ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σχεδιασμός και ανάπτυξη Android εφαρμογής ασφαλείας

Δημήτριος Δαμανάκης
Α.Μ. 711141085

Εισηγητής: ΙΩΑΝΝΗΣ ΒΟΓΙΑΤΖΗΣ

KLEIO SGOUROPOULOU
Digitally signed by KLEIO SGOUROPOULOU
Date: 2021.09.20 15:26:06 +03'00'

Ioannis Voyiatzis
Digitally signed by Ioannis Voyiatzis
Date: 2021.09.20 13:29:36 +03'00'

ΑΘΗΝΑ 2021

Georgios Meletiou
Digitally signed by Georgios Meletiou
Date: 2021.09.09 10:49:00 +03'00'

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σχεδιασμός και ανάπτυξη Android εφαρμογής ασφαλείας

Δημήτριος Δαμανάκης
Α.Μ. 711141085

Εισηγητής: ΙΩΑΝΝΗΣ ΒΟΓΙΑΤΖΗΣ

Εξεταστική Επιτροπή:
ΙΩΑΝΝΗΣ ΒΟΓΙΑΤΖΗΣ
ΣΓΟΥΡΟΠΟΥΛΟΥ ΚΛΕΙΩ
ΜΕΛΕΤΙΟΥ ΓΕΩΡΓΙΟΣ

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Δαμανάκης Δημήτριος του Εμμανουήλ, με αριθμό μητρώου 711141085 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών

Δαμανάκης Δημήτριος

(Κενό φύλλο)

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα διπλωματική εργασία ολοκληρώθηκε μετά από επίμονες προσπάθειες σε ένα ενδιαφέρον γνωστικό αντικείμενο, όπως αυτό της κρυπτογράφησης δεδομένων σε συσκευή Android. Την προσπάθειά μου αυτή υποστήριξε ο επιβλέπων καθηγητής μου, Μελετίου Γεώργιος, και η οικογένειά μου τους οποίους και θα ήθελα να ευχαριστήσω.

(Κενό φύλλο)

ΠΕΡΙΛΗΨΗ

Είναι φανερό ότι στις μέρες μας το διαδίκτυο παγιώνεται όλο και περισσότερο στην καθημερινή ζωή των ανθρώπων μέσω των προσωπικών ηλεκτρονικών λογαριασμών αλληλογραφίας καθώς και εφαρμογών κοινωνικής δικτύωσης και συνδρομών σε διαδικτυακές επιχειρήσεις παροχής υπηρεσιών. Έτσι γίνεται όλο και πιο επιτακτική η ανάγκη δημιουργίας μιας εφαρμογής η οποία θα εξασφαλίζει τόσο την άμεση πρόσβαση στα διαπιστευτήρια εισαγωγής στους υφιστάμενους λογαριασμούς όσο και την εμπιστευτικότητα και μοναδικότητα της πληροφορίας. Καθώς οι διαδικτυακές υπηρεσίες αυξάνονται ο κάθε χρήστης για να διαφυλάξει τους λογαριασμούς του χρειάζεται να έχει διαφορετικά διαπιστευτήρια για την είσοδο σε κάθε έναν από αυτούς πράγμα που καθιστά πολύ δύσκολη την απομνημόνευσή τους λόγω της πληθικότητάς τους. Η παρούσα διπλωματική εργασία ασχολείται με την ανάπτυξη εφαρμογής Android η οποία θα εξασφαλίζει ότι ο κάθε χρήστης θα μπορεί να αποθηκεύει τοπικά στην συσκευή του όλους τους κωδικούς του για κάθε υπηρεσία που χρησιμοποιεί (ηλεκτρονική αλληλογραφία, μέσα κοινωνικής δικτύωσης κλπ.) με ασφάλεια και με άμεση πρόσβαση σε αυτούς. Η αποθήκευση των παραπάνω στοιχείων θα γίνεται σε αρχείο αφού πρώτα θα έχουν κρυπτογραφηθεί μέσω της εφαρμογής. Ο αλγόριθμος που χρησιμοποιείται με σκοπό να επιτευχθεί η κρυπτογράφηση είναι ο AES με 128-bit μήκος κλειδιού σε CBC mode και PKCS5 padding.

ABSTRACT

It is obvious that nowadays internet is becoming more and more entrenched in people's daily lives through personal e-mail accounts as well as social networking applications and subscriptions to online service providers. Thus, the need to create an application that will ensure both direct access to credentials in existing accounts and the confidentiality and uniqueness of information becomes more and more urgent.As internet services grow every user needs to have different credentials to access each of them in order to safeguard their accounts which makes it very difficult to memorize them due to their population.This dissertation deals with the development of an Android application which will ensure that each user will be able to store locally on his device all his passwords for each service he uses (email, social media, etc.) safely and instantly access to them. The above data will be saved in a file after they have first been encrypted through the application. The algorithm used to achieve encryption is AES with 128-bit key length in CBC mode and PKCS5 padding.

ΠΕΡΙΕΧΟΜΕΝΑ

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

AES - AdvancedEncryptionStandard

ciphertext- το αποτέλεσμα κρυπτογράφησης που εκτελείται σε απλό κείμενο χρησιμοποιώντας έναν αλγόριθμο

NSA - National Security Agency

DES - Data Encryption Standard

NIST- National Institute of Standards and Technology

OpenSSL- βιβλιοθήκη λογισμικού για εφαρμογές που ασφαλίζουν τις επικοινωνίες μέσω δικτύων υπολογιστών

ΚΕΦΑΛΑΙΟ 1ο Κρυπτογράφηση – Ιστορική Αναδρομή

ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο αναλύεται το αντικείμενο της διπλωματικής εργασίας, παρουσιάζεται ο ορισμός της κρυπτογραφίας, αναλύεται ο αλγόριθμος κρυπτογράφησης AES και γίνεται μια ιστορική αναδρομή γύρω από τις περιόδους της κρυπτογραφίας.

1.1 Περιγραφή του αντικειμένου της διπλωματικής εργασίας

Αντικείμενο της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη εφαρμογής στην πλατφόρμα του Android με σκοπό την ασφαλή αποθήκευση εντός της συσκευής όλων των διαπιστευτηρίων που μπορεί να έχει ο κάθε χρήστης. Με αυτή την δυνατότητα ο χρήστης θα έχει άμεση πρόσβαση σε όλους τους αποθηκευμένους κωδικούς του εντός της εφαρμογής χωρίς την αναγκαιότητα απομνημόνευσης τους ή τη δυνατότητα πρόσβασης σε αυτούς χωρίς εξουσιοδότηση.

1.2Τι είναι η κρυπτογράφηση

Η κρυπτογράφηση ορίζεται ως μια διαδικασία κωδικοποίησης πληροφοριών δηλαδή μια αρχική αναπαράσταση πληροφοριών, γνωστή ως απλό κείμενο, μετατρέπεται σε μια εναλλακτική μορφή γνωστή ως ciphertext. Το ciphertext μπορεί να αποκρυπτογραφηθεί μόνο από εξουσιοδοτημένους χρήστες οι οποίοι και θα αποκτήσουν πρόσβαση στις αρχικές πληροφορίες σε ιδανικές περιπτώσεις. Το κάθε σχήμα κρυπτογράφησης, συνήθως, χρησιμοποιεί ένα ψευδό-τυχαίο κλειδί κρυπτογράφησης το οποίο δημιουργείται από έναν αλγόριθμο. Παρόλα αυτά, είναι δυνατόν να αποκρυπτογραφηθεί ένα μήνυμα χωρίς να γνωρίζουμε το κλειδί, αλλά για να γίνει αυτό σε ένα καλά σχεδιασμένο σχήμα κρυπτογράφησης απαιτούνται σημαντικοί υπολογιστικοί πόροι και γνώσεις. Η κρυπτογράφηση αποτελεί κλάδο της κρυπτολογίας, ενός διεπιστημονικού γνωστικού πεδίου το οποίο μπορεί να μελετηθεί ως όψη των εφαρμοσμένων μαθηματικών, της θεωρητικής πληροφορικής ή της επιστήμης ηλεκτρονικού μηχανικού. Η κρυπτολογία είναι πάρα πολύ σημαντική στους

τομείς της ασφάλειας των υπολογιστικών συστημάτων και των τηλεπικοινωνιών. Κύριος στόχος της είναι να παρέχει μηχανισμούς ώστε δύο ή περισσότερα άκρα επικοινωνίας (άνθρωποι, προγράμματα υπολογιστών, κλπ) να μπορούν να ανταλλάσσουν μηνύματα χωρίς κανένας τρίτος να είναι ικανός να διαβάζει τα μηνύματα αυτά. Στις πρώτες μορφές κρυπτογράφησης η επεξεργασία γινόταν πάνω στην γλωσσική δομή του μηνύματος ενώ πλέον χρησιμοποιεί το αριθμητικό ισοδύναμο. Τέλος, ένα σύνολο διαδικασιών κρυπτογράφησης – αποκρυπτογράφησης ονομάζεται κρυπτοσύστημα και αποτελείται από μια πεντάδα (P,C,k,E,D) όπου P ο χώρος των δυνατών μηνυμάτων , C ο χώρος των δυνατών κρυπτογραφημένων μηνυμάτων, k ο χώρος των δυνατών κλειδιών, E η κρυπτογραφική συνάρτηση και D ο μετασχηματισμός αποκρυπτογράφησης.

1.2 Περίοδοι κρυπτογραφίας

Η κρυπτογραφία έχει ξεκινήσει από τα αρχαία χρόνια και εξελίσσεται μέχρι και σήμερα. Έτσι από τον Μεσαίωνα (476-1492 μ.Χ.) έως το 1900 μ.Χ. έχουμε την πρώτη περίοδο κρυπτογραφίας. Κατά την διάρκεια αυτής της περιόδου αναπτύχθηκαν πολλοί αλγόριθμοι κρυπτογράφησης που βασίζονταν ωστόσο σε απλές αντικαταστάσεις γραμμάτων. Αυτό είχε αποτέλεσμα να μην χρειάζονται εξειδικευμένες γνώσεις και πολύπλοκες συσκευές και όπως προκύπτει από την κρυπτανάλυση αυτών των αλγορίθμων εάν είναι γνωστό ένα μεγάλο κομμάτι κρυπτογραφημένου μηνύματος τότε όλο το αρχικό κείμενο μπορεί να επανακτηθεί. Από μια σφηνοειδή επιγραφή που ανακαλύφθηκε στις όχθες του ποταμού Τίγρη οι πολιτισμοί που αναπτύχθηκαν στη Μεσοποταμία ασχολήθηκαν με την κρυπτογραφία ήδη από το 1500 π.Χ. Η επιγραφή αυτή περιγράφει μία μέθοδο κατασκευής σμάλτων για αγγειοπλαστική και θεωρείται ως το αρχαιότερο κρυπτογραφημένο κείμενο. Επιπλέον μια σφηνοειδής επιγραφή στην Σούσα της Περσίας η οποία περιλαμβάνει τους αριθμούς 1 έως 8 και από το 32 έως το 35, τοποθετημένους τον ένα κάτω από τον άλλο, ενώ απέναντι τους βρίσκονται τα αντίστοιχα για τον καθένα σφηνοειδή σύμβολα αποτελεί το αρχαιότερο βιβλίο κρυπτοκωδικών. Έπειτα, γύρω στο 5ο π.Χ. αιώνα έγινε η πρώτη στρατιωτική χρήση της κρυπτογράφησης από τους Σπαρτιάτες που εφηύραν την «σκυτάλη» , την πρώτη κρυπτογραφική συσκευή, και με την βοήθειά της κρυπτογραφούσαν με την μέθοδο της μετάθεσης. Η «σκυτάλη» ήταν μια ξύλινη ράβδος, ορισμένης διαμέτρου, γύρω από την

