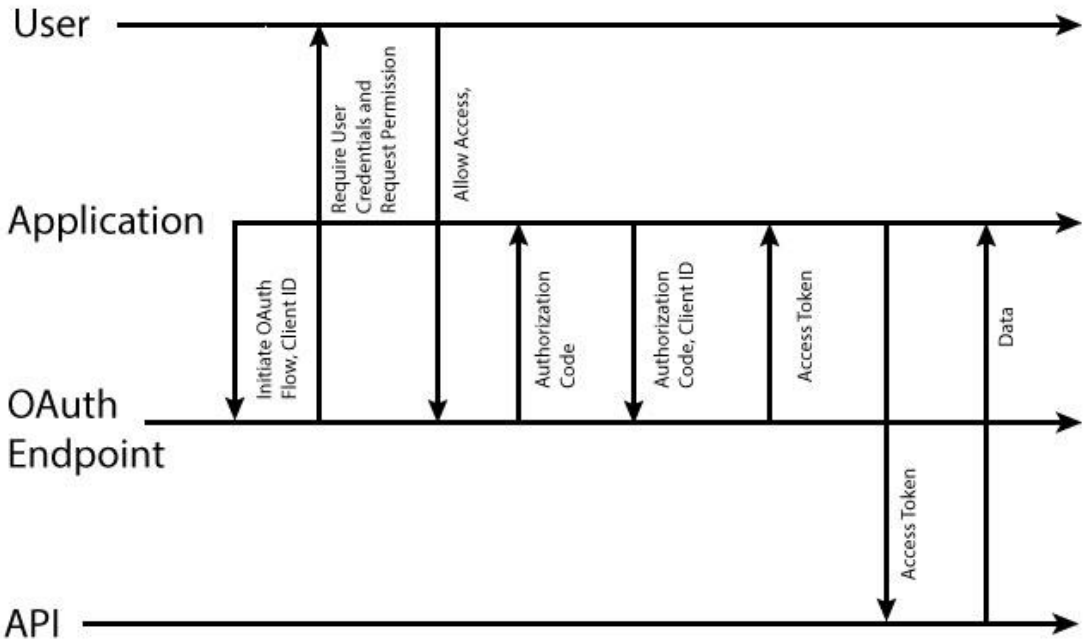
Για την κλήση των περισσότερων APIs που αναφέρθηκαν στην προηγούμενη ενότητα χρειάζεται να ακολουθηθεί ένα flow αυθεντικοποίησης που βασίζεται στο OAuth 2

πρωτόκολλο. Αυτή η διαδικασία πιστοποιεί τόσο εμάς(3rd party παρόχους) όσο και τους πελάτες και καθιστά σαφές ότι δεν θα αποθηκευτούν τα στοιχεία σύνδεσης του πελάτη (username, password) στην εφαρμογή μας. Με αυτό τον τρόπο παρέχει ασφάλεια και στις 2 πλευρές.

Το flow που αναφέραμε, επιτυγχάνεται με τα επόμενα 3 βήματα:

- Κλήση του Authorization URL , το οποίο είναι της μορφής **https://oauth.piraeusbank.gr/oauth/connect/authorize?response_type=code&client_id=APPLICATION_ID&redirect_uri=APPLICATION_REDIRECT_URI&scope=API_SCOPE** και επιστρέφει ένα code που θα χρησιμποποιηθεί στο επόμενο βήμα.
- Κλήση του Token URL με τον κωδικό που πήραμε πριν. Το token url είναι της μορφής **https://oauth.piraeusbank.gr/oauth/connect/token** και επιστρέφει το access token.
- Κλήση api με την χρήση του access token που πήραμε στο προηγούμενο βήμα

Στην φωτογραφία που ακολουθεί βλέπουμε το flow που περιγράψαμε



Εικόνα 27-Authorization flow

4.2.3 Second Factor Authentication

Σε όλες τις κλήσεις που περιλαμβάνουν αποστολή χρημάτων σε λογαριασμό που δεν είναι του χρήστη, υπάρχει κι ένα δεύτερο βήμα ασφάλειας που ονομάζεται Second Factor Authentication. Με αυτό το επιπρόσθετο βήμα, την πρώτη φορά μετά το login που μια χρηματική κίνηση πρέπει να πραγματοποιηθεί, ο χρήστης θα πρέπει να δώσει ένα δεύτερο pin μιας χρήσης.

Υπάρχουν 2 μέθοδοι για να επιτευχθεί αυτό

- Μέσω μιας συσκευής που παράγει pins. Η συσκευή αυτή παρέχεται από την τράπεζα και το pin που παράγει ονομάζεται OTP(one time pin)
- Μέσω sms

Η διαδικασία αυτή γίνεται μία φορά ανά session (login). Μόλις επιβεβαιωθεί το pin το session του χρήστη μπαίνει σε elevated security mode. Στη συνέχεια όσο ο χρήστης παραμένει συνδεδεμένος, δεν θα ξαναζητηθεί pin.

4.3 Άλλες τεχνολογίες που χρησιμοποιήθηκαν

4.3.1 HTML

Η HTML (αρχικοποίηση του αγγλικού HyperText Markup Language) είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες, και τα στοιχεία της είναι τα βασικά δομικά στοιχεία των ιστοσελίδων.[28]

Η HTML γράφεται υπό μορφή στοιχείων HTML τα οποία αποτελούνται από ετικέτες (tags), οι οποίες περικλείονται μέσα σε σύμβολα «μεγαλύτερο από» (>) και «μικρότερο από»(<) (για παράδειγμα <html>), μέσα στο περιεχόμενο της ιστοσελίδας. Οι ετικέτες HTML συνήθως λειτουργούν ανά ζεύγη (για παράδειγμα <h1> και </h1>), με την πρώτη να ονομάζεται ετικέτα έναρξης και τη δεύτερη ετικέτα λήξης (ή σε άλλες περιπτώσεις ετικέτα ανοίγματος και ετικέτα κλεισίματος αντίστοιχα).Υπάρχουν περιπτώσεις που αυτό δεν είναι απαραίτητο (
) Ανάμεσα στις ετικέτες, οι σχεδιαστές ιστοσελίδων μπορούν να τοποθετήσουν κείμενο, πίνακες, εικόνες κλπ.

4.3.2 JavaScript –Sencha

Η JavaScript (JS) είναι διερμηνευμένη γλώσσα προγραμματισμού για ηλεκτρονικούς υπολογιστές. Αρχικά αποτέλεσε μέρος της υλοποίησης των φυλλομετρητών Ιστού, ώστε τα σενάρια από την πλευρά του πελάτη (client-side scripts) να μπορούν να

επικοινωνούν με τον χρήστη, να ανταλλάσσουν δεδομένα ασύγχρονα και να αλλάζουν δυναμικά το περιεχόμενο του εγγράφου που εμφανίζεται.[31]

Πλεονεκτήματα JavaScript

- Δεν χρειάζεται εγκατάσταση. Περιέχεται σε όλους τους σύγχρονους browsers.
- Δεν επιβαρύνει τον server με περιττούς υπολογισμούς που μπορούν να γίνουν client-side
- Εύκολη στην υλοποίηση και την εκμάθηση
- Είναι γρήγορη γιατί δεν απαιτεί την επικοινωνία με server.
- Συνεργάζεται εύκολα με πολλές γλώσσες και μπορεί να χρησιμοποιηθεί μέσα σε άλλα γλώσσες χωρίς δυσκολία

Μειονεκτήματα JavaScript

- Δεν είναι απόλυτα ασφαλής. Επειδή εκτελείται από την πλευρά του πελάτη είναι ευάλωτη σε κακοήθεις επιθέσεις.
- Αξιοπιστία στον τελικό χρήστη: Η JavaScript ερμηνεύεται μερικές φορές διαφορετικά από διαφορετικά προγράμματα περιήγησης. Ενώ τα scripts στον διακομιστή θα παράγουν πάντα την ίδια έξοδο, τα scripts στον πελάτη μπορεί να είναι λίγο απρόβλεπτα.[30]

Αυτή την στιγμή η JavaScript είναι μία από τις top-10 σε δημοφιλία γλώσσες. Αυτό έχει ως αποτέλεσμα να υπάρχουν πολλές εκδόσεις βιβλιοθηκών που ουσιαστικά επεκτείνουν τις δυνατότητες της γλώσσας. Μία τέτοια βιβλιοθήκη είναι η ExtJS, την οποία θα χρησιμοποιήσουμε στην εφαρμογή. Με αυτή την βιβλιοθήκη μπορούμε να δημιουργήσουμε single page εφαρμογές. Το μεγάλο της πλεονέκτημα είναι ότι μας παρέχει έτοιμα components (πχ toolbar,grids) που μας γλιτώνουν από το να χάσουμε χρόνο σχεδιάζοντας τα με απλή html.[29]

4.3.3 CSS- Bootstrap

Με την css [32]ουσιαστικά μορφοποιούμε την εμφάνιση της HTML σελίδας. Ένα αρχείο .css περιέχει κανόνες που ορίζουν πως θα φαίνεται κάθε αντικείμενο της HTML. Μαζί με την HTML και την JavaScript αποτελούν θεμελιώδεις δομές του World Wide Web.

Η Bootstrap είναι μια βιβλιοθήκη που αναπτύχθηκε από το Twitter. Το βασικό του πλεονέκτημα είναι η δομή grid που βοηθάει στο σχεδιασμό σελίδων για οθόνες με διαφορετικές αναλύσεις.

4.3.4 C#

Η C# είναι μια γλώσσα προγραμματισμού Η/Υ. Δημιουργήθηκε από την Microsoft μέσα από την πλατφόρμα .NET και αργότερα αναγνωρίστηκε επισήμως από την Ecma (ECMA-334) και την ISO (ISO/IEC 2327:2006). Είναι μια από τις γλώσσες προγραμματισμού που δημιουργήθηκαν για την Common Language Infrastructure. Ο κύριος σκοπός της γλώσσας είναι να είναι απλή αντικειμενοστραφής γλώσσα για γενική χρήση. Ο διοικητής της ομάδα που διαχειρίζεται την γλώσσα ονομάζεται Anders Hejlsberg. Στις 7 Μαΐου 2018 κυκλοφόρησε η έκδοση 7.2 η οποία είναι η πιο πρόσφατη μέχρι σήμερα

4.3.5 IIS

Ο IIS είναι ένας web server που έρχεται προ εγκατεστημένος στα Windows. Θα τον χρησιμοποιήσουμε για να στήσουμε το project μας.[33]

4.3.6 REST

Το REST (REpresentational State Transfer) είναι ένα στυλ αρχιτεκτονικής για την ανάπτυξη web services. Η REST είναι διάσημη για την απλότητά της και γιατί στηρίζεται σε υπάρχοντα συστήματα και χαρακτηριστικά της HTTP για να επιτύχει τους στόχους της.[34]

Οι REST εφαρμογές δίνουν τη δυνατότητα μέσω του HTTP να δημιουργηθούν δεδομένα, να ανανεωθούν να διαγραφούν ή να διαβαστούν. Με μία λέξη οι ενέργειες αυτές θα μπορούσαν να ονομαστούν CRUD (Create/Read/Update/Delete). Η αντιστοίχιση με τις http μεθόδους φαίνεται στην παρακάτω εικόνα.

HTTP Verb	CRUD
POST	Create
GET	Read
PUT	Update/Replace
PATCH	Update/Modify
DELETE	Delete

Εικόνα 28-HTTP/CRUD

Πλεονεκτήματα

- Εύκολο για όποια κατέχει γνώσεις HTTP
- Η ασφάλεια επιτυγχάνεται με τα γνωστά SSL και TLS
- Είναι ανεξάρτητη από γλώσσες

Μειονεκτήματα

- Ισχύουν οι περιορισμοί που υπάρχουν και στην HTTP
- Δεν υποστηρίζει push notifications

4.3.7 Google Maps Platform

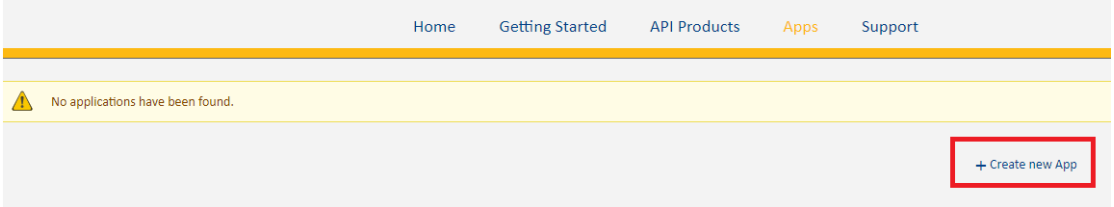
Το Google Maps Platform μας δίνει πρόσβαση στους χάρτες της Google, πάνω στους οποίους θα τοποθετήσουμε διάφορα σημεία ενδιαφέροντος.

Κεφάλαιο 5

Για να επιτύχουμε την διασύνδεση της εφαρμογής μας με την πλατφόρμα rAPId LINK, είναι απαραίτητο να δημιουργήσουμε ένα application ώστε να αποκτήσουμε client id και secret key τα οποία είναι απαραίτητα για τις κλήσεις μας.