οποία ήταν τυλιγμένη ελικοειδώς μια λωρίδα περγαμηνής. Το κείμενο ήταν γραμμένο σε στήλες, ένα γράμμα σε κάθε έλικα, όταν δε ξετύλιγαν τη λωρίδα, το κείμενο ήταν ακατάληπτο εξαιτίας της αναδιάταξης των γραμμάτων. Το «κλειδί» ήταν η διάμετρος της σκυτάλης. Αργότερα, ο Ιούλιος Καίσαρας έγραφε στον Κικέρωνα και σε άλλους φίλους του, αντικαθιστώντας τα γράμματα του κειμένου, με γράμματα, που βρίσκονται 3 θέσεις μετά, στο Λατινικό Αλφάβητο. Από αυτό το γεγονός, σήμερα, το σύστημα κρυπτογράφησης που στηρίζεται στην αντικατάσταση των γραμμάτων του αλφαβήτου με άλλα που βρίσκονται σε καθορισμένο αριθμό θέσης πριν ή μετά, λέγεται κρυπτοσύστημα αντικατάστασης του Καίσαρα. Το σύστημα αντικατάστασης του Καίσαρα, χρησιμοποιήθηκε ευρύτατα και στους επόμενους αιώνες. Η κρυπτογραφία, λόγω των στρατιωτικών εξελίξεων, σημείωσε σημαντική ανάπτυξη στους επόμενους αιώνες. Ο Ιταλός Giovanni Batista Porta, το 1563, δημοσίευσε το περίφημο για την κρυπτολογία βιβλίο «De furtivis literarum notis», με το οποίο έγιναν γνωστά τα πολυαλφαβητικά συστήματα κρυπτογράφησης και τα διγραφικά κρυπτογραφήματα, στα οποία, δύο γράμματα αντικαθίστανται από ένα. Σημαντικός εκπρόσωπος εκείνης της εποχής είναι και ο Γάλλος Vigenere, του οποίου ο πίνακας πολυαλφαβητικής αντικατάστασης, χρησιμοποιείται ακόμη και σήμερα. Αργότερα, ο Άγγλος αρχιτέκτονας και ερασιτέχνης αρχαιολόγος Μ.Βέντρις ασχολήθηκε με την Γραμμική Γραφή Β και ήταν ο πρώτος που κατάλαβε ότι επρόκειτο για κάποιο είδος ελληνικής γραφής, αλλά η άποψη του αυτή δεν έγινε δεκτή αρχικά από τους ειδικούς. Στη συνέχεια, όμως, αρκετοί προσχώρησαν στην άποψή του. Ένας από αυτούς ήταν ο κρυπταναλυτής Τζον Τσάντγουικ, ο οποίος, στη διάρκεια του πολέμου, είχε εργασθεί στην ανάλυση της γερμανικής κρυπτομηχανής Enigma. Προσπάθησε να μεταφέρει την πείρα του στην κρυπτανάλυση της Γραμμικής Β, αλλά χωρίς επιτυχία μέχρι τότε. Όμως, ο συνδυασμός των δύο επιστημόνων έφερε το πολυπόθητο αποτέλεσμα. Το 1953 κατέγραψαν τα συμπεράσματά τους στο μνημειώδες έργο «Μαρτυρίες για την ελληνική διάλεκτο στα μυκηναϊκά αρχεία», που έγινε το πιο διάσημο άρθρο κρυπτανάλυσης. Η αποκρυπτογράφηση της Γραμμικής Β απέδειξε ότι επρόκειτο για ελληνική γλώσσα, ότι οι Μινωίτες της Κρήτης μιλούσαν ελληνικά και ότι η δεσπόζουσα δύναμη εκείνη την εποχή ήταν οι Μυκήνες. Η αποκρυπτογράφηση της Γραμμικής Β θεωρήθηκε επίτευγμα ανάλογο της κατάκτησης του Έβερεστ, που συνέβη την ίδια ακριβώς εποχή. Για αυτό και έγινε γνωστή σαν το «Έβερεστ της Ελληνικής αρχαιολογίας». Περνώντας λοιπόν στην δεύτερη περίοδο κρυπτογραφίας (1900-1950) η κρυπτογραφία αναπτύσσεται

όσο δεν είχε αναπτυχθεί τα τελευταία 3000 χρόνια και αυτό γιατί υπήρχε εξαιρετικά μεγάλη ανάγκη για ασφάλεια κατά τη μετάδοση ζωτικών πληροφοριών μεταξύ των στρατευμάτων των χωρών στους δύο παγκόσμιους πολέμους. Τα κρυπτοσυστήματα αυτής της περιόδου αρχίζουν να γίνονται πολύπλοκα, και να αποτελούνται από μηχανικές και ηλεκτρομηχανικές κατασκευές, οι οποίες ονομάζονται «κρυπτομηχανές». Η κρυπτανάλυση τους, απαιτεί μεγάλο αριθμό προσωπικού, το οποίο εργαζόταν επί μεγάλο χρονικό διάστημα ενώ ταυτόχρονα γίνεται εξαιρετικά αισθητή η ανάγκη για μεγάλη υπολογιστική ισχύ. Παρά την πολυπλοκότητα που αποκτούν τα συστήματα κρυπτογράφησης κατά τη διάρκεια αυτής της περιόδου η κρυπτανάλυση τους είναι συνήθως επιτυχημένη. Οι Γερμανοί έκαναν εκτενή χρήση, σε διάφορες παραλλαγές, ενός συστήματος γνωστού ως Enigma Ωστόσο το 1932 ο Πολωνός Marian Rejewski κατάφερε χρησιμοποιώντας θεωρητικά μαθηματικά να παραβιάσει την πρώτη μορφή του γερμανικού στρατιωτικού συστήματος Enigma. Ήταν η μεγαλύτερη σημαντική ανακάλυψη στην κρυπτολογική ανάλυση της εποχής. Μέχρι και το 1939 οι Πολωνοί συνέχισαν να αποκρυπτογραφούν τα μηνύματα που βασίζονταν στην κρυπτογράφηση με το Enigma ώσπου εν τέλει ο γερμανικός στρατός έκανε κάποιες σημαντικές αλλαγές τις οποίες οι Πολωνοί δεν μπόρεσαν να τις παρακολουθήσουν επειδή η αποκρυπτογράφηση πλέον απαιτούσε περισσότερους πόρους από όσους μπορούσαν να διαθέσουν. Μετά από αυτήν την εξέλιξη οι Πολωνοί άρχισαν να μεταβιβάζουν την γνώση τους και μερικές μηχανές στους Βρετανούς και στους Γάλλους. Έτσι στο κέντρο της Βρετανικής Υπηρεσίας από/κρυπτογράφησης, στο Μπλέτσλεϊ Παρκ, ο Άλαν Τούρινγκ (Alan Turing), ο Γκόρντον Ουέλτσμαν (Gordon Welchman) και πολλοί άλλοι μαζί με την βοήθεια ενός υπολογιστή, που ονομάστηκε Colossus, συνεχώς αποκρυπτογραφούσαν τις διάφορες παραλλαγές του Enigma. Αργότερα και μετά το 1940 το αμερικανικό ναυτικό σε συνεργασία με Βρετανούς και Ολλανδούς επιστήμονες έσπασαν πολλά κρυπτοσυστήματα του Ιαπωνικού ναυτικού. Μεγάλη επιτυχία σημειώθηκε κατά την αποκρυπτογράφηση του JN-25 η οποία οδήγησε στην αμερικάνικη νίκη στη Ναυμαχία της Μιντγουέι όπως και στην εξόντωση του Αρχηγού του Ιαπωνικού Στόλου Ιζορόκου Γιαμαμότο. Πριν καν αρχίσει ο Β’ Παγκόσμιος Πόλεμος μια ομάδα του αμερικανικού στρατού, η αποκαλούμενη SIS, έσπασε το ασφαλέστερο ιαπωνικό διπλωματικό σύστημα κρυπτογράφησης μια ηλεκτρομηχανική συσκευή που ονομάζεται «Purple». Κατά την διάρκεια του δεύτερου παγκόσμιου πολέμου χρησιμοποιήθηκαν κρυπτομηχανές παρόμοιες με το

Enigmaαλλά με σημαντικές βελτιώσεις, δύο πιο γνωστές μηχανές ήταν το βρετανικό TypeX και το αμερικανικό SIGABA. Για την εμπόλεμη περίοδο η κρυπτομηχανή LCD Lacida που είχε φτιαχτεί από τους Πολωνούς κρατήθηκε μυστική ακόμα και από τον Rejewski ο οποίος όταν, το 1941, έλεγξε την ασφάλειά της μέσα σε λίγες μόνο ώρες κατάφερε να την «σπάσει». Έτσι φτάνουμε στην τρίτη περίοδο κρυπτογραφίας , 1950 έως σήμερα , η οποία χαρακτηρίζεται από την μεγάλη ανάπτυξη στους επιστημονικούς κλάδους των μαθηματικών, της μικροηλεκτρονικής και των υπολογιστικών συστημάτων. Ουσιαστικά τώρα ξεκινάει η εποχή της σύγχρονης κρυπτογραφίας με τον Claude Shannon οποίος μαζί με τον Warren Weaver, το 1949, δημοσιεύουν το έγγραφο «Θεωρία επικοινωνίας των συστημάτων μυστικότητας» (Communication Theory of Secrecy Systems) στο τεχνικό περιοδικό Bell System και λίγο αργότερα στο βιβλίο, «Μαθηματική Θεωρία της Επικοινωνίας» (Mathematical Theory of Communication). Επιπλέον, ο Claude Shannon εκτός από τις λοιπές του εργασίες πάνω στη θεωρία δεδομένων και επικοινωνίας καθιέρωσε και μια θεωρητική βάση για την κρυπτογραφία και την κρυπτανάλυση. Τότε, η κρυπτογραφία σταμάτησε να δημοσιεύεται ξανά και φυλασσόταν από μυστικές υπηρεσίες κυβερνητικών επικοινωνιών όπως η NSA. Όλα αυτά όμως άλλαξαν στα μέσα της δεκαετίας του ’70 όπου και έγιναν δύο σημαντικές δημόσιες πρόοδοι. Πρώτα δημοσιεύτηκε το σχέδιο προτύπου κρυπτογράφησης DES, στις 17 Μαρτίου 1975, στον ομοσπονδιακό κατάλογο της Αμερικής. Η IBM πρότεινε το σχέδιο DES στο Εθνικό Γραφείο Προτύπων (NIST) προκειμένου να αναπτυχθούν ασφαλείς ηλεκτρονικές εγκαταστάσεις επικοινωνίας για τράπεζες και άλλους οργανισμούς. Έτσι το 1977 υιοθετήθηκε και δημοσιεύτηκε το πρότυπο επεξεργασίας πληροφοριών DES ύστερα από την τροποποίησή του από την NSA (αναφέρεται τότε ως FIPS 46-3). Για πρώτη φορά ένας αλγόριθμος κρυπτογράφησης που έχει εγκριθεί από μια εθνική αντιπροσωπεία όπως η NSA είναι δημόσια προσιτός. Μερικά χρόνια αργότερα και μάλιστα το 2001 ο NIST ανήγγειλε το FIPS 197 και έτσι επίσημα πλέον ο DES αντικαταστάθηκε από τον AES. Παρόλο την αντικατάστασή του ο DES και κυρίως ασφαλέστερες παραλλαγές του, όπως 3DES ή TDES, χρησιμοποιούνται ακόμα και σήμερα σε πολλά εθνικά και οργανωτικά πρότυπα. Το βασικό μέγεθος των 56-bit έχει αποδειχθεί πως δεν μπορεί να αντισταθεί σε επιθέσεις ωμής βίας (bruteforce attack) καθώς μία τέτοια επίθεση κατάφερε να σπάσει τον DES σε 56 ώρες. Επομένως, και όλα τα μηνύματα και

πληροφορίες που έχουν αποσταλεί από το 1976 με την χρήση DES διατρέχουν σοβαρό κίνδυνο αποκρυπτογράφησης.

1.3 Σύγχρονη κρυπτογραφία

Στην σύγχρονη εποχή η κρυπτογράφηση επιτυγχάνεται με την χρήση αλγορίθμων που έχουν το κλειδί κρυπτογράφησης και αποκρυπτογράφησης οι οποίοι μετατρέπουν τα δεδομένα σε μία «ψηφιακή ασυναρτησία» μέσω της κρυπτογράφησης και στην συνέχεια στην αρχική μορφή μέσω της αποκρυπτογράφησης. Γενικότερα υπάρχει ο κανόνας ότι όσο μεγαλύτερο είναι το κλειδί τόσο πιο δύσκολο είναι κάποιος να σπάσει τον κώδικα και να δει τις κρυπτογραφημένες πληροφορίες. Αυτό ισχύει γιατί εάν ένας επιτιθέμενος προσπαθήσει να αποκρυπτογραφήσει ένα μήνυμα χωρίς να ξέρει το κλειδί θα πρέπει να δοκιμάσει όλα τα πιθανά κλειδιά με την χρήση ωμής βίας (bruteforce). Άρα, επειδή η κάθε δυαδική μονάδα πληροφοριών ή bit αποτελείται από τιμές 0 ή 1 ένα κλειδί 8-bit θα έχει 256 ή 2^8 πιθανά κλειδιά και ένα κλειδί 56-bitθα είχε 2^56 πιθανά κλειδιά. Ο αλγόριθμος DESόπως έχουμε προαναφέρει χρησιμοποιεί μήκος κλειδιού 56-bits και σε κάποια δοκιμαστικά μηνύματα που είχαν αποσταλεί έχουν σπάσει με την χρήση bruteforce. Ωστόσο, ο DES αντικαταστάθηκε από τον AES γύρω στα τέλη του 1990 όπου και ανακαλύφθηκε πως ο πλέον αποδεκτός τρόπος για να προχωρήσουν οι επιχειρήσεις ηλεκτρονικού εμπορίου ήταν ο συνδυασμός των 2 προηγούμενων συστημάτων δηλαδή του AESκαι του DES.Αργότερα, άρχισαν να πραγματοποιούνται και οι πρώτες ασφαλείς διαδικτυακές συναλλαγές με την χρήση ενός νέου πρωτοκόλλου το γνωστό έως σήμερα Secure Socket Layer ή SSL.

ΚΕΦΑΛΑΙΟ 2ο Είδη κρυπτοσυστημάτων

Τα κρυπτοσυστήματα χωρίζονται σε 2 μεγάλες κατηγορίες, τα κλασσικά κρυπτοσυστήματα (συμμετρικά) και τα μοντέρνα κρυπτοσυστήματα (ασύμμετρα). Στην πρώτη κατηγορία (συμμετρικά κρυπτοσυστήματα) το σύστημα προκειμένου να προβεί στην κρυπτογράφηση και αποκρυπτογράφηση χρησιμοποιεί ένα κοινό κλειδί. Έτσι, η ασφάλεια αυτών των αλγορίθμων βασίζεται στην μυστικότητα του κλειδιού. Οι συμμετρικοί αλγόριθμοι χωρίζονται και αυτοί σε δύο διαφορετικές κατηγορίες ανάλογα

με τον τρόπο κρυπτογράφησης των πληροφοριών. Από την μία είναι οι αλγόριθμοι που χωρίζουν το μήνυμα σε κομμάτια και κρυπτογραφούν το κάθε κομμάτι ξεχωριστά και ονομάζονται Δέσμης (BlockCiphers) και από την άλλη οι αλγόριθμοι που κρυπτογραφούν μία ροή μηνύματος αλλά δεν το χωρίζουν σε τμήματα και ονομάζονται Ροής (StreamCiphers). Αλγόριθμοι Δέσμης αποτελούν οι DES, BlowFish, CMEA, RC5, Triple-DES ενώ Ροής έχουμε τους ORYX, RC4, SEAL.

Προκειμένου να καλυφθεί η αδυναμία μεταφοράς κλειδιών που παρουσιάζουν τα συμμετρικά συστήματα δημιουργήθηκαν τα ασύμμετρα κρυπτοσυστήματα ή αλλιώς κρυπτοσυστήματα δημόσιου κλειδιού. Σε αυτά τα συστήματα υπάρχουν δύο είδη κλειδιών ένα ιδιωτικό και ένα δημόσιο όπου το δημόσιο είναι διαθέσιμο σε όλους ενώ το ιδιωτικό είναι μυστικό. Η σχέση που έχουν αυτά τα κλειδιά μεταξύ τους είναι πως ότι μπορεί να κρυπτογραφήσει το ένα τότε μπορεί να αποκρυπτογραφήσει μόνο το άλλο. Με την βοήθεια της ασύμμετρης κρυπτογραφίας δημιουργήθηκαν οι ψηφιακές υπογραφές και στην συνέχεια αναπτύχθηκε η Υποδομή Δημόσιου Κλειδιού και στα ψηφιακά πιστοποιητικά. Ασύμμετροι κρυπταλγόριθμοι αποτελούν οι RSA, DSA, Pailier, το πρότυπο ElGamal, το πρωτόκκολο Diffie-Hellman.

Επομένως, η κρυπτογραφία εξελίσσεται όλο και περισσότερο και η χρήση της γίνεται όλο και πιο επιτακτική καθώς αυξάνεται η ανάγκη για αξιόπιστη μεταφορά της πληροφορίας. Μερικές από τις χρήσεις που έχει σήμερα η κρυπτογραφία είναι στις ασφαλείς συναλλαγές στα τραπεζικά δίκτυα – ATM, στην κινητή τηλεφωνία, στα συστήματα συναγερμών, σε ιδιωτικά VPN δίκτυα, έξυπνες κάρτες, κρυπτονομίσματα κλπ.

2.1Publickey (δημόσιο κλειδί)

Μετά το 1976 και την δημοσίευση στην εφημερίδα New Directions in Cryptography από τους Whitfield Diffie και Martin Hellman εισάχθηκε μία νέα μέθοδος διανομής κρυπτογραφικών κλειδιών η οποία και έλυσε το πρόβλημα διανομής κλειδιών και έγινε γνωστή ως ανταλλαγή κλειδιών Diffie-Hellman. Έτσι, ξεκίνησε μια ανάπτυξη νέων αλγορίθμων κρυπτογράφησης των ασύμμετρων αλγορίθμων κλειδιών. Στην ασύμμετρη κλειδιών κρυπτογράφηση χρησιμοποιείται ένα ζευγάρι μαθηματικά σχετικών κλειδιών, καθένα από τα οποία αποκρυπτογραφεί την κρυπτογράφηση που εκτελείται χρησιμοποιώντας το άλλο. Ένας τέτοιου είδους αλγόριθμος είναι γνωστός

ως σύστημα δημόσιου ή ασύμμετρου κλειδιού. Ο κάθε χρήστης, επομένως, απαιτείται να έχει μόνο ένα ζεύγος κλειδιών ορίζοντας το ένα κλειδί του ζεύγους ως ιδιωτικό και το άλλο ως δημόσιο και εφόσον το ιδιωτικό κλειδί παραμένει μυστικό, το δημόσιο κλειδί μπορεί να παραμένει ευρέως γνωστό και να επαναχρησιμοποιείται για μεγάλο χρονικό διάστημα. Έτσι, μπορεί να πραγματοποιηθεί ασφαλή επικοινωνία μέσω ενός μη ασφαλούς καναλιού, μεταξύ δύο χρηστών, με την προϋπόθεση ο κάθε χρήστης να γνωρίζει το δικό του δημόσιο και ιδιωτικό κλειδί καθώς και το δημόσιο κλειδί του άλλου χρήστη. Η αποτελεσματικότητα των ασύμμετρων αλγορίθμων βασίζεται σε μία κατηγορία μαθηματικών προβλημάτων που ονομάζονται μονόδρομες συναρτήσεις, οι οποίες χρειάζονται μικρή υπολογιστική ισχύ για να εκτελεστούν αλλά αντιθέτως τεράστιες ποσότητες ισχύος για να αντιστραφούν όταν είναι δυνατή η αντιστροφή τους όπως για παράδειγμα όταν εύκολα μπορούμε να πολλαπλασιάσουμε δύο μεγάλους πρώτους αριθμούς αλλά είναι πολύ δύσκολο να βρεθούν οι παράγοντες του προϊόντος από τον προηγούμενου πολλαπλασιασμού. Ωστόσο πριν την δημόσια ανακοίνωση από τους Diffie και Hellman το 1976 μια βρετανική υπηρεσία πληροφοριών του Αρχηγείου Επικοινωνιών της Κυβέρνησης (GCHQ) ισχυρίστηκε, κυκλοφορόντας έγγραφα, πως είχε ήδη αναπτύξει την ασύμμετρη κρυπτογραφία κλειδιού.