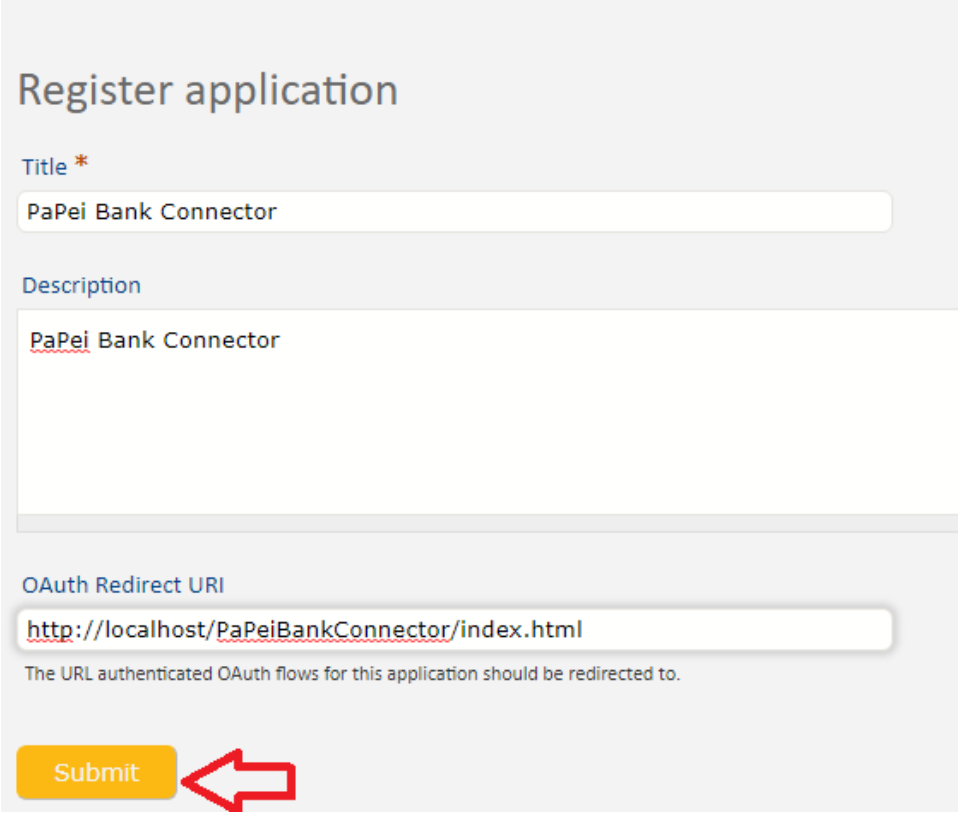
5.1 Δημιουργία application στο rAPId LINK

Βήμα 1: Πατάμε την επιλογή Apps για να μεταφερθούμε στη σελίδα με τις εφαρμογές μας. Εφόσον δεν έχουμε κάποια εφαρμογή πατάμε την επιλογή "+ Create new App"



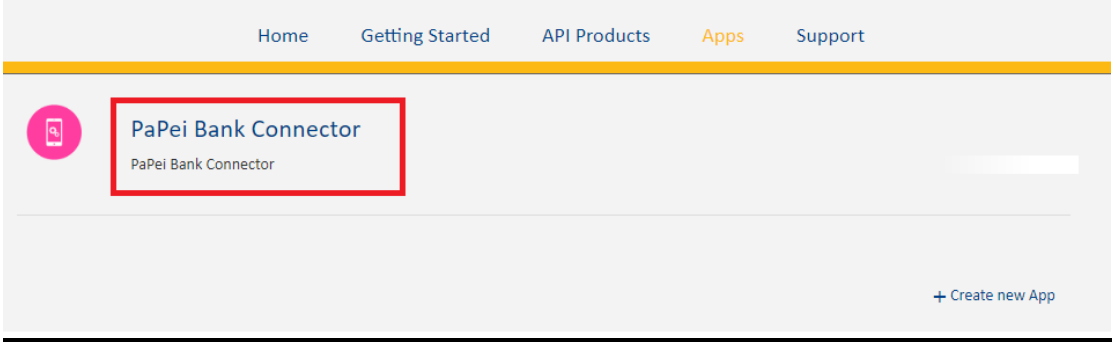
Εικόνα 29- Δημιουργία App Βήμα 1

Βήμα 2: Συμπληρώνουμε το όνομα της εφαρμογής, μια σύντομη περιγραφή και το url στο οποίο θα κάνει redirect μετά το login.



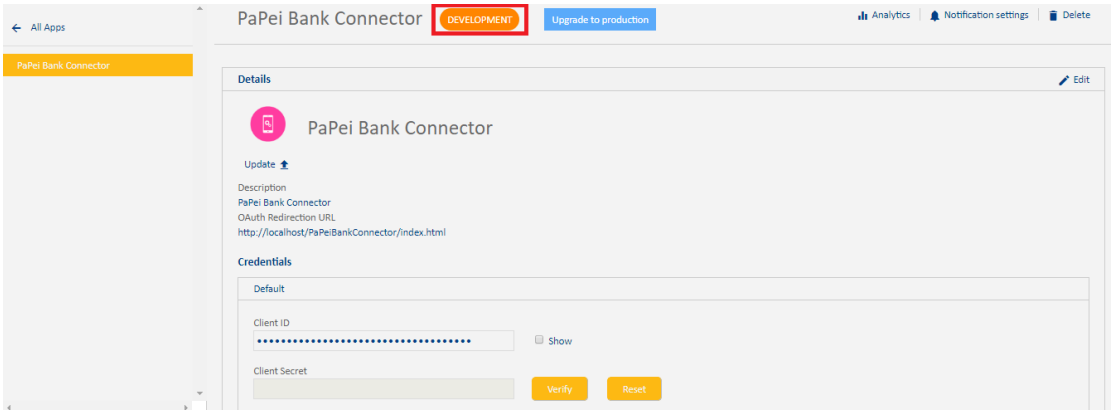
Εικόνα 30-Δημιουργία App Βήμα 2

Βήμα 3: Βλέπουμε ότι η εφαρμογή μας έχει δημιουργηθεί. Πατάμε για να δούμε λεπτομέρειες.



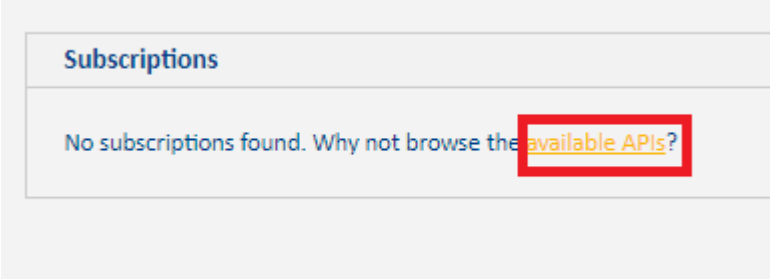
Εικόνα 31-Δημιουργία App Βήμα 3

Βήμα 4:Παρατηρούμε ότι είμαστε σε development mode. Αυτό σημαίνει ότι προς το παρόν χρησιμοποιούμε sandbox λογαριασμούς για τις εφαρμογές μας.



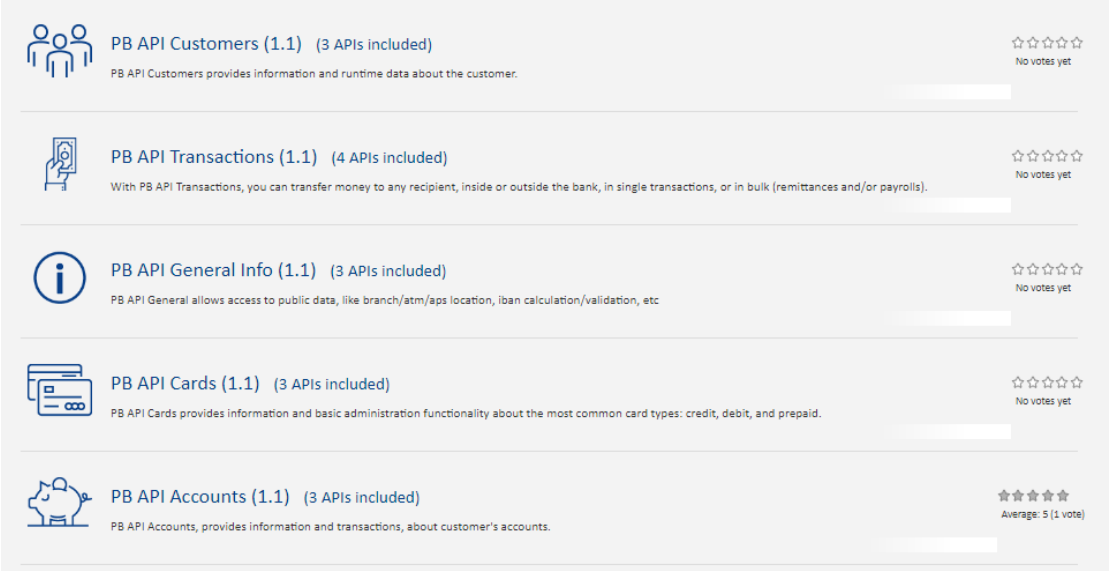
Εικόνα 32-Δημιουργία App Βήμα 4

Βήμα 5: Δεν έχουμε εγγράψει την εφαρμογή μας σε κάποιο API. Πατάμε για να δούμε τα διαθέσιμα και να εγγραφούμε σε αυτά.



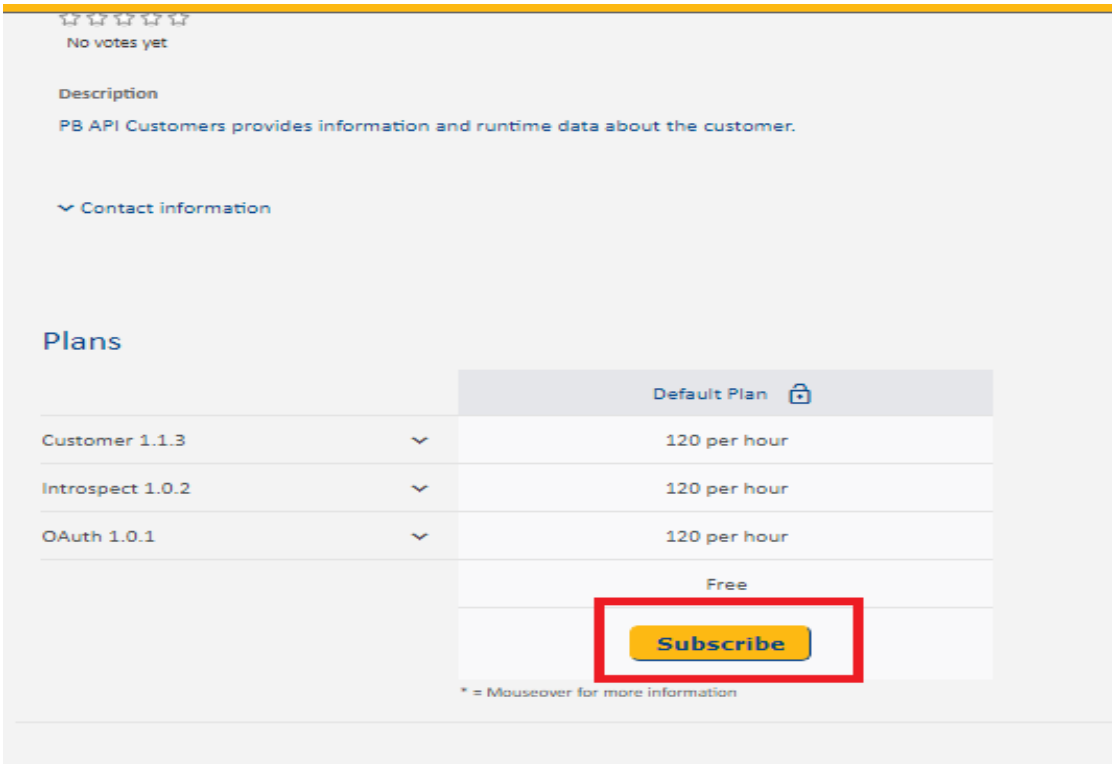
Εικόνα 33-Δημιουργία App Βήμα 5

Βήμα 6:Τα διαθέσιμα APIs. Πατάμε σε όσα θέλουμε να εγγραφούμε. Εμείς θα γραφτούμε σε όλα, αλλά θα δείξουμε την διαδικασία μόνο για το ένα. Ομοίως γίνεται και για τα υπόλοιπα.



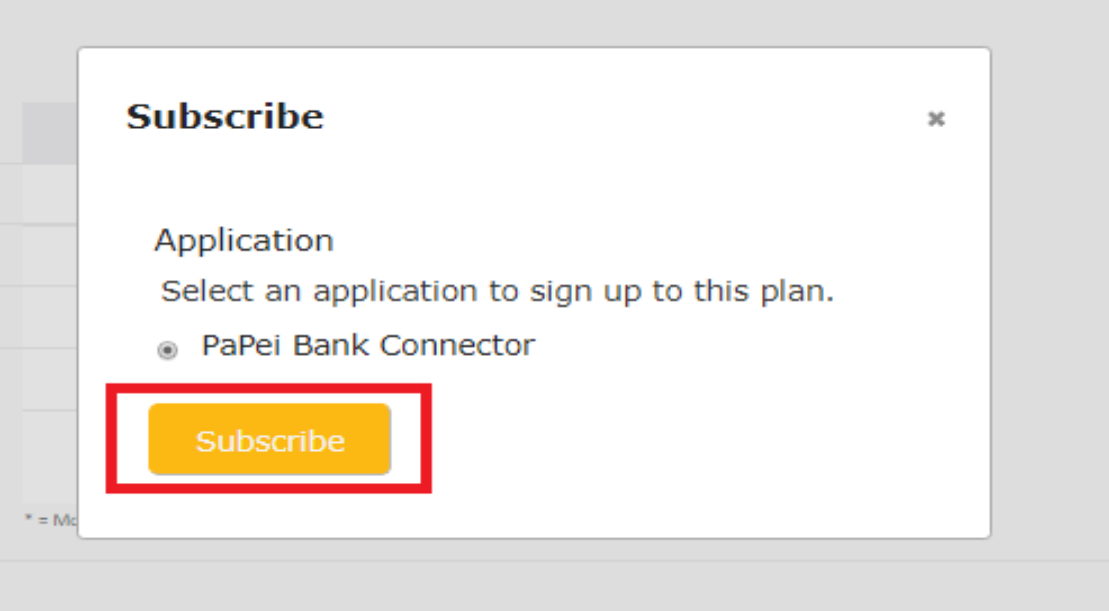
Εικόνα 34-Δημιουργία App Βήμα 6

Βήμα 7: Εγγραφή στο συγκεκριμένο API. Παρατηρούμε τα όρια που υπάρχουν στο πλήθος των κλήσεων που μπορεί να εκτελέσει η εφαρμογή μας ανά ώρα.



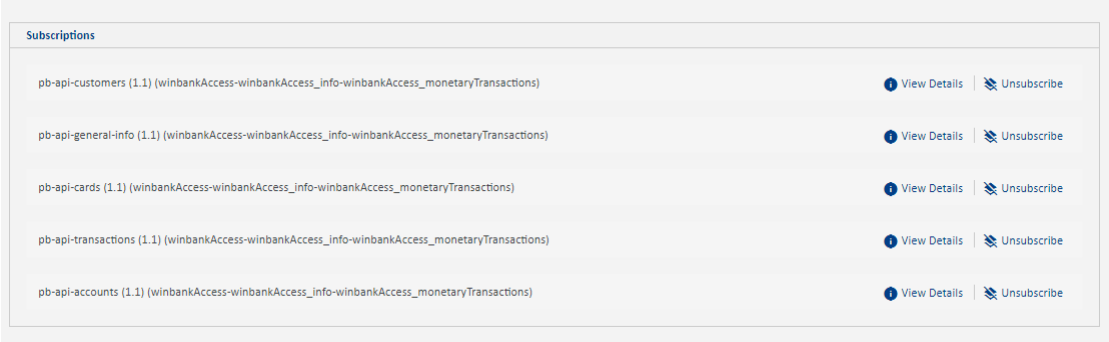
Εικόνα 35-Δημιουργία App Βήμα 7

Βήμα 8: Επιλογή εφαρμογής που θέλουμε να γραφτεί στο συγκεκριμένο API.