2.2 Κατακερματισμός (hashing)

Ο κατακερματισμός αποτελεί μια μέθοδο για την γρήγορη κωδικοποίηση πληροφοριών. Σε μια συμβολοσειρά κειμένου αφού εφαρμοστεί ένας αλγόριθμος προκύπτει μία νέα συμβολοσειρά η οποία και αποτελεί το «ψηφιακό δακτυλικό αποτύπωμα» του μηνύματος καθώς η συγκεκριμένη τιμή προσδιορίζει μόνο ένα συγκεκριμένο μήνυμα. Με τη μέθοδο του κατακερματισμού μπορούμε εύκολα να προσδιορίσουμε εάν οι πληροφορίες έχουν αλλάξει κατά την διαδικασία της μετάδοσής τους κοιτάζοντας απλά την τιμή κατακερματισμού η οποία θα πρέπει να είναι ίδια τόσο πριν την αποστολή όσο και μετά. Ωστόσο ο κατακερματισμός δεν είναι ίδιος με την κρυπτογράφηση καθώς αποτελεί μία μονόδρομη λειτουργία ενώ η κρυπτογράφηση είναι αμφίδρομη αφού μετατρέπει ένα απλό κείμενο σε κρυπτογραφημένο και αντίστροφα. Ο κατακερματισμός εφαρμόζεται σε επαλήθευση ψηφιακών υπογραφών μέσω Διαδικτύου, σε κωδικούς πρόσβασης υπολογιστών και διαδικτυακές εφαρμογές.

ΚΕΦΑΛΑΙΟ3ο Advanced Encryption Standard (AES)

Γνωστό και ως Rijndael το Advanced Encryption Standard ιδρύθηκε το 2001 από το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογίας των ΗΠΑ και αποτελεί μια προδιαγραφή για την κρυπτογράφηση ηλεκτρονικών δεδομένων. Στην πραγματικότητα το AES είναι ένα υποσύνολο του Rijndael που κατασκευάστηκε από τους κρυπτογράφους Vincent Rijmen και Joan Daemen. Αν και το Rijndael είχε διαφορετικά μεγέθη κλειδιών και μπλοκ για τον AES, επιλέχθηκε από τον NIST, να έχει μπλοκ 128bit και τρία διαφορετικά μήκη κλειδιού 128, 192 και 256 bit. Πλέον, το AES χρησιμοποιείται παγκοσμίως και έχει αντικαταστήσει το Πρότυπο Κρυπτογράφησης Δεδομένων (DES) και αποτελεί έναν αλγόριθμο συμμετρικού κλειδιού όπου το κλειδί χρησιμοποιείται τόσο στην κρυπτογράφηση όσο και στην αποκρυπτογράφηση των δεδομένων. Το πρότυπο ISO/IEC 18033-3 το οποίο καθορίζει συστήματα κρυπτογράφησης για σκοπούς εμπιστευτικότητας των δεδομένων περιλαμβάνει το AES. Αποτελεί την πρώτη και τη μόνη, έως σήμερα, ελεύθερη για όλους κρυπτογράφηση που είναι εγκεκριμένη από την Υπηρεσία Εθνικής Ασφάλειας των ΗΠΑ (NSA). Η αρχή σχεδιασμού του AES είναι γνωστή ως δίκτυο υποκατάστασης-παραλλαγής και σε αντίθεση με τον DES δεν χρησιμοποιεί δίκτυο Feistel. Σε έναν κρυπτογράφο AES το μέγεθος κλειδιού που χρησιμοποιείται καθορίζει τον αριθμό των κύκλων μετασχηματισμών της εισόδου δηλαδή για κλειδιά 128-bit έχουμε 10 γύρους, για κλειδιά 192-bit 12 γύρους και για κλειδιά 256-bit 14 γύρους. Ο κάθε γύρος αποτελεί ένα σύνολο διάφορων σταδίων επεξεργασίας συμπεριλαμβανομένου και ενός που εξαρτάται από το ίδιο το κλειδί κρυπτογράφησης. Αντίστροφα για την αποκρυπτογράφηση εφαρμόζονται πάλι κάποια σύνολα γύρων που μετατρέπουν το κρυπτογραφημένο κείμενο σε αρχικό πάντα με την χρήση του ίδιου κλειδιού κρυπτογράφησης.

3.1Περιγραφή υψηλού επιπέδου του αλγόριθμου

Αρχικά γίνεται KeyExpansion (χρησιμοποιώντας το πρόγραμμα κλειδιών AES κυκλικά κλειδιά προέρχονται από το κλειδί κρυπτογράφησης) και μετά εφαρμόζεται το AddRoundKey (χρησιμοποιώντας bitwise xor κάθε byte της κατάστασης συνδυάζεται

με ένα byte του κυκλικού κλειδιού). Έπειτα στους 9, 11 ή 13 γύρους εφαρμόζονται με τη σειρά SubBytes (κάθε byte αντικαθίσταται από ένα άλλο σύμφωνα με ένα πίνακα αναζήτησης μέσω ενός μη γραμμικού βήματος), ShiftRows (σε αυτό το βήμα μεταφοράς οι τρεις τελευταίες σειρές της κατάστασης μετατοπίζονται κυκλικά με συγκεκριμένο αριθμό βημάτων), MixColumns (αποτελεί βήμα γραμμικής ανάμιξης που λειτουργεί στις στήλες της κατάστασης και συνδυάζει τα 4 byte σε κάθε στήλη) και τέλος AddRoundKey. Για να φτάσει και να ολοκληρώσει τους 10, 12 ή 14 γύρους αρχικά εφαρμόζεται SubBytes μετά MixColumns και μετά AddRoundKey.

3.2 Βελτιστοποίηση λειτουργίας κρυπτογράφησης

Η εκτέλεση του cipher μπορεί να γίνει ταχύτερη, σε συστήματα με 32-bit ή μεγαλύτερες λέξεις, συνδυάζοντας τα βήματα «SubBytes» και «ShiftRows», που προαναφέρθηκαν, με το βήμα «MixColumns” δημιουργώντας έτσι μια ακολουθία αναζητήσεων πίνακα. Για να γίνει όμως αυτό εφικτό απαιτούνται τέσσερεις πίνακες των 32-bit με 256 είσοδο σύνολο 4096 bytes. Έτσι ένας γύρος μπορεί να εκτελεστεί με 16 αναζητήσεις στον πίνακα και 12 λειτουργίες από 32-bitexclusive-or κατά την εκτέλεση του «AddRoundKey».

3.3 Γνωστές επιθέσεις

Ένα κρυπτογραφικό «σπάσιμο» αποτελεί οτιδήποτε είναι πιο γρήγορο από μία επίθεση ωμής βίας (bruteforce) στην οποία δοκιμάζονται όλα τα πιθανά κλειδιά. Το 2006, η distributed.net, κατάφερε την μεγαλύτερη επιτυχημένη δημόσια γνωστή επίθεση brute-force εναντίον ενός κλειδιού 64-bit RC5. Αφού ο χώρος του κλειδιού αυξάνεται κατά έναν συντελεστή 2 για κάθε μήκος του κλειδιού και κάθε δυνατό κλειδί είναι εξοπλισμένο έχουμε στην πραγματικότητα διπλασιασμό του μέσου χρόνου αναζήτησης μέσω brute-force στο κλειδί δηλαδή εκθετική αύξηση της προσπάθειας που καταβάλλεται όσο αυξάνεται το μήκος του κλειδιού. Παρόλα αυτά το μήκος του κλειδιού αυτό κάθε αυτό δεν συνεπάγεται την ασφάλεια έναντι επιθέσεων όπως έχει αποδειχθεί και σε επιθέσεις που έχουν γίνει σε κρυπτογράφους με πολύ μεγάλα κλειδιά. Το 2002 ο Nicolas Courtois και ο Josef Pieprzyk ανακοίνωσαν μια θεωρητική επίθεση, γνωστή ως επίθεση «XSL», με σκοπό να αναδείξουν την αδυναμία του αλγόριθμου AES λόγω της χαμηλής πολυπλοκότητας των μη γραμμικών του στοιχείων

όμως πολλά έγγραφα έχουν δείξει πως, αυτού του είδους η επίθεση, όπως παρουσιάστηκε είναι ανεφάρμοστη. Έτσι, μέχρι το 2009, οι επιθέσεις μέσω καναλιού σε συγκεκριμένες εφαρμογές του AES ήταν οι μοναδικές δημοσιευμένες επιτυχείς. Τότε όμως, ανακαλύφθηκε μία νέα επίθεση που συσχετιζόταν με το κλειδί, είχε πολυπλοκότητα 2^{119} , και εκμεταλλευόταν την απλότητα του βασικού προγράμματος του AES. Αργότερα, βελτιώθηκε η πολυπλοκότητα της επίθεσης σε $2^{99,5}$. Ωστόσο, οι επιθέσεις που αφορούν το κλειδί δεν προκαλούν ανησυχία σε κανένα σωστά σχεδιασμένο κρυπτογραφικό πρωτόκολλο καθώς αυτό δεν θα επιτρέπει τα σχετικά κλειδιά και έτσι θα περιορίζει τον επιτιθέμενο. Μία ακόμα επίθεση δημοσιευμένη από τον Bruce Schneier και υλοποιημένη από τον Alex Biryukov, Orr Dunkelman και τον Dmitry Khovratovich, το 2009, εναντίον του AES-256 χρησιμοποιώντας μόνο δύο σχετικά κλειδιά και 2^{39} χρόνο για να ανακτήσει το πλήρες 256-bit κλειδί 9 κύκλων έκδοσης, ή 2^{45} χρόνο για ένα 10 γύρων έκδοση ή 2^{70} χρόνο για ένα 11 γύρων έκδοση. Παρόλα αυτά όμως ο AES χρησιμοποιεί 14 γύρους άρα αυτές οι επιθέσεις δεν έχουν αποτελεσματικότητα έναντι του πλήρους AES. Επόμενο ήταν λοιπόν, να επικριθεί η πρακτικότητα επιθέσεων με ισχυρότερα πλήκτρα. Τον Νοέμβρη του 2009, δημοσιεύτηκε ως πρώτη επίθεση εναντίον του 8-γύρων έκδοση του AES-128. Η επίθεση αυτή αποτελεί μια βελτιωμένη έκδοση της επίθεσης ξεκινώντας από την μέση «start-from-the-middle attack» η οποία εφαρμοζόταν σε παραλλαγές του AES. Λειτουργεί στην 8 γύρων έκδοση του AES-128 με χρονική πολυπλοκότητα 2^{48} και πολυπλοκότητα μνήμης 2^{32}. Επίσης αυτή η επίθεση είναι μη αποτελεσματική έναντι του πλήρους AES-128 διότι ο πλήρες AES-128 χρησιμοποιεί 10 γύρους. Πρώτοι λοιπόν, ο AndreyBogdanov,ο DmitryKhovratovich και ο ChristianRechberger, το 2011 έκαναν επιθέσεις ανάκτησης κλειδιών σε πλήρη AES. Η επίθεση ήταν μια επίθεση biclique η οποία είναι ταχύτερη, περίπου τέσσερεις φορές, από μία brute-force επίθεση. Αυτή η επίθεση απαιτεί $2^{126.2}$ λειτουργίες για την ανάκτηση AES-128 κλειδιού, $2^{190.2}$ για AES-192 και $2^{254.6}$ για AES-256. Ο προηγούμενος αριθμός των λειτουργιών που απαιτούνται μειώθηκε λίγο αργότερα χωρίς ωστόσο να σημειωθεί κανένα μεγάλο κέρδος καθώς ένα κλειδί 126-bit χρειάζεται ακόμα κα σήμερα δισεκατομμύρια χρόνια για να «σπάσει» με την χρήση brute-force πόσο μάλλον ένα κλειδί των 128-bits. Έτσι υπολογίζεται ότι χρειάζονται 38 τρισεκατομμύρια terabytes δεδομένων τα οποία είναι περισσότερα από όλα τα δεδομένα που είχαν αποθηκευτεί στους υπολογιστές το 2016. Προς το παρόν, δεν υπάρχει καμία γνωστή πρακτική επίθεση που θα επέτρεπε σε

κάποιον που δεν έχει την γνώση του κλειδιού να διαβάσει δεδομένα κρυπτογραφημένα από τον AES.

3.4 Επιθέσεις πλευρικού καναλιού (Side-channel attacks)

Τέτοιου τύπου επιθέσεις δεν πραγματοποιούνται εναντίον του κρυπτογράφου και επομένως δεν σχετίζονται με την ασφάλεια της κρυπτογράφησης αυτήν κάθε αυτήν ωστόσο αποτελούν ένα πολύ σημαντικό κομμάτι. Οι επιθέσεις αυτές πραγματοποιούνται σε υλοποιήσεις του αλγόριθμου πάνω σε hardware ή software συστήματα τα οποία διαρρέουν δεδομένα λόγω λάθους υλοποίησης. Ο D.J. Bernstein, το 2005, ανακοίνωσε μια επίθεση χρονισμού προσωρινής μνήμης με την οποία κατάφερε να «σπάσει» έναν διακομιστή που χρησιμοποιούσε κρυπτογράφηση AES του OpenSSL. Ο διακομιστής ήταν σχεδιασμένος να παρέχει όσο το δυνατόν περισσότερες πληροφορίες σχετικά με τον χρονισμό δηλαδή έδινε στοιχεία για τον αριθμό των κύκλων που χρειαζόταν για την κρυπτογράφηση. Έτσι, τον Οκτώβρη του 2005, ο Dag Arne Osvik, ο Adi Shamir και ο Eran Tromer δημοσίευσαν ένα έγγραφο επιδεικνύοντας πολλές επιθέσεις χρονισμού προσωρινής μνήμης κατά των εφαρμογών του AES στο OpenSSL και στην λειτουργία κρυπτογράφησης διαμερισμάτων (dm-crypt) σε Linux συστήματα. Μια επίθεση πυροδοτώντας συνολικά 800 λειτουργίες κρυπτογράφησης, μέσα 65 milliseconds, κατάφερε να αποκτήσει ένα ολόκληρο κλειδί AES ωστόσο αυτή η επίθεση προϋποθέτει ο επιτιθέμενος να έχει την δυνατότητα να τρέξει προγράμματα στο ίδιο σύστημα που εκτελεί την κρυπτογράφηση AES. Το 2009, δημοσιεύτηκε μια επίθεση η οποία στόχευε μερικές hardware υλοποιήσεις και χρησιμοποιούσε την ανάλυση διαφορικών σφαλμάτων ώστε να ανακτήσει κλειδί με πολυπλοκότητα 2^{32}. Επίσης ένα ακόμη έγγραφο δημοσιεύτηκε το 2010 από τους David Gullasch και Stephan Krenn, όπου περιέγραφε μια προσέγγιση για μια «σχεδόν πραγματικού χρόνου» επίθεση ανάκτησης μυστικών κλειδιών από το AES-128 χωρίς την ανάγκη ούτε απλού ούτε κρυπτογραφημένου κειμένου, επίθεση η οποία μπορεί να χρησιμοποιηθεί και εναντίον εφαρμογών AES-128 που χρησιμοποιούν πίνακες συμπίεσης όπως δηλαδή το OpenSSL. Παρόμοια όμως και αυτή η επίθεση προϋποθέτει από τον επιτιθέμενο να μπορεί να εκτελέσει μη-προνομιακό κώδικα στο μηχάνημα που εκτελεί την κρυπτογράφηση AES, το οποίο βέβαια μπορεί να επιτευχθεί με μόλυνση από κακόβουλο λογισμικό. Επίσης, τον

Μάρτη του 2016, οι Ashokkumar C., Ravi Prakash Giri και Bernard Menezes παρουσίασαν μία επίθεση σε υλοποιήσεις του AES όπου μπορούσαν να ανακτήσουν το κλειδί σε έναν ολόκληρο 128-bit AES με μόνο 6-7 μπλοκ απλού/κρυπτογραφημένου κειμένου. Αυτή η επίθεση αποτελούσε τεράστια βελτίωση σε σχέση με τις προηγούμενες υλοποιήσεις που χρειαζόντουσαν από 100 έως 1 εκατομμύριο κρυπτογραφήσεις. Πλέον, πολλές σύγχρονες CPU διαθέτουν ενσωματωμένους drivers για το AES και οι οποίες προστατεύουν από Side-channel attacks.

ΚΕΦΑΛΑΙΟ4ο Android

Το Android αποτελεί ένα τροποποιημένο λειτουργικό σύστημα βασισμένο στον πυρήνα του Linux και είναι σχεδιασμένο κυρίως για συσκευές με οθόνη αφής όπως smartphone και tablet. Με την χρηματοδότηση της Google η Open Handset Alliance, μια κοινοπραξία προγραμματιστών, ανέπτυξαν το Android τον Νοέμβρη του 2007 με την πρώτη εμπορική συσκευή Android να κυκλοφορεί τον Σεπτέμβρη του 2008. Ουσιαστικά, αποτελεί ένα δωρεάν λογισμικό του οποίου ο ανοιχτός πηγαίος κώδικας είναι γνωστός ως Android Open Source Project (AOSP) με κύρια άδεια χρήσης από την Apache. Πλέον, σχεδόν όλες οι συσκευές Android αποστέλλονται με πρόσθετο προεγκατεστημένο λογισμικό όπως τα Google Mobile Services (GMS) , την ψηφιακή πλατφόρμα διανομής GooglePlay και τα GooglePlay Services. Ο πηγαίος κώδικας έχει χρησιμοποιηθεί για ανάπτυξη λειτουργικών συστημάτων σε άλλες συσκευές όπως κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, υπολογιστές με τα πιο γνωστά από αυτά να είναι το AndroidTV για τις τηλεοράσεις και το WearOS για φορητές συσκευές. Τα πακέτα λογισμικού Android έχουν μορφή APK και γίνονται διαθέσιμα στις συσκευές μέσω εφαρμογών «καταστημάτων» όπως το GooglePlay Store, το SamsungGalaxy Store, το Aptoide και το F-Droid. Από το 2011 οι μεγαλύτερες πωλήσεις παγκοσμίως σε OS ήταν του Android το οποίο σήμερα έχει μια τρέχουσα σταθερή έκδοση το Android 11.

4.1 Εφαρμογές

Οι περισσότερες, αν όχι όλες, συσκευές Android έρχονται με προ εγκατεστημένες εφαρμογές Google όπως το Gmail, Google Maps, Google Chrome και το Youtube. Οι εφαρμογές δίνουν περισσότερες δυνατότητες και λειτουργίες σε μια

Android συσκευή και είναι γραμμένες χρησιμοποιώντας κάποιο Android software development (SDK) και την Kotlin ως γλώσσα προγραμματισμού η οποία αντικατέστησε την Java από το 2019. Το SKD περιλαμβάνει ένα μεγάλο set από εργαλεία ανάπτυξης λογισμικού όπως debugger, διάφορες χρήσιμες βιβλιοθήκες, προσομοιωτή Android συσκευής, documentation και παραδείγματα με κώδικα. Αρχικά η Google υποστήριζε το Eclipse, με την προσθήκη του πρόσθετου Android Development Tools, ως ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) μέχρι το 2014 όπου και παρουσίασε το Android Studio βασισμένο στο Intellij IDEA. Εκτός από το SDK υπάρχουν και άλλα εργαλεία ανάπτυξης όπως το native development kit (NDK) για εφαρμογές σε γραμμένες σε C ή C++. Το 2014 η Google παρουσίασε ένα framework βασισμένο στο Apache Cordova στο οποίο δινόταν η δυνατότητα εισαγωγής HTML 5 δικτυακής εφαρμογής στο Android. Επίσης, το 2014 παρουσιάστηκε και το Firebase το οποίο περιείχε βοηθητικά εργαλεία για ανάπτυξη εφαρμογών. Οι συσκευές Android συνήθως χρησιμοποιούν μπαταρία για να λειτουργήσουν έτσι το λογισμικό τους είναι σχεδιασμένο ώστε να ελαχιστοποιεί την κατανάλωση ενέργειας. Όταν μια εφαρμογή δεν χρησιμοποιείται τότε το σύστημα την θέτει σε αδράνεια με σκοπό να μην σπαταλάει ενέργεια από την μπαταρία ή πόρους από τον επεξεργαστή ωστόσο είναι άμεσα διαθέσιμη όταν ο χρήστης την χρειαστεί ξανά.

4.2 Interface (διεπαφή)

Στο Android η κύρια διεπαφή χρήστη βασίζεται κυρίως στον άμεσο χειρισμό με την χρήση αφής. Οι είσοδοι αφής αντιστοιχούν σε πραγματικές ενέργειες, όπως για παράδειγμα κύλιση σελίδας, pinching, and reverse pinching εικόνων και συμπληρωματικά ένα εικονικό πληκτρολόγιο. Επίσης, το Android υποστηρίζει και φυσικά πληκτρολόγια και controllers παιχνιδιών με την βοήθεια μιας σύνδεσης Bluetooth ή USB. Οι συσκευές Android περιλαμβάνουν γυροσκόπια, επιταχυνσιόμετρα, γυροσκόπια και αισθητήρες εγγύτητας με την βοήθεια των οποίων εκτελούνται κάποιες συγκεκριμένες ενέργειες όπως προσανατολισμός της οθόνης σε οριζόντιο ή κάθετο άξονα, ηλεκτρονική πυξίδα και κατεύθυνση οχήματος σε ένα παιχνίδι περιστρέφοντας μόνο την συσκευή. Η αρχική οθόνη μιας Android συσκευής αποτελείται από εικονίδια εφαρμογών και γραφικά στοιχεία. Τα εικονίδια εφαρμογών