Εικόνα 36-Δημιουργία App Βήμα 8

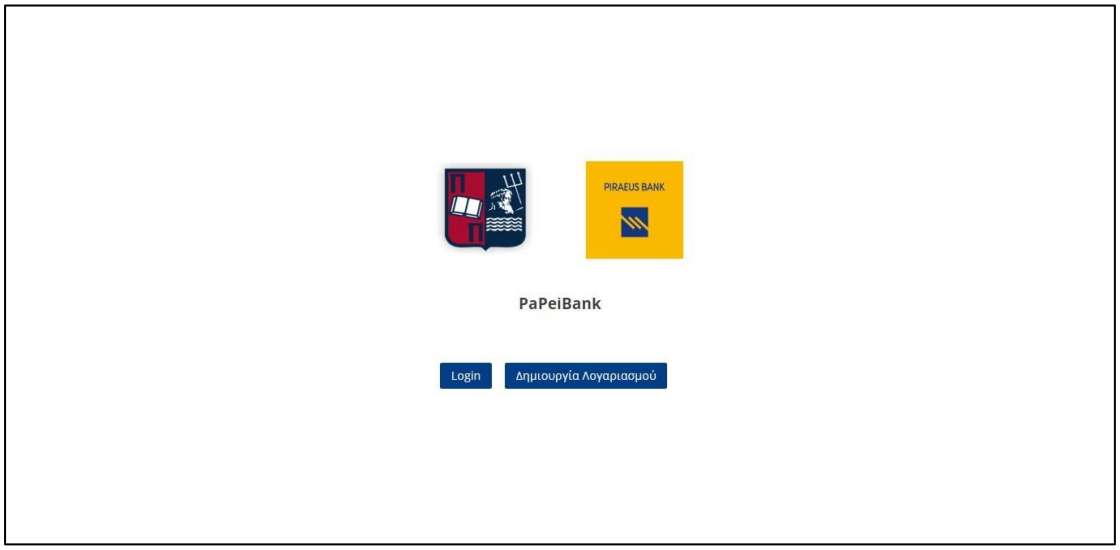
Βήμα 9: Ξαναγυρνάμε στις λεπτομέρειες της εφαρμογής μας και βλέπουμε όλα τα APIs στα οποία έχουμε γραφτεί. Τώρα είμαστε έτοιμοι να κάνουμε τις κλήσεις στην τράπεζα.



Εικόνα 37-Δημιουργία App Βήμα 9

5.2 Παρουσίαση εφαρμογής διασύνδεσης με Τραπεζικό API

5.2.1 Login



Εικόνα 38-Αρχική οθόνη login

Η αρχική οθόνη login αποτελείται από τα λογότυπα του Πανεπιστημίου Πειραιά και της τράπεζας Πειραιώς, από την επωνυμία της εφαρμογής μας και από 2 κουμπιά. Το πρώτο θα μας ανοίξει την φόρμα για το να κάνουμε login στο σύστημα και η δεύτερη θα μας κάνει redirect στο site της τράπεζας Πειραιώς , όπου θα βρούμε πληροφορίες για το πώς θα αποκτήσουμε λογαριασμό και κωδικούς e-banking στην τράπεζα.

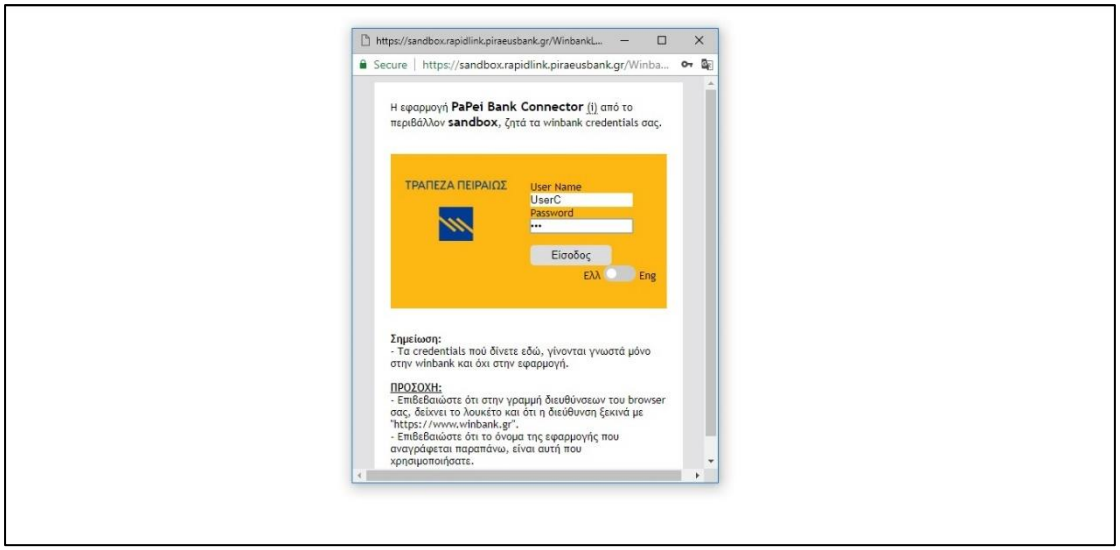
Για να κάνουμε login στην εφαρμογή είναι απαραίτητο να έχουμε :

- Τραπεζικό λογαριασμό στην τράπεζα Πειραιώς
- Κωδικούς e-banking

Στα πλαίσια της εργασίας η τράπεζα μας παρέχεις δοκιμαστικούς κωδικούς , οι οποίοι μας δίνουν πρόσβαση σε ένα sandbox περιβάλλον με fake data αλλά με πραγματική λειτουργικότητα. Για περισσότερες πληροφορίες σχετικά με το sandbox περιβάλλον μπορούμε να ανατρέξουμε στον παρακάτω σύνδεσμο:

https://rapidlink.piraeusbank.gr/node/241

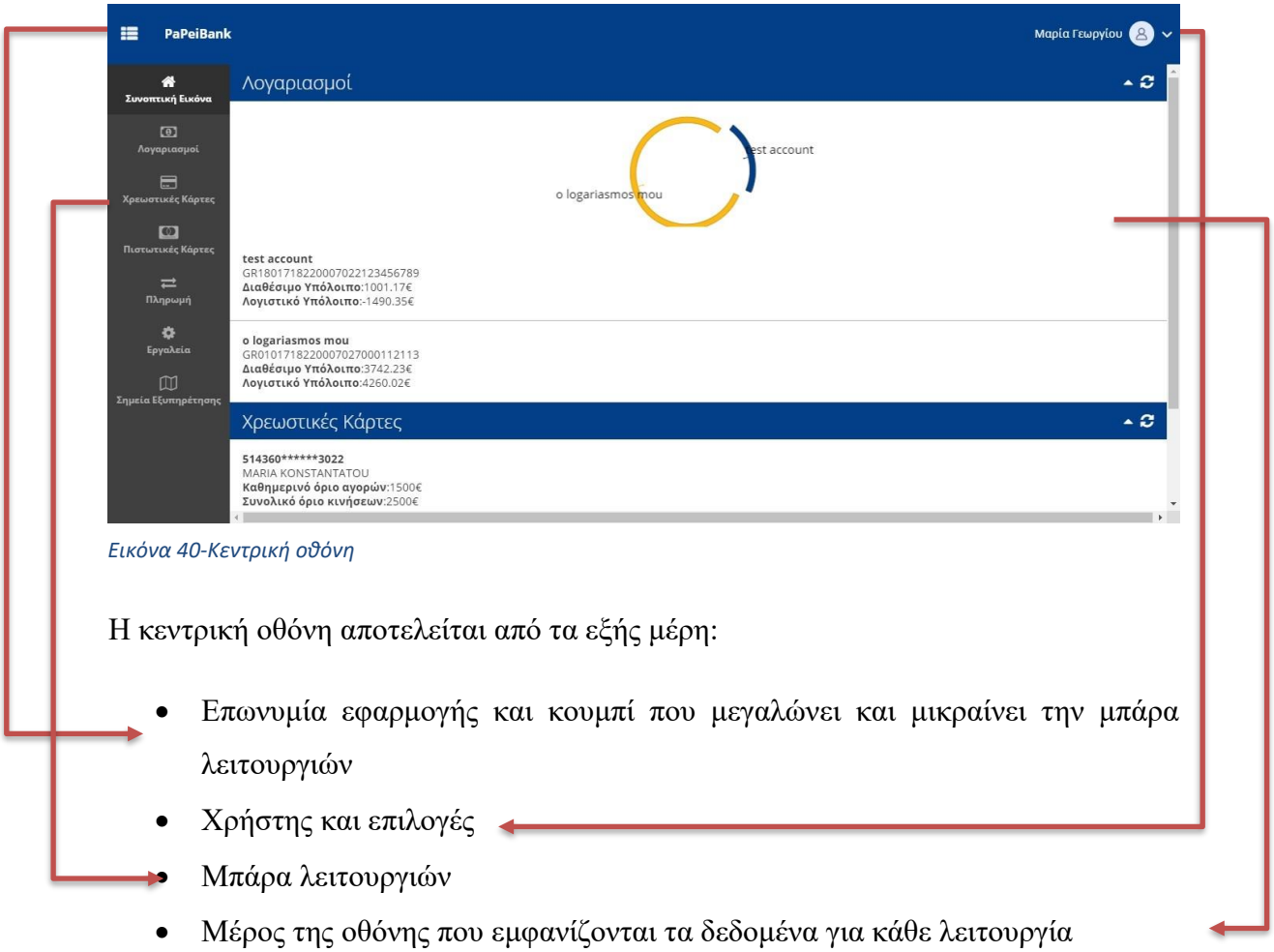
Πατώντας το κουμπί login οδηγούμαστε στην παρακάτω οθόνη στην οποία θα συμπληρώσουμε τα διαπιστευτήρια μας (username, password). Ο χρήστης που θα χρησιμοποιήσουμε έχει username UserC και password 123.



Εικόνα 39-Pop up φόρμα για login

Αφού συμπληρώσουμε επιτυχώς τα στοιχεία μας η εφαρμογή ακολουθεί το flow που περιεγράφηκε στην ενότητα 4.2.2 και απεικονίζεται στην εικόνα 27 και μας μεταφέρει στην κεντρική οθόνη της εφαρμογής.

5.2.2 Κεντρική οθόνη



Εικόνα 40-Κεντρική οθόνη

Η κεντρική οθόνη αποτελείται από τα εξής μέρη:

- Επωνυμία εφαρμογής και κουμπί που μεγαλώνει και μικραίνει την μπάρα λειτουργιών
- Χρήστης και επιλογές
- Μπάρα λειτουργιών
- Μέρος της οθόνης που εμφανίζονται τα δεδομένα για κάθε λειτουργία

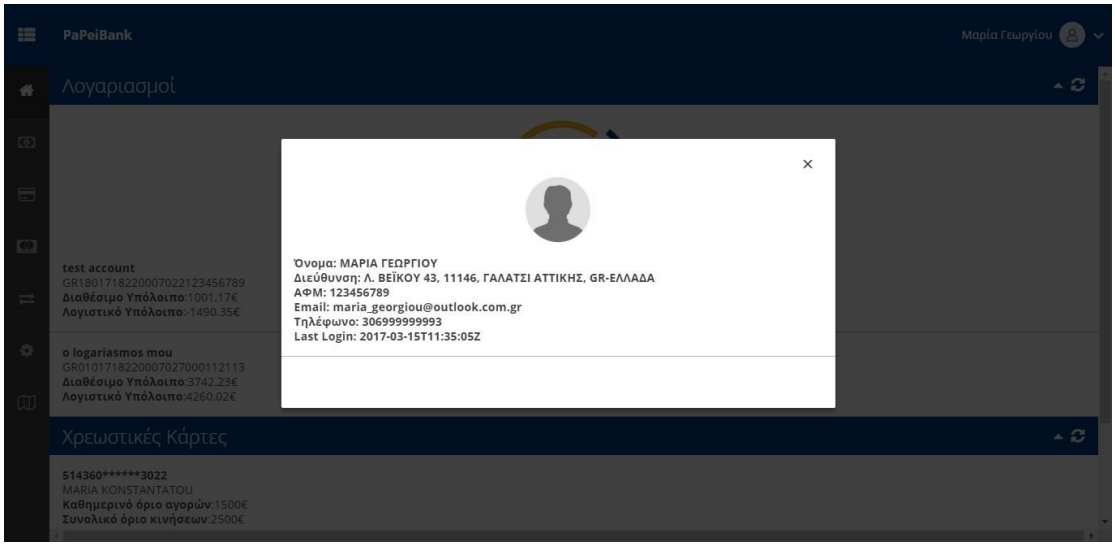
5.2.3 Χρήστης και επιλογές

Στο άνω δεξιά κομμάτι της οθόνης υπάρχει το όνομα του συνδεδεμένου χρήστη και ένα εικονίδιο με ένα βελάκι το οποίο μας δείχνει τις διαθέσιμες επιλογές.

Εικόνα 41-Επιλογές χρήστη

Οι διαθέσιμες επιλογές είναι :

- User Info



Εικόνα 42-User Info

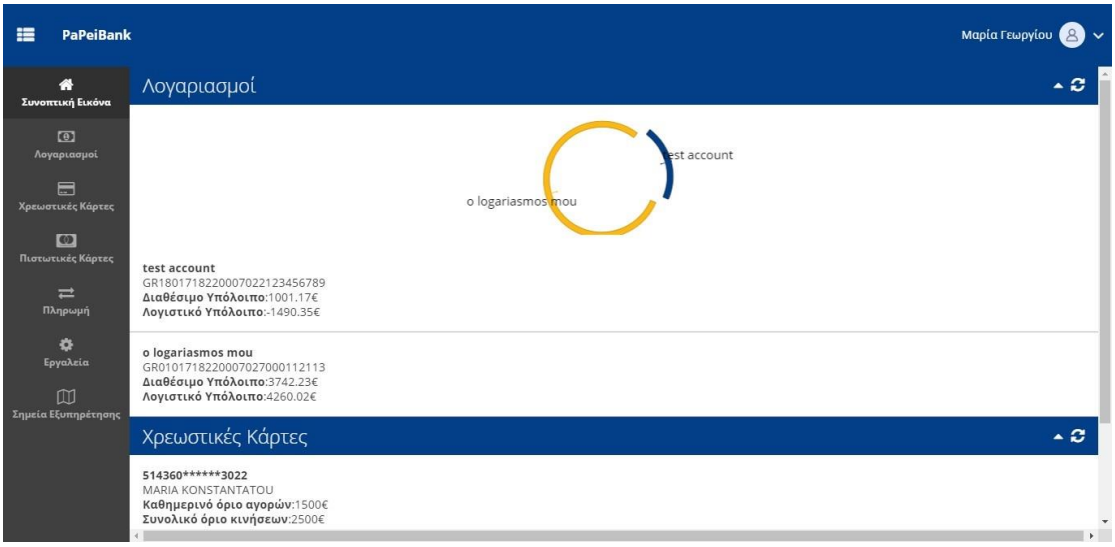
Αυτή η επιλογή μας εμφανίζει ένα παράθυρο με όλες τις διαθέσιμες πληροφορίες για τον συνδεδεμένο χρήστη.

- Logout : Επιλογή αποσύνδεσης από την εφαρμογή.

5.2.4 Μπάρα λειτουργιών

Στα αριστερά της εφαρμογής βρίσκεται η μπάρα με τις διαθέσιμες λειτουργίες , οι οποίες είναι:

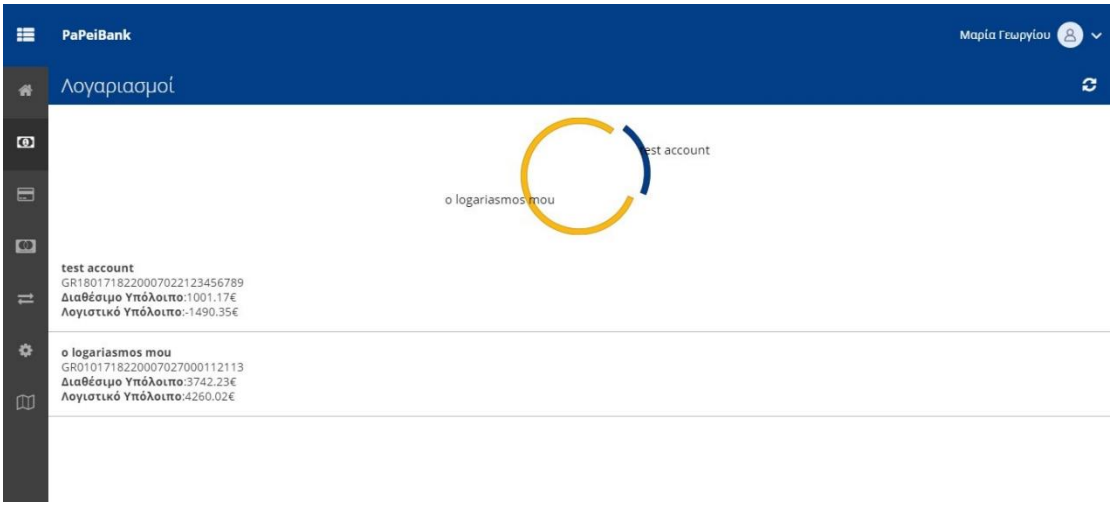
- **<u>Συνοπτική Εικόνα:</u>**



Εικόνα 43- Συνοπτική εικόνα

Σε αυτή την επιλογή ο χρήστης βλέπει όλα τα στοιχεία του συγκεντρωμένα σε 3 κατηγορίες: λογαριασμοί, πιστωτικές κάρτες, χρεωστικές κάρτες. Κάθε μια από αυτές τις κατηγορίες αποτελεί και ξεχωριστή επιλογή της εφαρμογής και θα τις αναλύσουμε παρακάτω. Επιπλέον ο χρήστης μπορεί να δει περισσότερες πληροφορίες για το κάθε στοιχείο.

- **Λογαριασμοί:**

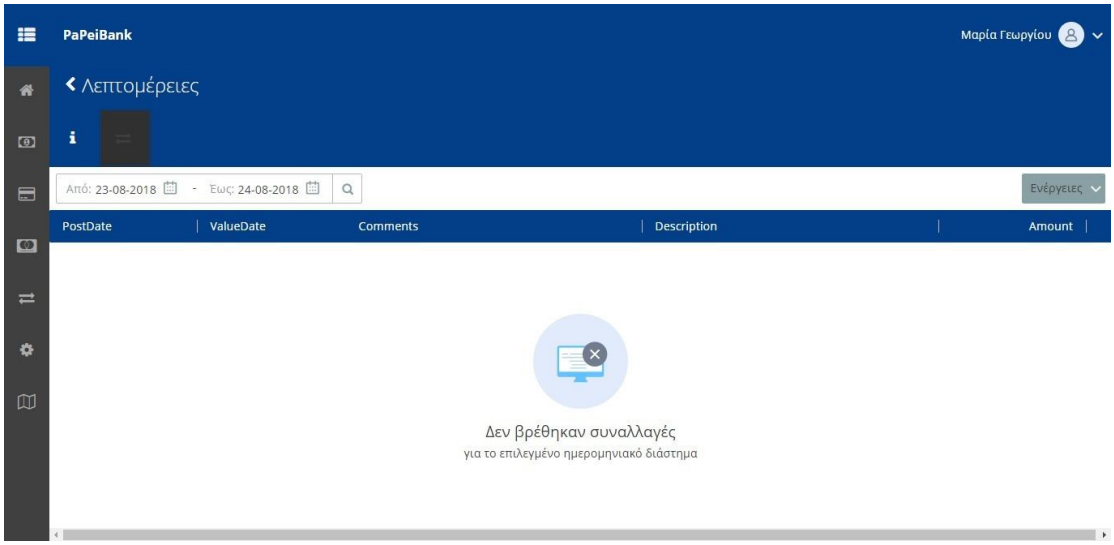


Εικόνα 44-Λογαριασμοί

Στην επιλογή αυτή ο χρήστης το πρώτο πράγμα που παρατηρεί είναι ένα γράφημα. Το γράφημα αυτό απεικονίζει τι μέρος των συνολικών του υπολοίπων αποτελεί ο κάθε λογαριασμός. Στη συνέχεια υπάρχει μια λίστα με τους διαθέσιμους λογαριασμούς του χρήστη. Για τον κάθε λογαριασμό ο χρήστης βλέπει την «φιλική» ονομασία του λογαριασμού, το IBAN του, το λογιστικό και το διαθέσιμο υπόλοιπο. Πατώντας κάποιον από τους λογαριασμούς έχουμε πρόσβαση σε περισσότερες λεπτομέρειες.

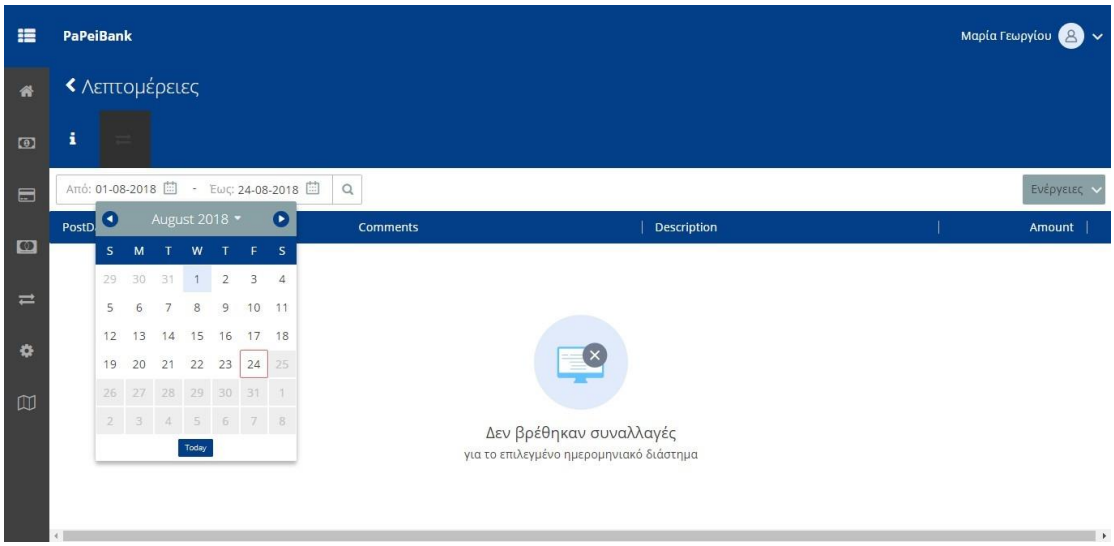
Εικόνα 45-Λεπτομέρειες Λογαριασμού

Όπως βλέπουμε στην εικόνα 45 στις περισσότερες λεπτομέρειες του λογαριασμού έχουμε δύο επιλογές. Η πρώτη μας δίνει περισσότερες πληροφορίες και στοιχεία για τον λογαριασμό.



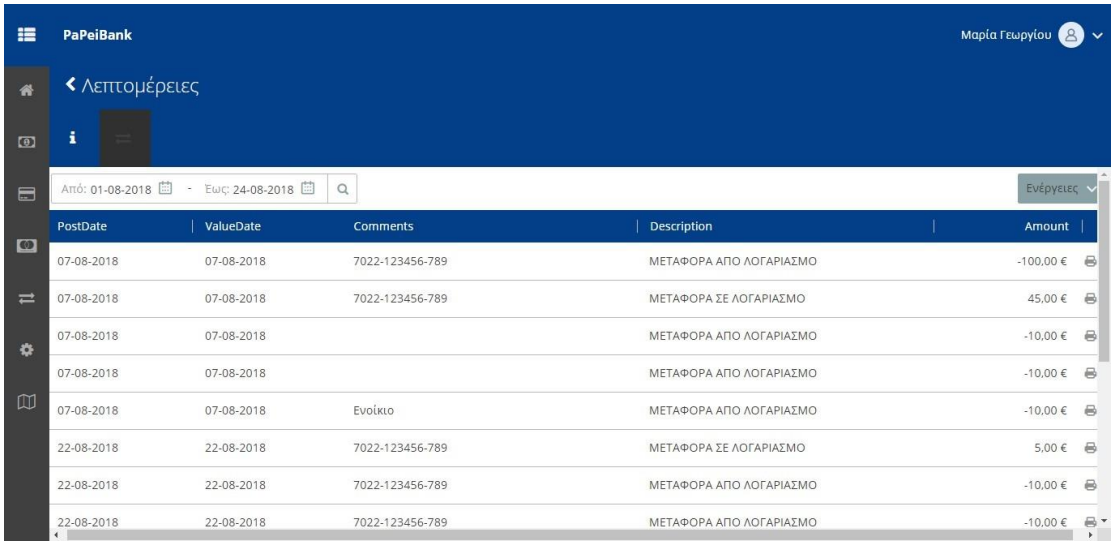
Εικόνα 46-Κινήσεις Λογαριασμού

Η δεύτερη επιλογή μας δίνει τις κινήσεις που έχουν εκτελεστεί για τον λογαριασμό αυτό.



Εικόνα 47-Φίλτρα Κινήσεων

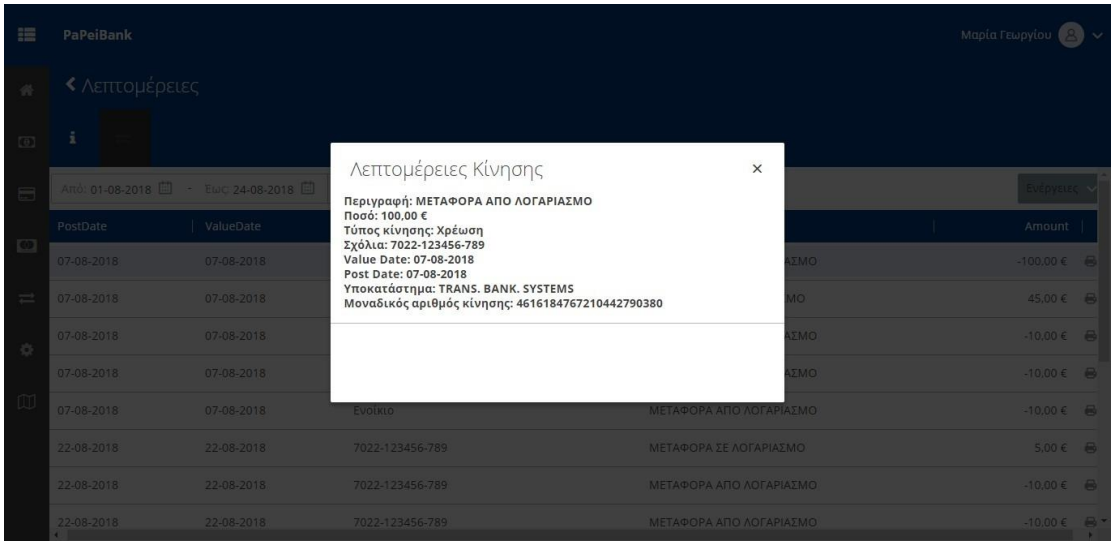
Ορίζοντας τα κατάλληλα ημερομηνιακά διαστήματα το σύστημα μας επιστρέφει τις κινήσεις που έχουν εκτελεστεί σε αυτό τον λογαριασμό εκείνη την περίοδο.



Εικόνα 48-Αναλυτικές Κινήσεις

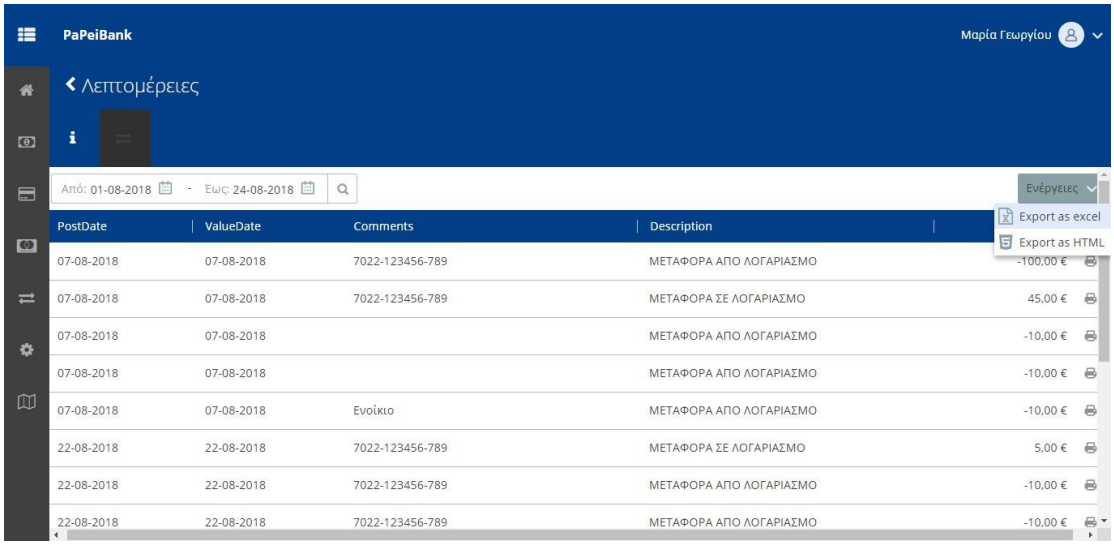
Για τις κινήσεις έχουμε τις εξής ενέργειες:

- Κάνοντας διπλό κλικ πάνω σε κάθε εγγραφή βλέπουμε περισσότερες λεπτομέρειες για την συγκεκριμένη κίνηση.