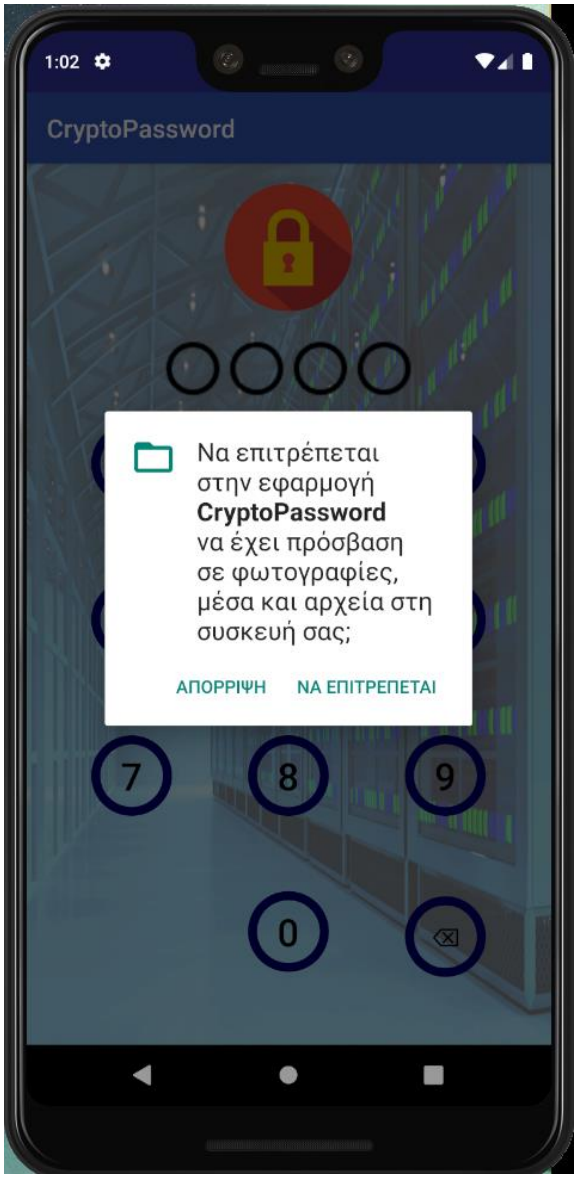
είναι στην ουσία τα εικονίδια που όταν τα πατήσει ο χρήστης τότε ξεκινάει η εν λόγω εφαρμογή ενώ τα γραφικά στοιχεία εμφανίζουν περιεχόμενο, το οποίο ενημερώνεται αυτόματα και είναι συνήθως μια πρόγνωση καιρού, το mail ενός χρήστη ή κάποιες ειδήσεις

ΚΕΦΑΛΑΙΟ5ο Πρακτικό μέρος – κώδικας σε Java και στιγμιότυπα

Για την υλοποίηση της διπλωματικής αυτής εργασίας χρησιμοποιήθηκε το AndroidStudio (4.1.2). Παρακάτω σας παραθέτω στιγμιότυπα του περιβάλλοντος της εφαρμογής που ανέπτυξα, τα απαραίτητα αρχεία και κλάσεις που την απαρτίζουν καθώς και μια σύντομη περιγραφή επεξήγησης της λειτουργικότητας του κώδικα.

5.1 Αρχική οθόνη LogIn

Όταν ανοίγουμε πρώτη φορά την εφαρμογή στην αρχική οθόνη εμφανίζεται ένα μήνυμα για το αν ο χρήστης επιτρέπει στην εφαρμογή να έχει πρόσβαση στις φωτογραφίες, μέσα και αρχεία της συσκευής (εικόνα 1.1). Στην συνέχεια, εμφανίζεται ένα πληκτρολόγιο πάνω στην οθόνη και ένα μήνυμα προτροπής αρχικοποίησης ενός 4ψήφιου PIN (εικόνα 1.2). Το PIN αυτό αποτελεί τον κωδικό με τον οποίο ένας χρήστης θα μπαίνει μέσα στην εφαρμογή και θα μπορεί να χρησιμοποιήσει τις βασικές της λειτουργίες όπως δημιουργία νέου wallet κωδικών, εισαγωγή wallet κωδικών στην εφαρμογή ή δημιουργία wallet κωδικών μέσω excel αρχείου. Επίσης, ο χρήστης μπορεί να κάνει είσοδο στην εφαρμογή χωρίς να εισάγει PIN καθώς η εφαρμογή υποστηρίζει την είσοδο με χρήση δακτυλικού αποτυπώματος όπως θα δούμε και παρακάτω.

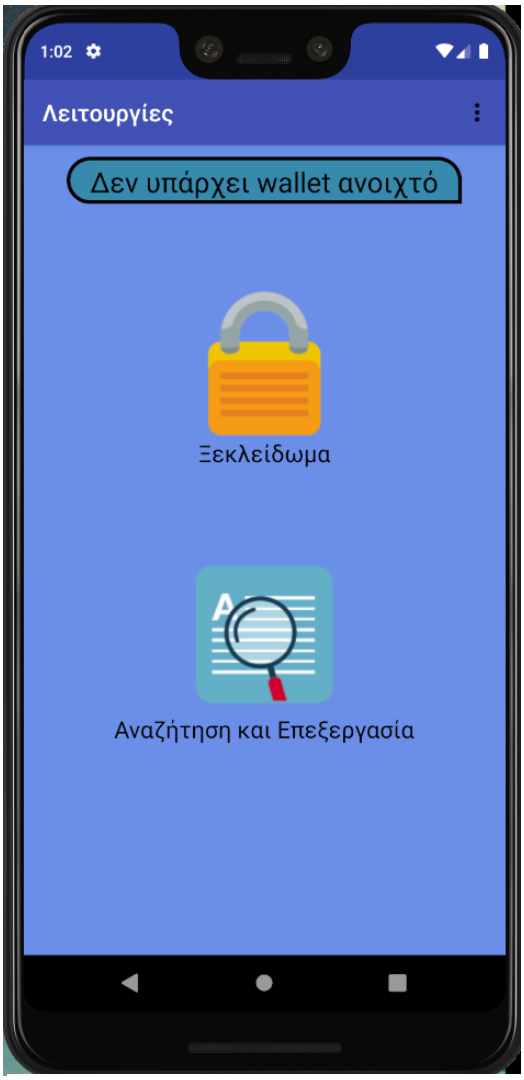


Εικόνα 1.1

Εικόνα 1.2

5.2 Πρώτη οθόνη επιλογών

Εφόσον ο χρήστης πραγματοποιήσει επιτυχημένη είσοδο στην εφαρμογή τότε βλέπει το menu των βασικών επιλογών (εικόνα 2.1) . Ψηλά στην οθόνη μας παρέχεται η πληροφορία για το wallet κωδικών που έχουμε ανοιχτό ή ότι δεν έχουμε κάποιο wallet ξεκλείδωτο και στην συνέχεια έχουμε τις επιλογές ξεκλειδώματος ενός wallet και αναζήτηση/εμφάνιση και επεξεργασία των credentials που είναι αποθηκευμένα σε ένα wallet. Επιπλέον, στην πάνω δεξιά γωνία της οθόνης υπάρχει το υπο-μενού με επιπλέον λειτουργίες της εφαρμογής όπως αλλαγή του αρχικού PIN, ενεργοποίηση/απενεργοποίηση της χρήσης δακτυλικού αποτυπώματος, δημιουργία νέου wallet κωδικών, εισαγωγή/εξαγωγή wallet κωδικών ή excel κωδικών κ.α. (εικόνα 2.2) .



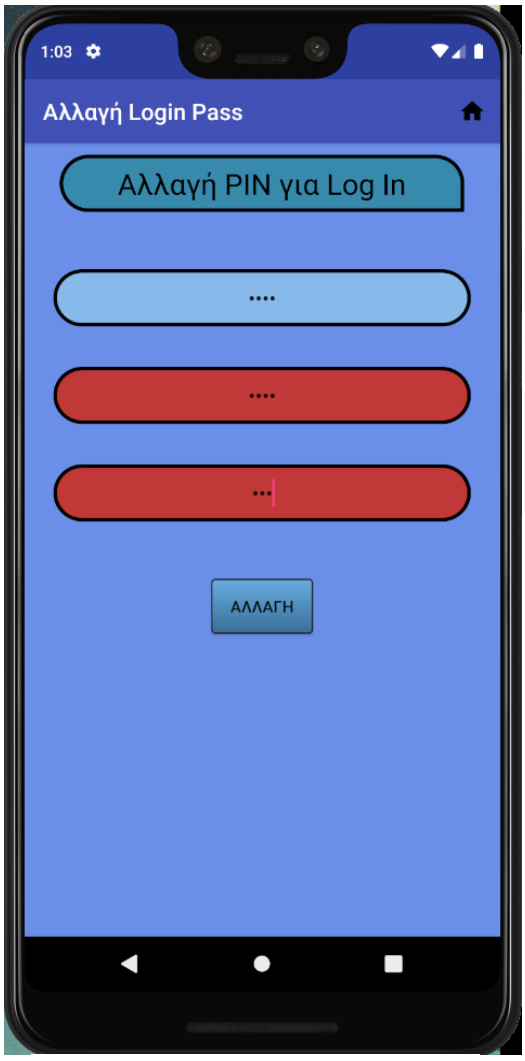
Εικόνα 2.1

Εικόνα 2.2

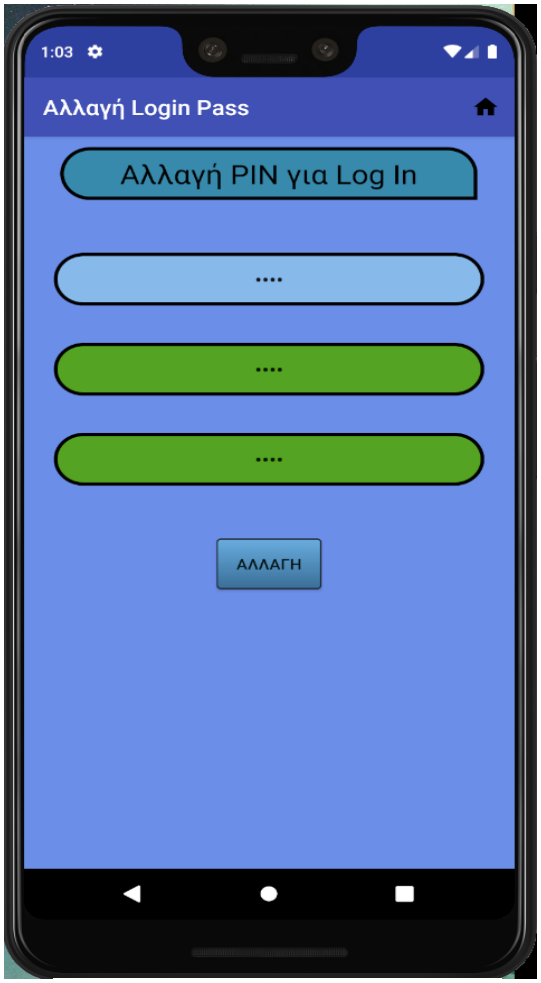
5.3 Αλλαγή LogInPIN

Ο χρήστης επιλέγοντας την πρώτη λειτουργία από το υπο-μενού μεταφέρεται σε μια νέα δραστηριότητα (εικόνα 3.1) προκειμένου να αλλάξει το PIN με το οποίο κάνει είσοδο στην εφαρμογή. Εκεί εισάγει το ήδη υπάρχον PIN και το νέο PIN μαζί με την επαλήθευσή του. Εάν το αρχικό PIN δεν είναι σωστό η εφαρμογή βγάζει το κατάλληλο μήνυμα. Επίσης, η εφαρμογή μας δείχνει πότε το νέο PIN που έχουμε εισάγει δεν ταιριάζει με την επαλήθευσή του αλλάζοντας χρώμα στο πεδίο: πράσινο αν συμπίπτουν και κόκκινο αν διαφέρουν (εικόνα 3.2,3.3).

Εικόνα 3.1



Εικόνα 3.2



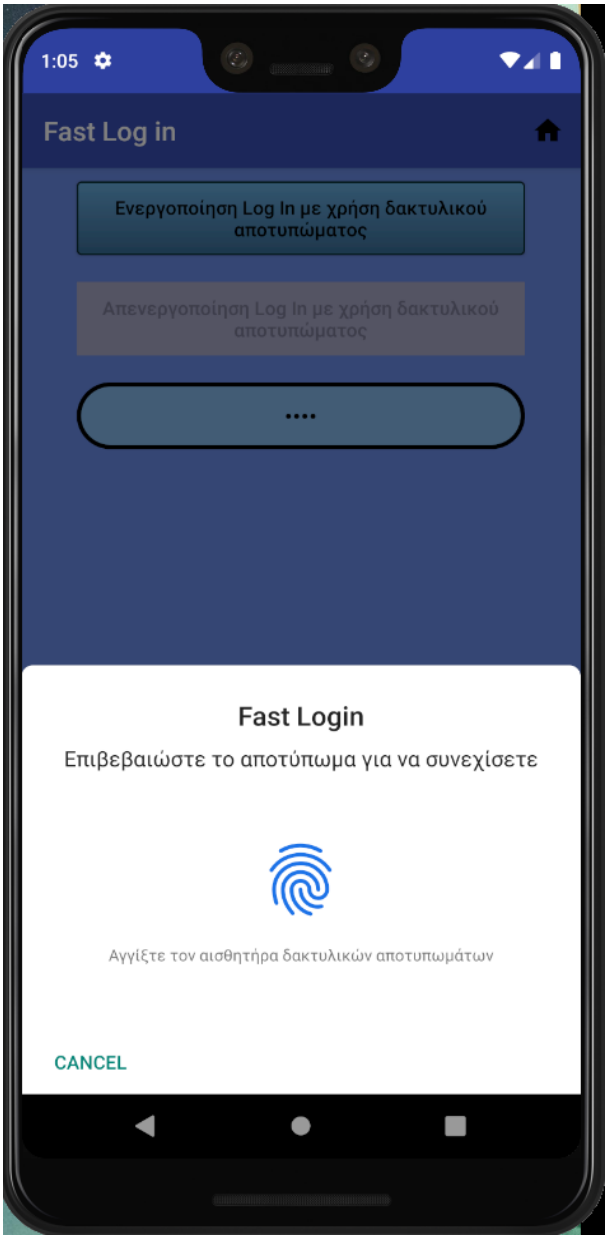
Εικόνα 3.3

5.4 FastLogin

Η δραστηριότητα FastLogin είναι μια διαδικασία όπου ο χρήστης μπορεί να ενεργοποιήσει ή απενεργοποιήσει την δυνατότητα εισόδου στην εφαρμογή με την χρήση δακτυλικού αποτυπώματος (εικόνα 4.1). Προϋπόθεση είναι η συσκευή που τρέχει η εφαρμογή να έχει την δυνατότητα ελέγχου δακτυλικού αποτυπώματος και ο χρήστης να έχει καταγράψει κάποιο δακτυλικό αποτύπωμα. Έτσι, σε αυτή την δραστηριότητα εμφανίζεται ένα πεδίο όπου ο χρήστης πρέπει να εισάγει το PIN της εφαρμογής και αν είναι σωστό στην συνέχεια εισάγει το δακτυλικό του αποτύπωμα (εικόνα 4.2). Αν κάνει τα παραπάνω επιτυχώς τότε το κουμπί της Ενεργοποίησης γίνεται disabled και το κουμπί της Απενεργοποίησης enabled (εικόνα 4.3) ενώ στην αρχική οθόνη του LogIn εμφανίζεται ένα εικονίδιο με το σήμα του δακτυλικού

αποτυπώματος το οποίο πατάει ο χρήστης και εμφανίζεται η επιλογή για είσοδο του δακτυλικού αποτυπώματος του (εικόνα 4.4).

Εικόνα 4.1



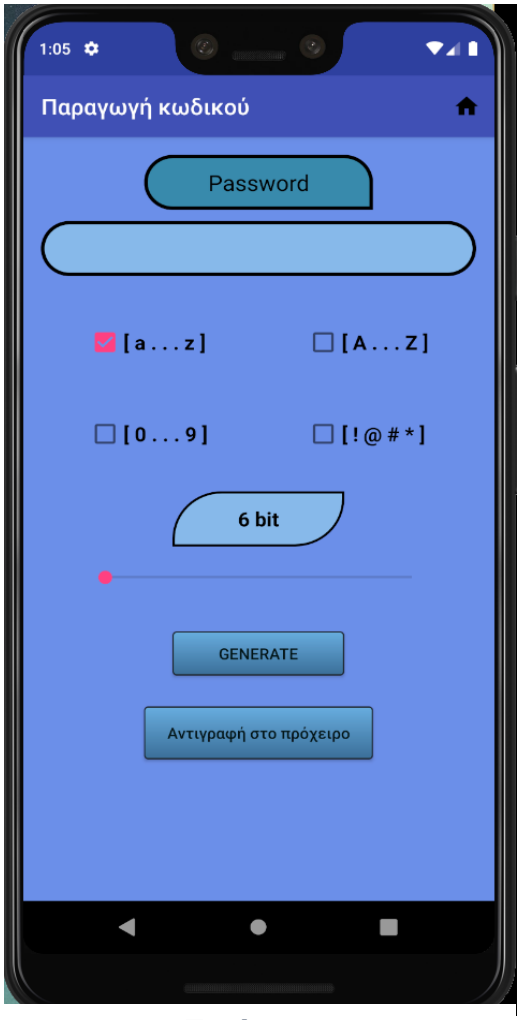
Εικόνα 4.2

Εικόνα 4.3

Εικόνα 4.4

5.5 Παραγωγή κωδικού

Η εφαρμογή υποστηρίζει την δημιουργία τυχαίου κωδικού σύμφωνα με pattern και μέγεθος λέξης που επιλέγει ο χρήστης (εικόνα 5.1). Έτσι μπορούμε να φτιάξουμε έναν τυχαίο κωδικό χρησιμοποιώντας μικρά γράμματα από το αγγλικό αλφάβητο μαζί με κεφαλαία, αριθμούς και σύμβολα ή να κάνουμε οποιοδήποτε συνδυασμό με τα παραπάνω. Επίσης, επιλέγουμε το μήκος του κωδικού που θα δημιουργηθεί ενδεικτικά από 6 έως 32 bit μήκος λέξης και στην συνέχεια πατώντας το κουμπί GENERATE δημιουργείται συνέχεια ένας τυχαίος κωδικός (εικόνα 5.2). Προς διευκόλυνση του χρήστη υπάρχει το κουμπί Αντιγραφή στο πρόχειρο όπου ο παραγόμενος κωδικός αποθηκεύεται προσωρινά στο πρόχειρο της συσκευής ώστε να χρησιμοποιηθεί σε κάποια άλλη λειτουργία αργότερα.