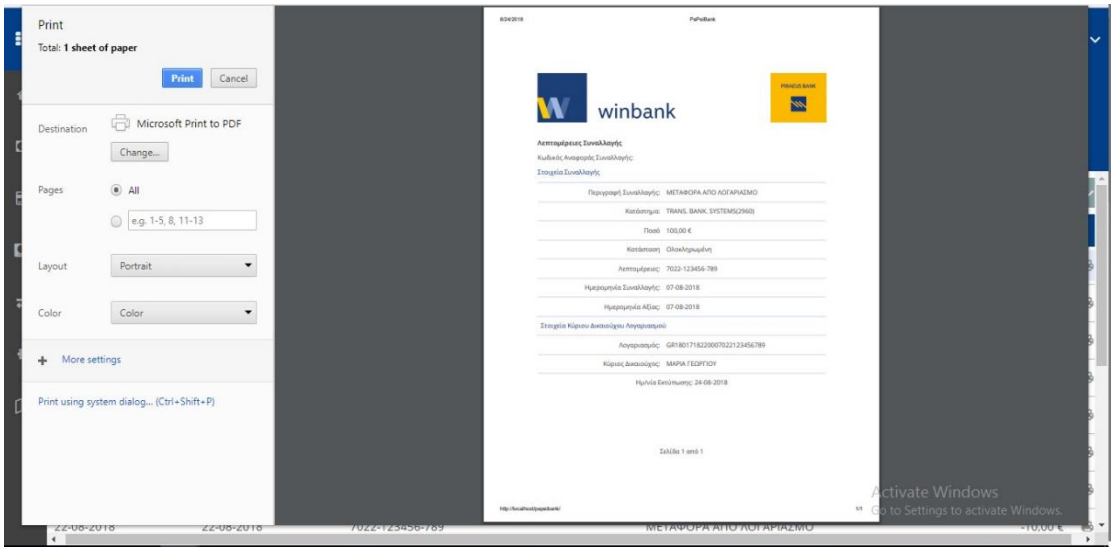
Εικόνα 49-Λεπτομέρειες Κίνησης

- Από το κουμπί «Ενέργειες» έχουμε τις επιλογές «Εξαγωγή σε excel» και «Εξαγωγή σε html» , οι οποίες δημιουργούν τα αντίστοιχα αρχεία στον δίσκο του υπολογιστή του χρήστη.



Εικόνα 50-Ενέργειες σε κινήσεις

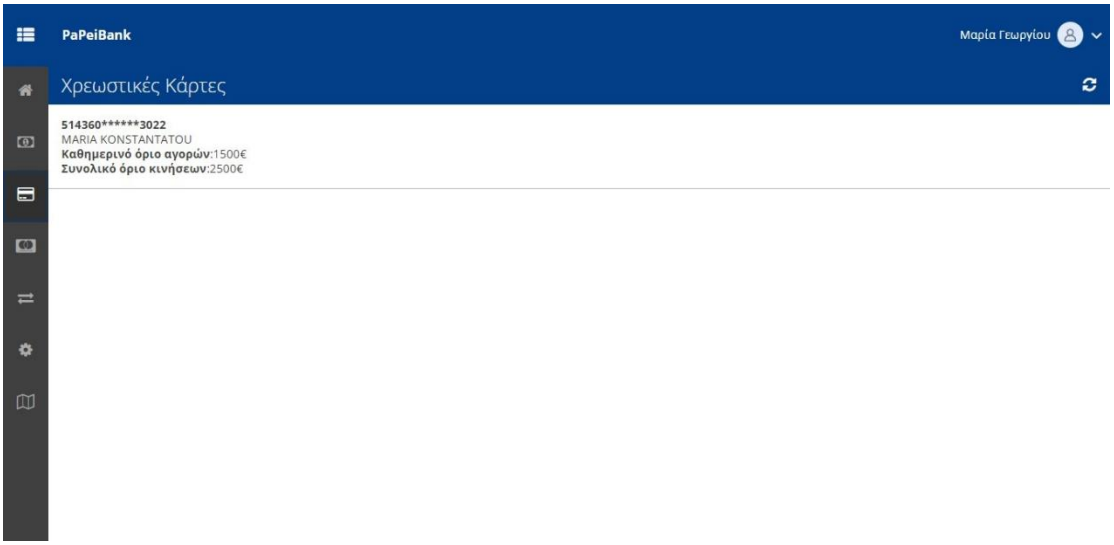
- Σε κάθε γραμμή η τελευταία στήλη του πίνακα είναι ένας εκτυπωτής. Πατώντας τον εκτυπωτή δημιουργούμε μια εκτύπωση , όμοια με την εκτύπωση της τράπεζας.



Εικόνα 51-Εκτύπωση κίνησης

- **<u>Χρεωστικές Κάρτες:</u>**

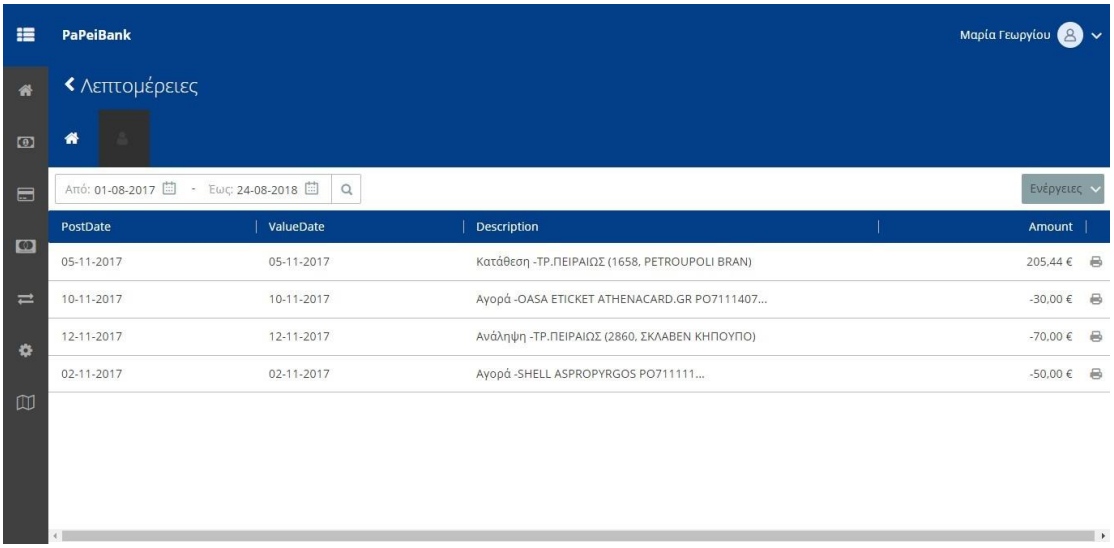
Η επιλογή αυτή εμφανίζει μια λίστα με τις διαθέσιμες χρεωστικές κάρτες του χρήστη.



Εικόνα 52-Χρεωστικές Κάρτες

Ομοίως με τους λογαριασμούς, μπορούμε να δούμε:

- Περισσότερες πληροφορίες για την κάθε κάρτα



Εικόνα 54-Κινήσεις Χρεωστικής Κάρτας

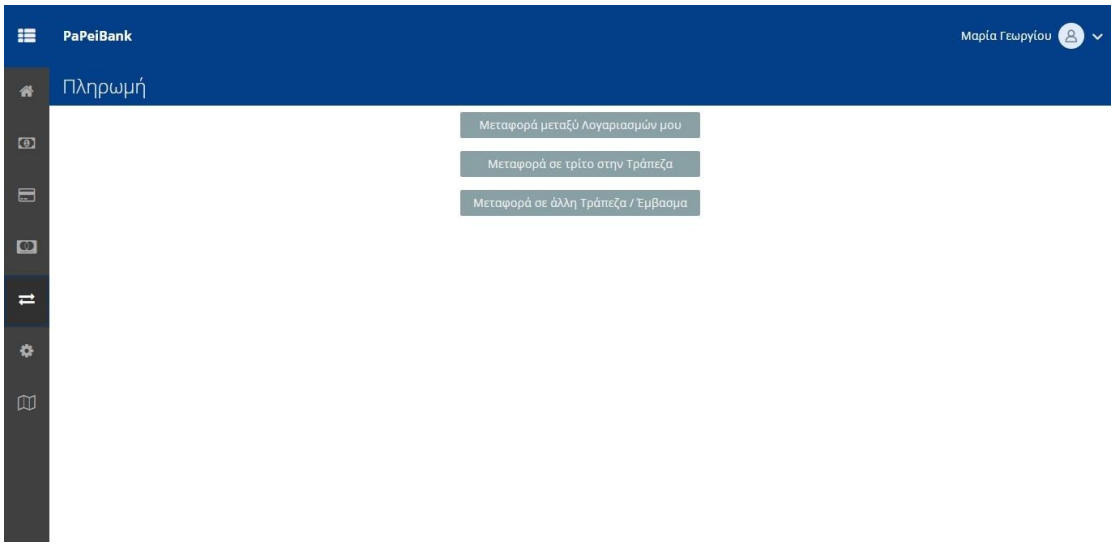
- Τις κινήσεις μας για την χρεωστική κάρτα
- Να δούμε περισσότερες πληροφορίες για την κάθε κίνηση
- Να εκτυπώσουμε την κίνηση
- Να εξάγουμε excel και html για τις κινήσεις αυτές

- **<u>Πιστωτικές Κάρτες:</u>**

Η επιλογή αυτή εμφανίζει μια λίστα με τις διαθέσιμες πιστωτικές κάρτες του χρήστη. Οι ενέργειες που μπορεί να εκτελέσει ο χρήστης είναι ίδιες με αυτές που κάνει στις χρεωστικές.

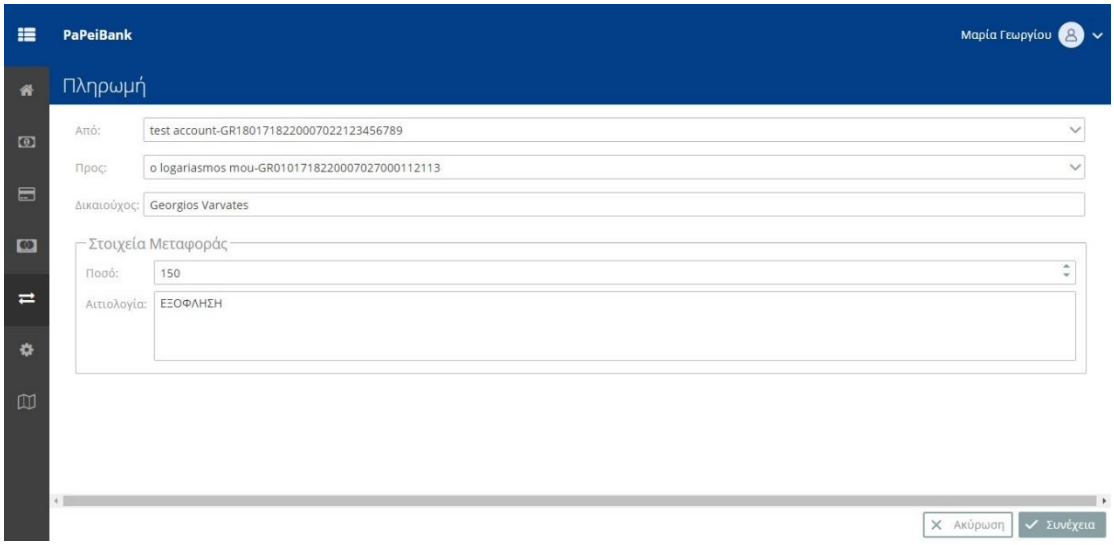
- **Πληρωμή:**

Η επιλογή αυτή επιτρέπει στον χρήστη να μεταφέρεις χρήματα μεταξύ λογαριασμών και να εκτελέσει εμβάσματα. Ο χρήστης μπορεί να στείλει χρήματα μεταξύ δικών τους λογαριασμών, να στείλει χρήματα σε άλλον λογαριασμό που ανήκει στην τράπεζα Πειραιώς, να στείλει χρήματα σε λογαριασμό άλλης τράπεζας. Θα δούμε και τις 3 περιπτώσεις πιο αναλυτικά.



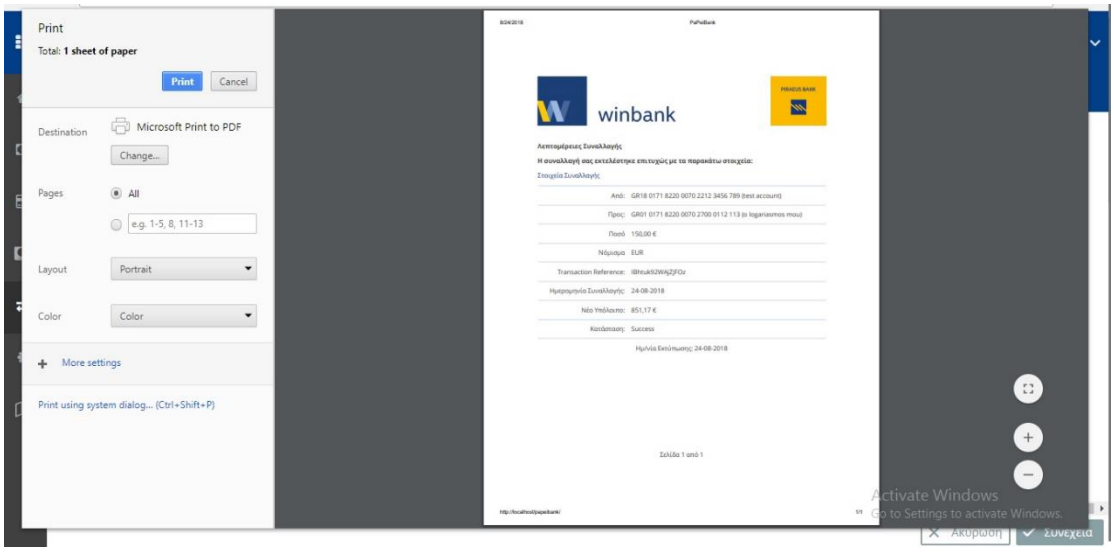
Εικόνα 55-Επιλογές πληρωμών

- Μεταφορά σε δικό του λογαριασμό



Εικόνα 56-Μεταφορά σε δικό του λογαριασμό

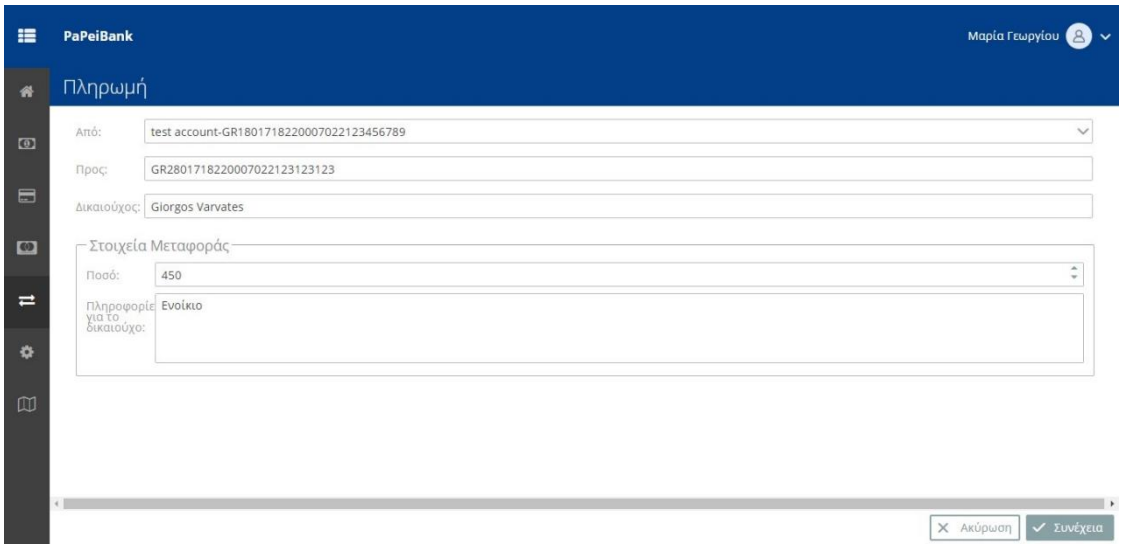
Σε αυτή την περίπτωση ο χρήστης επιλέγει από τους δύο selectors τον λογαριασμό που θα λάβει τα χρήματα και τον λογαριασμό που θα τα στείλει, τον δικαιούχο του λογαριασμού που λαμβάνει, το ποσό και μια αιτιολογία. Πατάει «Συνέχεια» και εκτελείται η κίνηση άμεσα. Μετά την εκτέλεση της κίνησης ο χρήστης ερωτάται αν θέλει να εκτυπώσει ένα αποδεικτικό της κίνησης. Σε περίπτωση που πατήσει «Ναι» το αποδεικτικό που θα λάβει είναι όπως φαίνεται στην εικόνα 57



Εικόνα 57-Αποδεικτικό πληρωμής μεταξύ δικών του λογαριασμών

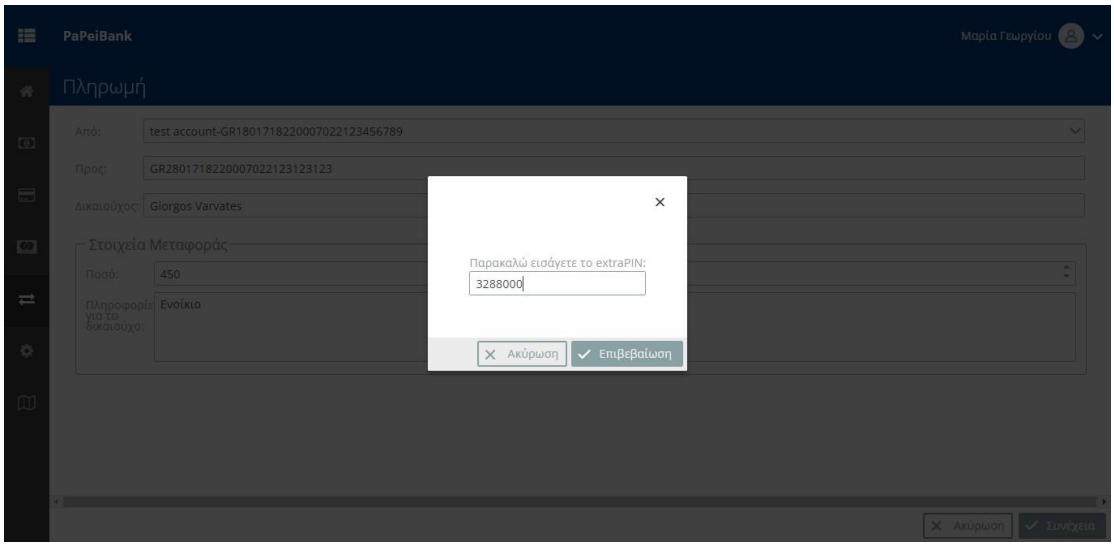
- Μεταφορά σε τρίτο στην Τράπεζα

Σε αυτή την περίπτωση ο χρήστης συμπληρώνει μόνος του το IBAN του παραλήπτη και γίνεται έλεγχος από το σύστημα ως προς την εγκυρότητα του IBAN.



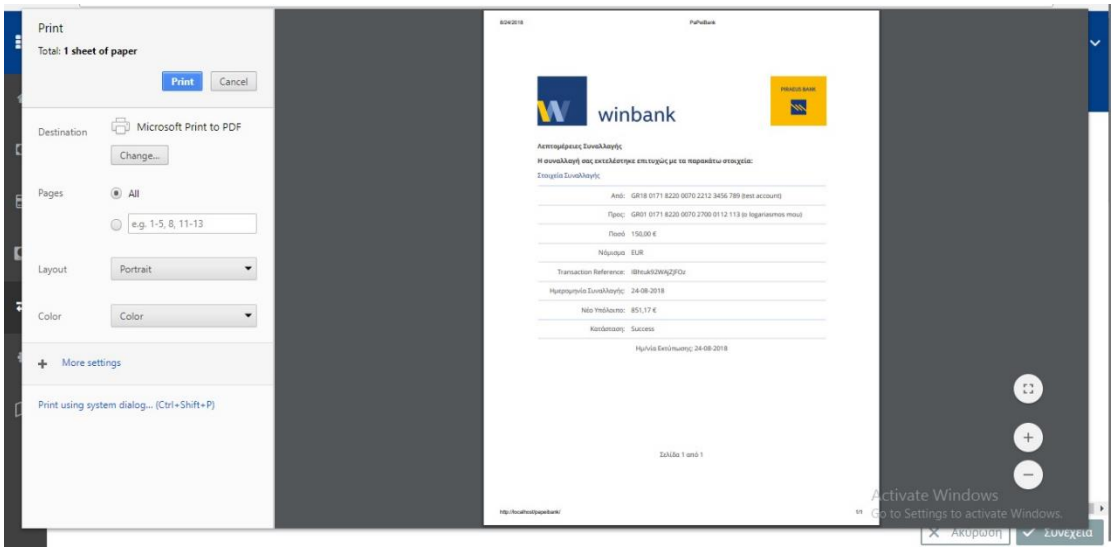
Εικόνα 58-Μεταφορά σε λογαριασμό Πειραιώς

Σε αντίθεση με την προηγούμενη περίπτωση , η κίνηση δεν θα εκτελεστεί άμεσα με το πάτημα του κουμπιού «Συνέχεια». Στα πλαίσια του Second Factor Authentication που περιγράψαμε στην ενότητα 4.2.3 , θα ζητηθεί από τον χρήστη να συμπληρώσει έναν κωδικό ασφαλείας που του έχει έρθει στο κινητό του τηλέφωνο.



Εικόνα 59-Second Factor Authentication

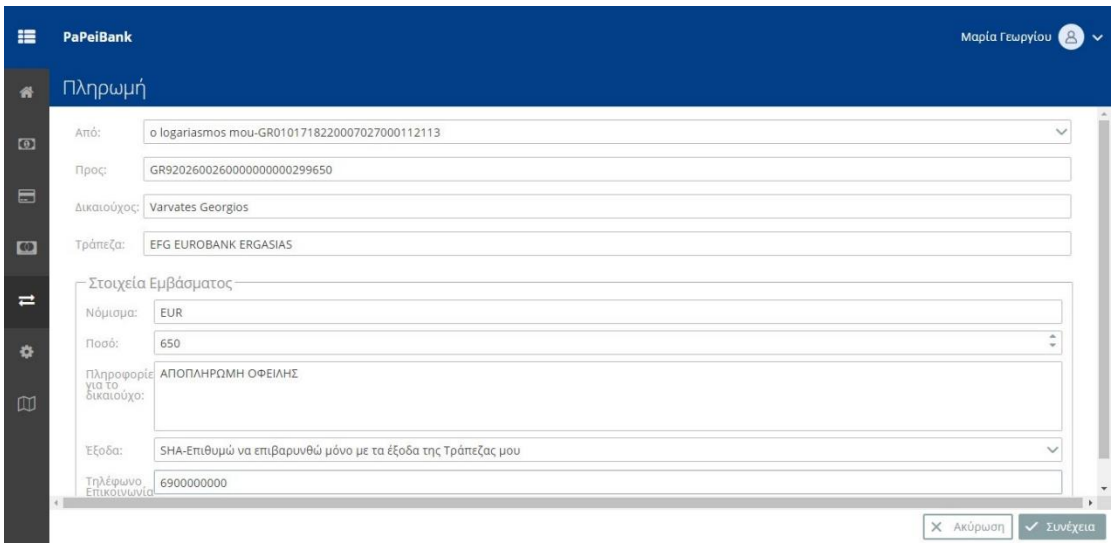
Ένα ενδεικτικό αποδεικτικό κίνησης για αυτή την περίπτωση είναι αυτό της εικόνας 60.



Εικόνα 60-Αποδεικτικό πληρωμής εντός τράπεζας Πειραιώς

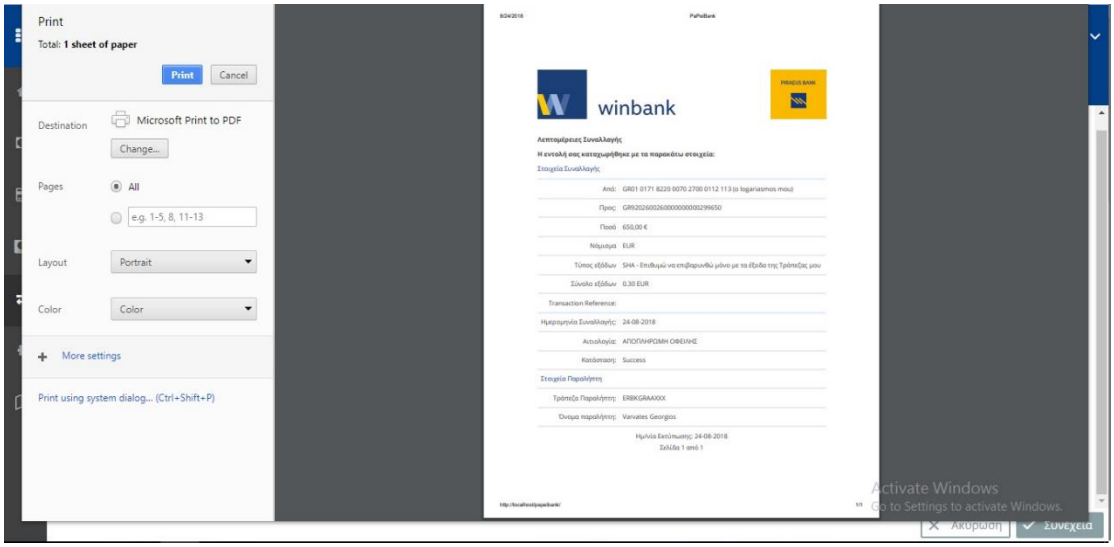
- Μεταφορά σε άλλη τράπεζα/Εμβάσματα

Σε αυτή την περίπτωση, είναι περισσότερα τα απαραίτητα στοιχεία για να εκτελεστεί η κίνηση



Εικόνα 61-Μεταφορά σε άλλη τράπεζα/Εμβάσματα

Το πεδίο τράπεζα συμπληρώνεται αυτόματα με την συμπλήρωση ενός έγκυρου IBAN.Η εκτύπωση σε αυτή την περίπτωση είναι η εξής:



Εικόνα 62-Εκτύπωση σε μεταφορά σε άλλη τράπεζα/εμβάσματα

- **<u>Εργαλεία:</u>**

Η επιλογή εργαλεία , δίνει στον χρήστη τα εξής 2 εργαλεία.

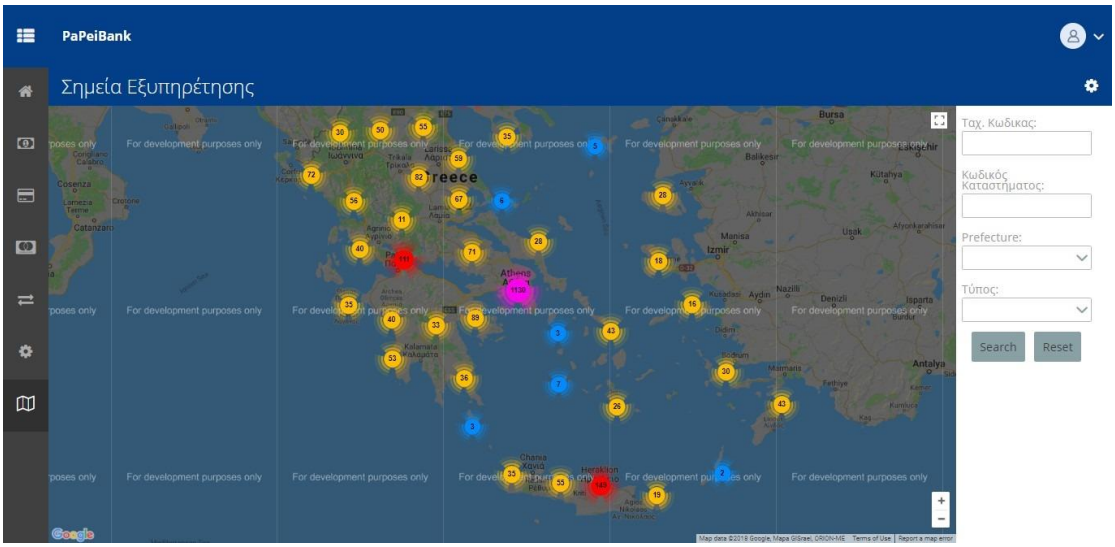
1) Έλεγχος εγκυρότητας του αριθμού λογαριασμού IBAN σε οποιασδήποτε Τράπεζα . Η λειτουργία αυτή επιβεβαιώνει την εγκυρότητα του αριθμού IBAN και δεν μπορεί να επιβεβαιώσει ότι ο λογαριασμός αυτός υφίσταται.
2) Μετατροπή αριθμού λογαριασμού της Τράπεζας Πειραιώς σε IBAN

Εικόνα 63-Εργαλεία

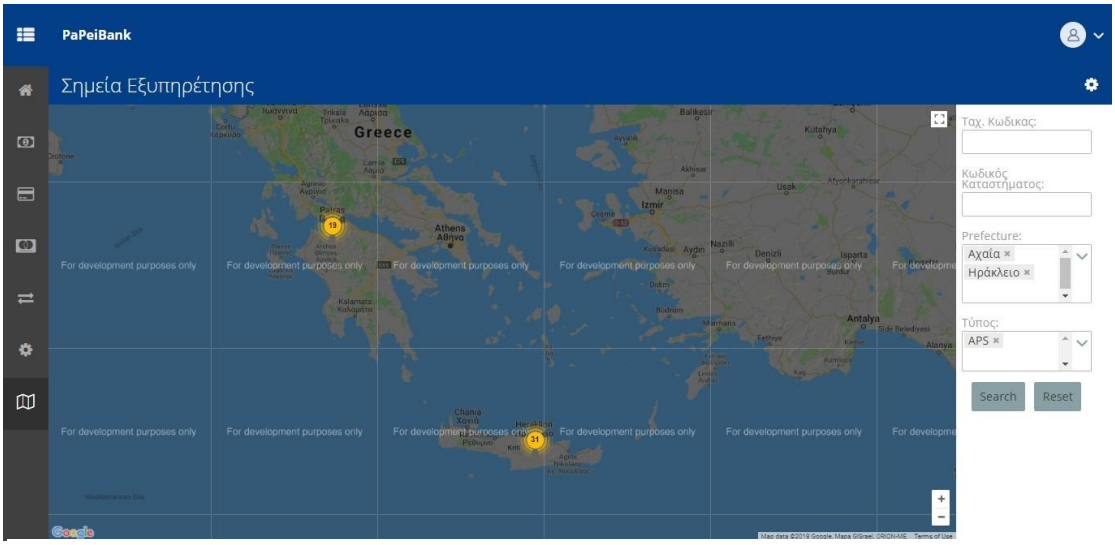
- **<u>Σημεία Εξυπηρέτησης:</u>**

Η επιλογή αυτή δίνει στον χρήστη την δυνατότητα να βρει όλα τα διαθέσιμα σημεία εξυπηρέτησης της Τράπεζας Πειραιώς καθώς και να μάθει περισσότερες πληροφορίες για αυτά όπως ώρες λειτουργίας, υπηρεσίες που εξυπηρετούν κ.α. Τα σημεία εξυπηρέτησης χωρίζονται σε 4 κατηγορίες : φυσικά καταστήματα, e-branches, ATM(automated teller machine), APS(automated payment systems).

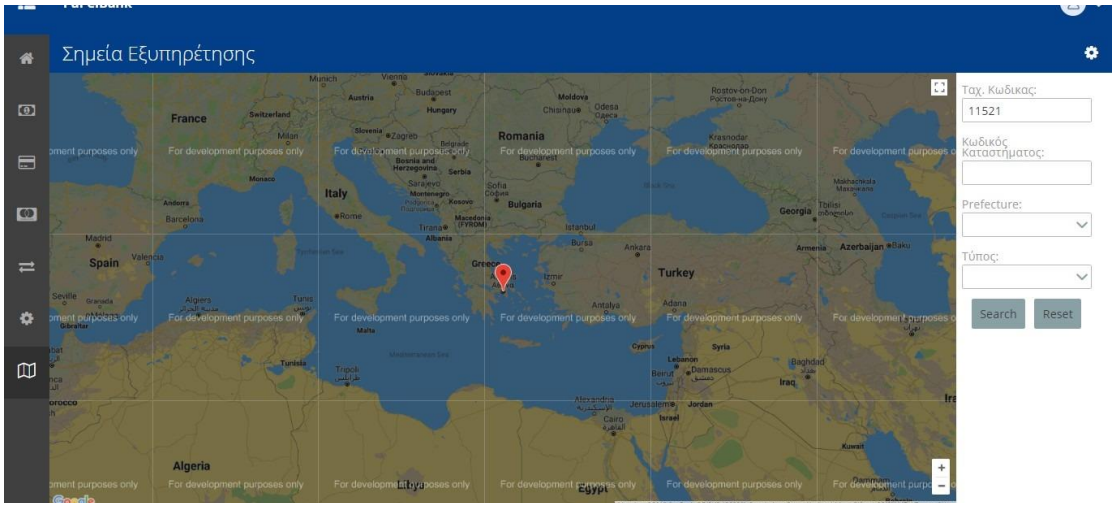
Τέλος ο χρήστης έχει τη δυνατότητα να φιλτράρει τα σημεία με βάση τον ταχυδρομικό κώδικα, τον κωδικό του καταστήματος, τον νομό που ανήκει και την κατηγορία. Τα φίλτρα αυτά μπορούν να χρησιμοποιηθούν μεμονωμένα ή συνδυαστικά.



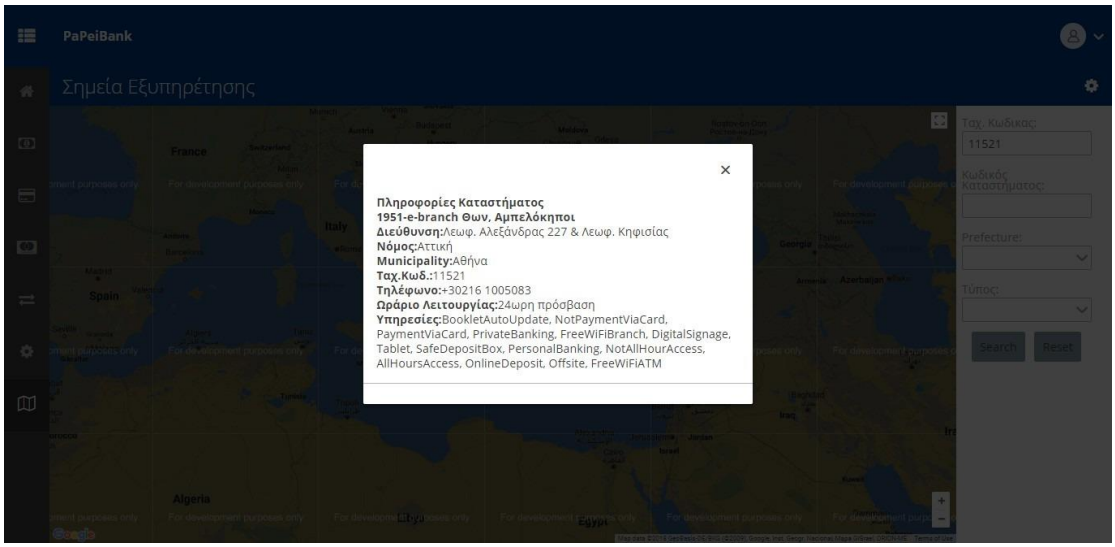
Εικόνα 64-Σημεία Εξυπηρέτησης



Εικόνα 66-Φιλτράρισμα με ταχυδρομικό κώδικα



Εικόνα 67-Πληροφορίες Καταστήματος



Εικόνα 65-Συνδυαστικό φιλτράρισμα

Συμπεράσματα- Επεκτάσεις

Τα συμπεράσματα που προκύπτουν από την παραπάνω εφαρμογή είναι ιδιαίτερα ενθαρρυντικά. Οι χρήστες της εφαρμογής ή άλλων παρόμοιων εφαρμογών μπορούν να έχουν πλήρη έλεγχο των τραπεζικών τους στοιχείων χωρίς να είναι αναγκασμένοι να συνδέονται απευθείας από τα συστήματα των τραπεζών. Ο πελάτης της τράπεζας δεν είναι υποχρεωμένος να χρησιμοποιεί τα εργαλεία που του προσφέρει η ίδια η τράπεζα, αλλά μπορεί μέσω τρίτων παρόχων να έχει πρόσβαση σε καλύτερα εργαλεία. Επιπλέον αυτός ο ανταγωνισμός μεταξύ τραπεζών και τρίτων παρόχων θα μπορούσε να οδηγήσει σε ακόμα μεγαλύτερη εξέλιξη της ηλεκτρονικής τραπεζικής.

Επίσης η εφαρμογή μας θα μπορούσε να διασυνδεθεί με άλλα συστήματα , πχ ERP, με σκοπό την καλύτερη διαχείριση και επεξεργασία των στοιχείων του πελάτη. Ένα παράδειγμα εφαρμογής θα μπορούσε να είναι η αυτόματη ενημέρωση του ERP συστήματος του χρήστη σχετικά με τα υπόλοιπα του στους λογαριασμούς του, χωρίς να χρειάζεται να κάνει χειροκίνητη καταχώριση ο ίδιος.

Τέλος προτείνονται οι εξής βελτιώσεις:

- Διασύνδεση και με άλλες τράπεζες.
- Παράλληλη διασύνδεση σε όλες τις τράπεζες μέσω μιας εφαρμογής.
- Πραγματοποίηση πληρωμών λογαριασμών, ΔΕΚΟ κτλ

Βιβλιογραφία

[1] https://el.wikipedia.org/wiki/%CE%A4%CF%81%CE%AC%CF%80%CE%B5%CE%B6%CE%B1

[2] https://en.wikipedia.org/wiki/History_of_banking

[3] https://www.investopedia.com/articles/07/banking.asp

[4] https://www.gobankingrates.com/banking/banks/history-online-banking/

[5] https://en.wikipedia.org/wiki/Online_banking

[6] https://en.wikipedia.org/wiki/Core_banking

[7] https://www.toptenreviews.com/money/services/best-online-only-banks/

[8] http://www.uth.gr/main/help/help-desk/internet/internet3.html

[9] https://www.investopedia.com/terms/h/home-banking.asp

[10] http://www.nielsen.com/jo/en/insights/news/2014/the-evolution-of-modern-banking.html

[11] https://thefinancialbrand.com/32428/pew-research-online-banking-users-demographic-trends/

[12] http://ec.europa.eu/eurostat/tgm/mapToolClosed.do?tab=map&init=1&plugin=1&language=en&pcode=tin00099&toolbox=types

[13] http://icteval.ktpae.gr/stats/delivery2/#

[14] https://www.nbs.sk/_img/Documents/BIATEC/BIA06_06/22_25.pdf

[15] https://www.eurobank.com.cy/el/tools/ibanGenerator

[16] https://www.nbg.gr/greek/i-bank/retail/internet-banking/Documents/03.i-bank-Pliroforisi_IB.pdf

[17] https://www.nbg.gr/greek/i-bank/retail/internet-banking/Documents/01.i-bank-Synallages_IB.pdf

[18] https://www.nbg.gr/el/the-group/the-bank/history

[19] https://www.nbg.gr/

[20] http://www.alpha.gr/page/default.asp?la=1&id=2578

[21] http://www.alpha.gr/

[22] https://www.eurobank.gr/el/retail

[23] https://el.wikipedia.org/wiki/Eurobank

[24] https://www.piraeusbank.gr/el/idiwtes

[25] https://el.wikipedia.org/wiki/%CE%A4%CF%81%CE%AC%CF%80%CE%B5%CE%B6%CE%B1_%CE%A0%CE%B5%CE%B9%CF%81%CE%B1%CE%B9%CF%8E%CF%82

[26] http://nkv.antenna.gr/news/Economy/article/494233/i-trapeza-peiraios-kainotomei-me-tin-platforma-rapid-link

[27] https://en.wikipedia.org/wiki/Open_banking

[28] https://en.wikipedia.org/wiki/HTML

[29] https://www.sencha.com/products/extjs/

[30] https://guide.freecodecamp.org/javascript/advantages-and-disadvantages-of-javascript/

[31] https://el.wikipedia.org/wiki/JavaScript

[32] https://el.wikipedia.org/wiki/CSS

[33] https://en.wikipedia.org/wiki/Internet_Information_Services

[34] https://searchmicroservices.techtarget.com/definition/REST-representational-state-transfer

Παράρτημα

Στο κεφάλαιο αυτό θα παρεμβάλουμε τα σημαντικότερα κομμάτια κώδικα.

Peiraios.cs

```
using Newtonsoft.Json;
using Newtonsoft.Json.Linq;
using RestSharp;
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web;
using System.Web.Script.Serialization;
using Utilities;
using Utilities.Helpers;
using Utilities.Objects;

namespace PaPeiBankConnector
{
public class Peiraios : BaseBank
{
private string FCurrentDir = "";

private static readonly HttpClient client = new HttpClient();
public Peiraios(string currentDir)
{
FCurrentDir = currentDir;
}

public string stringFromFile(string filename)
{
string text = "";
using (StreamReader sr = new StreamReader(FCurrentDir +
"\\FakeData\\" + filename, System.Text.Encoding.Default))
{
text = sr.ReadToEnd();
}
return text;
}

override public string getPretoken(string jsonParams)
{
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
string code = JObject.Parse(jsonParams)["code"].ToString();
string redirect_uri =
JObject.Parse(jsonParams)["redirect_uri"].ToString();
string scope = JObject.Parse(jsonParams)["scope"].ToString();
var arguments = "grant_type=authorization_code" +
"&client_id=" + Constants.client_id +
"&client_secret=" + Constants.client_secret +
"&code=" + code +
"&redirect_uri=" + redirect_uri;
string PreTokenUri = "";
```

```
if (scope.Equals("/sandboxapi"))
{
PreTokenUri =
"https://api.rapidlink.piraeusbank.gr/piraeusbank/production/oauth/oa
uth2/token";
}
else
{
PreTokenUri =
"https://openbank.piraeusbank.gr/identityserver/connect/token";
}
var client = new RestClient(PreTokenUri);
var request = new RestRequest(Method.POST);
request.AddHeader("accept", "application/json");
request.AddParameter("application/x-www-form-urlencoded", arguments,
ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var accountObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(accountObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);

}
}
override public string getProfile(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/customer/info");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
```

```
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var userObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(userObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);

}
}
override public string getAssets(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/accounts/");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
```

```
{
var emptyObjectArray = new object[1];
var accountObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(accountObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);

}
}

override public string getAccountDetails(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string iban = JObject.Parse(data)["account"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/accounts/" + iban + "/details");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var accountObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(accountObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}

override public string getAccountTransactions(string jsonParams)
{
var client = new RestClient();
```

```
var request = new RestRequest(Method.GET);
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string account = JObject.Parse(data)["account"].ToString();
string fromDate =
DateTime.Parse(JObject.Parse(data)["fromDate"].ToString()).ToString("
yyyy-MM-dd");
string toDate =
DateTime.Parse(JObject.Parse(data)["toDate"].ToString()).ToString("yy
yy-MM-dd");
string fromRow = JObject.Parse(data)["fromRow"].ToString();
var transactionObj = new TransactionData(fromDate, toDate, fromRow);
string filterInput = string.Empty;
string Uri = string.Empty;
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
List<NormalizedPiraeusTransaction> allTransactions = new
List<NormalizedPiraeusTransaction>();
// List<NormalizedPiraeusTransaction> groupedTransactions = new
List<NormalizedPiraeusTransaction>();
var normalizedContentFail = new NormalizedFailResponse(false, "",
"");
var normalizedContentSuccess = new NormalizedSuccessResponse(true,
"", allTransactions.ToArray(), "null");
while (!String.IsNullOrEmpty(fromRow) ||
String.IsNullOrEmpty(filterInput))
{
transactionObj = new TransactionData(fromDate, toDate, fromRow);
filterInput = new JavaScriptSerializer().Serialize(transactionObj);
Uri =
"https://api.rapidlink.piraeusbank.gr/piraeusbank/production/v1.1/ass
ets/accounts/" + account + "/transactions/" +
HttpUtility.UrlEncode(filterInput);
client = new RestClient(Uri);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
normalizedContentFail = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContentFail);
}
else
{
```

```
var transactions =
JsonConvert.DeserializeObject<NormalizedPiraeusTransactions>(response
.Content);
for (int i = 0; i < transactions.Transactions.Length; i++)
{
allTransactions.Add(transactions.Transactions[i]);
}
fromRow = transactions.PagingInfo.NextPagePositioningKey;

}

}
var groupedTransactions = allTransactions.GroupBy(u =>
u.TransactionReference).ToArray();//[0].ToList()[0];
foreach (var group in groupedTransactions)
{
if (group.Count() > 1)
{
var listItemsGroup = group.ToList();
var count = listItemsGroup.Count();
if (Math.Abs(listItemsGroup[0].Amount) <
Math.Abs(listItemsGroup[1].Amount))
{
listItemsGroup[0].TransactionReference =
System.Convert.ToBase64String(System.Text.Encoding.UTF8.GetBytes(list
ItemsGroup[0].TransactionReference));
}
else
{
listItemsGroup[1].TransactionReference =
System.Convert.ToBase64String(System.Text.Encoding.UTF8.GetBytes(list
ItemsGroup[0].TransactionReference));
}
}
}

normalizedContentSuccess = new NormalizedSuccessResponse(true, "200",
allTransactions.ToArray(), "");

return new
JavaScriptSerializer().Serialize(normalizedContentSuccess);

}

override public string getCreditCards(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/cards/credit");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
```

```
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var cardObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(cardObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);

}
}
override public string getCreditCardDetails(string jsonParams)
{
//Debugger.Launch();
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string card = JObject.Parse(data)["card"].ToString();
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/cards/credit/" + card + "/details");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
```

```
else
{
var emptyObjectArray = new object[1];
var cardObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(cardObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}

override public string getCreditCardsTransactions(string jsonParams)
{
var client = new RestClient();
var request = new RestRequest(Method.GET);
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string account = JObject.Parse(data)["card"].ToString();
string fromDate =
DateTime.Parse(JObject.Parse(data)["fromDate"].ToString()).ToString("
yyyy-MM-dd");
string toDate =
DateTime.Parse(JObject.Parse(data)["toDate"].ToString()).ToString("yy
yy-MM-dd");
string fromRow = JObject.Parse(data)["fromRow"].ToString();
var transactionObj = new TransactionData(fromDate, toDate, fromRow);
string filterInput = string.Empty;
string Uri = string.Empty;
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
List<NormalizedPiraeusDebitTransaction> allTransactions = new
List<NormalizedPiraeusDebitTransaction>();
// List<NormalizedPiraeusTransaction> groupedTransactions = new
List<NormalizedPiraeusTransaction>();
var normalizedContentFail = new NormalizedFailResponse(false, "",
"");
var normalizedContentSuccess = new NormalizedSuccessResponse(true,
"", allTransactions.ToArray(), "null");
while (!String.IsNullOrEmpty(fromRow) ||
String.IsNullOrEmpty(filterInput))
{
transactionObj = new TransactionData(fromDate, toDate, fromRow);
filterInput = new JavaScriptSerializer().Serialize(transactionObj);
Uri =
"https://api.rapidlink.piraeusbank.gr/piraeusbank/production/v1.1/ass
ets/cards/credit/" + account + "/transactions/" +
HttpUtility.UrlEncode(filterInput);
client = new RestClient(Uri);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
```

```
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
normalizedContentFail = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContentFail);
}
else
{
var transactions =
JsonConvert.DeserializeObject<NormalizedPiraeusDebitTransactions>(res
ponse.Content);
for (int i = 0; i < transactions.Transactions.Length; i++)
{
allTransactions.Add(transactions.Transactions[i]);
}
fromRow = transactions.PagingInfo.NextPagePositioningKey;

}

}
var groupedTransactions = allTransactions.GroupBy(u =>
u.ReferenceNo).ToArray();//[0].ToList()[0];
foreach (var group in groupedTransactions)
{
if (group.Count() > 1)
{
var listItemsGroup = group.ToList();
var count = listItemsGroup.Count();
if (Math.Abs(listItemsGroup[0].DebitAmount.Amount) <
Math.Abs(listItemsGroup[1].DebitAmount.Amount))
{
listItemsGroup[0].ReferenceNo =
System.Convert.ToBase64String(System.Text.Encoding.UTF8.GetBytes(list
ItemsGroup[0].ReferenceNo));
}
else
{
listItemsGroup[1].ReferenceNo =
System.Convert.ToBase64String(System.Text.Encoding.UTF8.GetBytes(list
ItemsGroup[0].ReferenceNo));
}
}
}

normalizedContentSuccess = new NormalizedSuccessResponse(true, "200",
allTransactions.ToArray(), "");

return new
JavaScriptSerializer().Serialize(normalizedContentSuccess);

}

override public string getCreditCardStatements(string jsonParams)
{
//Debugger.Launch();
```

```
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string card = JObject.Parse(data)["card"].ToString();
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/cards/credit/" + card + "/statements ");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var cardObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(cardObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}
override public string getDebitCards(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/cards/debit");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
```

```
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var cardObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(cardObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);

}
}
override public string getDebitCardDetails(string jsonParams)
{
//Debugger.Launch();
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string card = JObject.Parse(data)["card"].ToString();
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/cards/debit/" + card + "/details");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
```

```
else
{
var emptyObjectArray = new object[1];
var cardObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(cardObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}

override public string getDebitCardsTransactions(string jsonParams)
{
var client = new RestClient();
var request = new RestRequest(Method.GET);
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string account = JObject.Parse(data)["card"].ToString();
string fromDate =
DateTime.Parse(JObject.Parse(data)["fromDate"].ToString()).ToString("
yyyy-MM-dd");
string toDate =
DateTime.Parse(JObject.Parse(data)["toDate"].ToString()).ToString("yy
yy-MM-dd");
string fromRow = JObject.Parse(data)["fromRow"].ToString();
var transactionObj = new TransactionData(fromDate, toDate, fromRow);
string filterInput = string.Empty;
string Uri = string.Empty;
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
List<NormalizedPiraeusDebitTransaction> allTransactions = new
List<NormalizedPiraeusDebitTransaction>();
// List<NormalizedPiraeusTransaction> groupedTransactions = new
List<NormalizedPiraeusTransaction>();
var normalizedContentFail = new NormalizedFailResponse(false, "",
"");
var normalizedContentSuccess = new NormalizedSuccessResponse(true,
"", allTransactions.ToArray(), "null");
while (!String.IsNullOrEmpty(fromRow) ||
String.IsNullOrEmpty(filterInput))
{
transactionObj = new TransactionData(fromDate, toDate, fromRow);
filterInput = new JavaScriptSerializer().Serialize(transactionObj);
Uri =
"https://api.rapidlink.piraeusbank.gr/piraeusbank/production/v1.1/ass
ets/cards/debit/" + account + "/transactions/" +
HttpUtility.UrlEncode(filterInput);
client = new RestClient(Uri);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
```

```
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
normalizedContentFail = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContentFail);
}
else
{
var transactions =
JsonConvert.DeserializeObject<NormalizedPiraeusDebitTransactions>(res
ponse.Content);
for (int i = 0; i < transactions.Transactions.Length; i++)
{
allTransactions.Add(transactions.Transactions[i]);
}
fromRow = transactions.PagingInfo.NextPagePositioningKey;

}

}
var groupedTransactions = allTransactions.GroupBy(u =>
u.ReferenceNo).ToArray();//[0].ToList()[0];
foreach (var group in groupedTransactions)
{
if (group.Count() > 1)
{
var listItemsGroup = group.ToList();
var count = listItemsGroup.Count();
if (Math.Abs(listItemsGroup[0].DebitAmount.Amount) <
Math.Abs(listItemsGroup[1].DebitAmount.Amount))
{
listItemsGroup[0].ReferenceNo =
System.Convert.ToBase64String(System.Text.Encoding.UTF8.GetBytes(list
ItemsGroup[0].ReferenceNo));
}
else
{
listItemsGroup[1].ReferenceNo =
System.Convert.ToBase64String(System.Text.Encoding.UTF8.GetBytes(list
ItemsGroup[0].ReferenceNo));
}
}
}

normalizedContentSuccess = new NormalizedSuccessResponse(true, "200",
allTransactions.ToArray(), "");

return new
JavaScriptSerializer().Serialize(normalizedContentSuccess);

}

override public string transfer(string jsonParams)
{
```

```
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();
var convertedData =
JsonConvert.DeserializeObject<NormalizedPiraeusTransfer>(data);
string transferData = new
JavaScriptSerializer().Serialize(convertedData);
string Uri =
"https://api.rapidlink.piraeusbank.gr/piraeusbank/production/v1.1/tra
nsactions/transferToIban";
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
var client = new RestClient(Uri);
var request = new RestRequest(Method.POST);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("content-type", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
request.AddParameter("application/json", transferData,
ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
if (httperrorcode.Equals("401") && errorcode.Equals("API-011"))
{
var normalizedContent = new NormalizedFailResponse(false, message,
errorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}
else
{
var emptyObjectArray = new object[1];
var transferObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(transferObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}
```

```
override public string validateExtraPin(string jsonParams)
//override public string makeTransfer(string jsonParams)
{
//Debugger.Launch();
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string pin = JObject.Parse(data)["smsOtp"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/security/token/validate/" + pin);
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
//return validateExtraPin(jsonParams);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
//return makeTransfer(jsonParams);
var emptyObjectArray = new object[1];
var pinObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(pinObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}
override public string verifyIban(string jsonParams)
//override public string makeTransfer(string jsonParams)
{
//Debugger.Launch();
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string iban = JObject.Parse(data)["account"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
```

```
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/info/verifyIban/" + iban);
var request = new RestRequest(Method.GET);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var pinObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(pinObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}

override public string convertAccountToIban(string jsonParams)
//override public string makeTransfer(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string iban = JObject.Parse(data)["account"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/info/toIban/" + iban);
var request = new RestRequest(Method.GET);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
```

```
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var pinObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(pinObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}
override public string getCountries(string jsonParams)
//override public string makeTransfer(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/info/countries");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var pinObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(pinObject);
```

```
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}
override public string getPointsOfPresence(string jsonParams)
//override public string makeTransfer(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/info/pointsOfPresence");
var request = new RestRequest(Method.GET);
//request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("language", "EN");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var pinObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(pinObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}
override public string getPointsOfPresenceType(string jsonParams)
//override public string makeTransfer(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string type = JObject.Parse(data)["type"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
```

```
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/info/pointsOfPresence/"+ type);
var request = new RestRequest(Method.GET);
//request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("language", "GR");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var pinObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(pinObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}

override public string getIbanInfo(string jsonParams)
{
string bearer = JObject.Parse(jsonParams)["bearer"].ToString();
string data = JObject.Parse(jsonParams)["data"].ToString();//
HttpUtility.UrlDecode(JObject.Parse(jsonParams)["data"].ToString());
string iban = JObject.Parse(data)["account"].ToString();
ServicePointManager.ServerCertificateValidationCallback =
WebApiHelpers.ValidateServerCertificate;
var client = new
RestClient("https://api.rapidlink.piraeusbank.gr/piraeusbank/producti
on/v1.1/assets/accounts/" + iban + "/info");
var request = new RestRequest(Method.GET);
request.AddHeader("authorization", "Bearer " + bearer);
request.AddHeader("accept", "application/json");
request.AddHeader("x-ibm-client-id", Constants.client_id);
IRestResponse response = client.Execute(request);
if (response.ErrorException != null ||
!response.StatusCode.ToString().Equals("OK"))
{
var errorcode =
JObject.Parse(response.Content).GetValue("errorCode").ToString();
```

```
var errordescription =
JObject.Parse(response.Content).GetValue("errorMessage").ToString();
var httperrorcode =
JObject.Parse(response.Content).GetValue("httpCode").ToString();
var htttperrordescription =
JObject.Parse(response.Content).GetValue("httpMessage").ToString();
var message = errorcode + " " + htttperrordescription + " " +
errordescription;
var normalizedContent = new NormalizedFailResponse(false, message,
httperrorcode);
return new JavaScriptSerializer().Serialize(normalizedContent);
}
else
{
var emptyObjectArray = new object[1];
var accountObject = JObject.Parse(response.Content).ToString();
emptyObjectArray[0] = new
JavaScriptSerializer().DeserializeObject(accountObject);
var normalizedContent = new NormalizedSuccessResponse(true, "200",
emptyObjectArray, "");
return new JavaScriptSerializer().Serialize(normalizedContent);
}

}

}
}
```