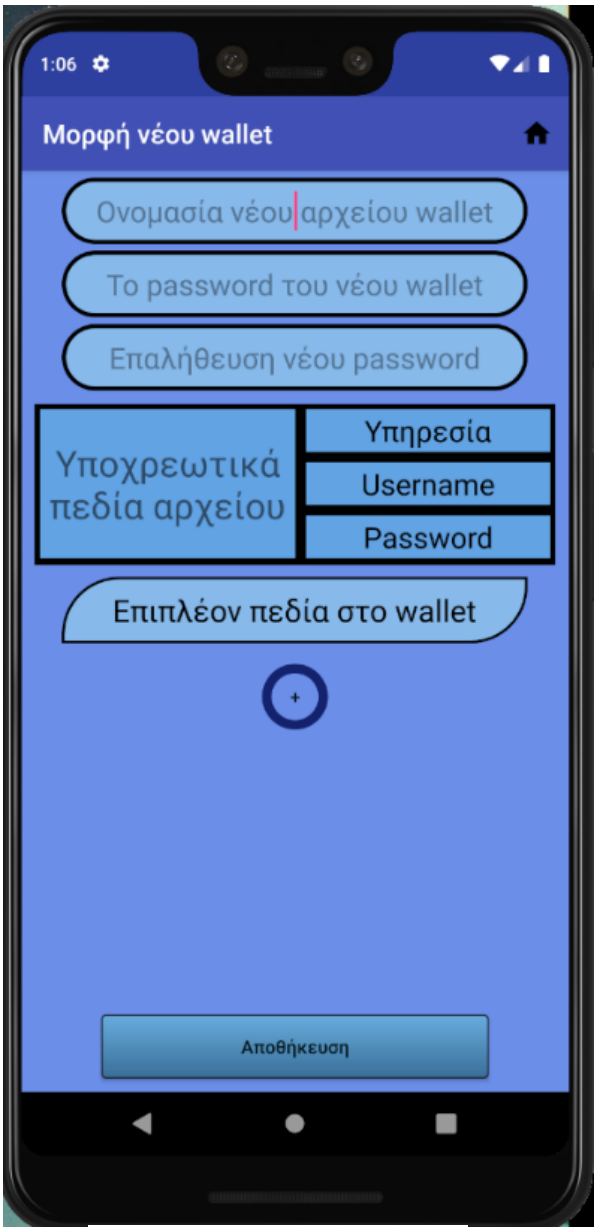
Εικόνα 5.1

Εικόνα 5.2

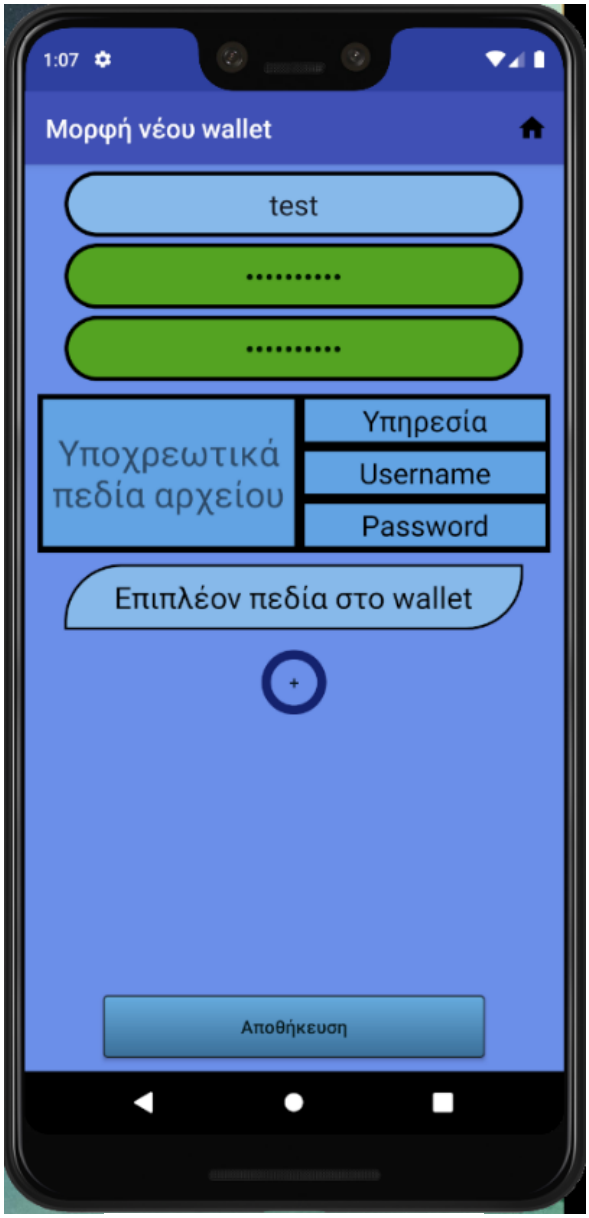
5.6 Δημιουργία νέου wallet

Στην δραστηριότητα δημιουργίας νέου wallet κωδικών (εικόνα 6.1) ο χρήστης επιλέγει ο ίδιος την μορφή που θα έχει το wallet. Αυτό σημαίνει πως θα πρέπει αρχικά να δώσει την ονομασία του νέου wallet και στην συνέχεια να εισάγει το password (εικόνα 6.2) με το οποίο θα κρυπτογραφηθεί. Έπειτα, το wallet πάντα θα έχει κάποια υποχρεωτικά πεδία τα οποία είναι «Υπηρεσία», για παράδειγμα Facebook,Instagram,Gmail, το «Username» και το «Password». Στην συνέχεια, ο χρήστης μπορεί να επιλέξει το wallet να έχει επιπλέον πεδία (εικόνα 6.3) πέραν των υποχρεωτικών τα οποία και μπορεί να τα ονομάσει όπως επιθυμεί (εικόνα 6.4). Αφού

είναι έτοιμο το wallet και αποθηκευμένο τότε η εφαρμογή μας παροτρύνει να εισάγουμε το πρώτο credential (εικόνα 6.5,6.6) δηλαδή την πρώτη μας εγγραφή κωδικών στο wallet σύμφωνα πάντα με την μορφή που αυτό έχει αποθηκευτεί (επιπλέον πεδία κλπ).



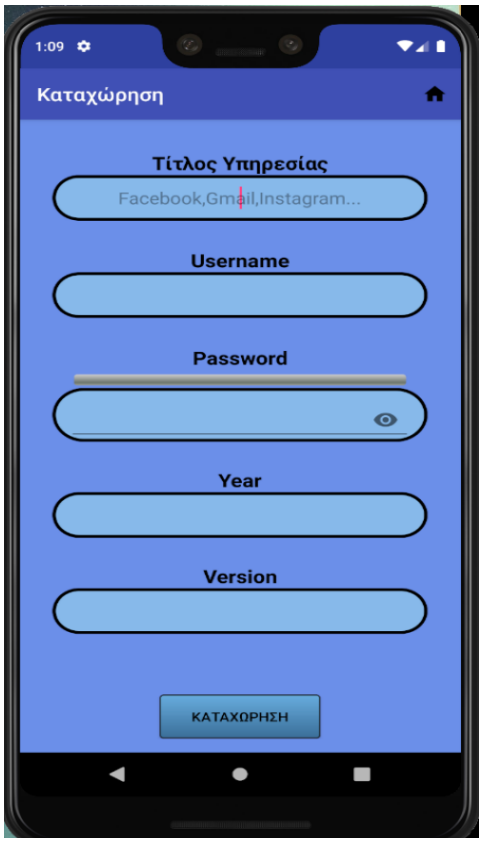
Εικόνα 6.1



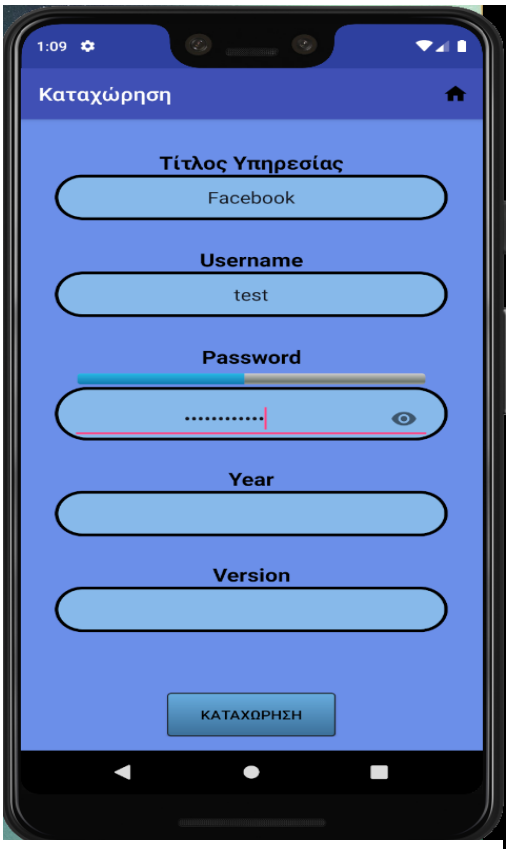
Εικόνα 6.2

Εικόνα 6.3

Εικόνα 6.4



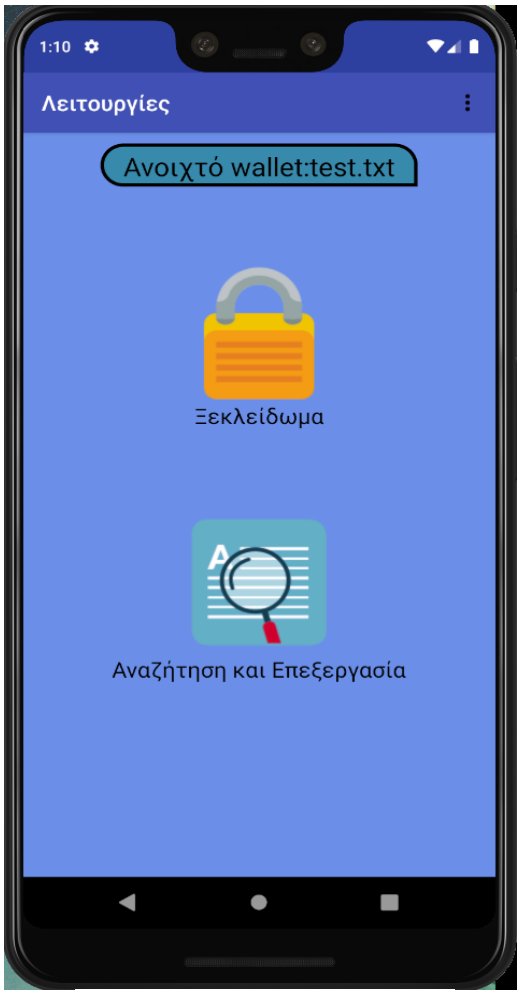
Εικόνα 6.5



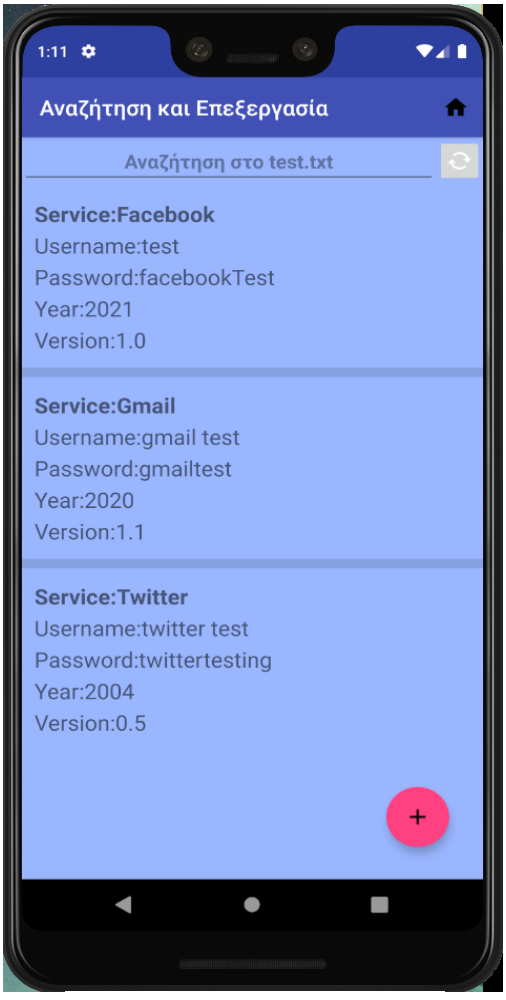
Εικόνα 6.6

5.7Αναζήτηση και επεξεργασία

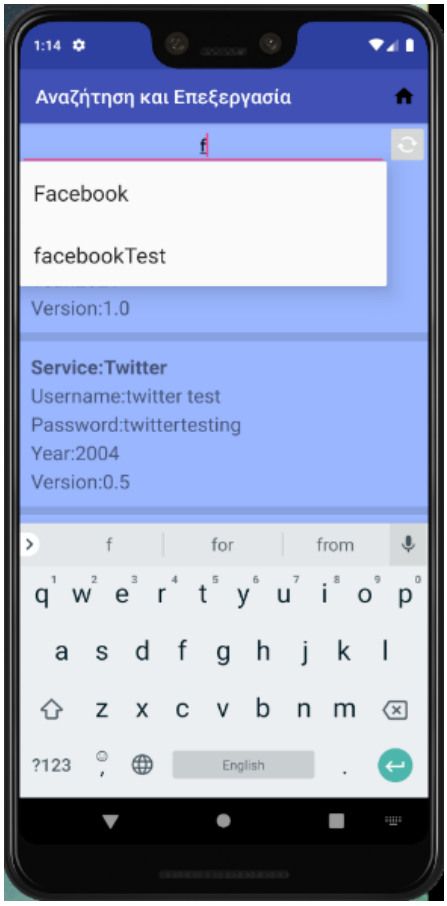
Μόλις ο χρήστης πατήσει το κουμπί «Αναζήτηση και Επεξεργασία» (εικόνα 7.1) μεταφέρεται στην δραστηριότητα όπου μπορεί να δει και να επεξεργαστεί όλες τις εγγραφές που έχει αποθηκεύσει στο συγκεκριμένο wallet (εικόνα 7.2). Στο πάνω μέρος της δραστηριότητας μπορεί να κάνει αναζήτηση στους κωδικούς του γράφοντας ολόκληρο ή κάποιο τμήμα του κωδικού που θέλει να αναζητήσει (εικόνα 7.3). Με παρατεταμένο πάτημα πάνω σε μία εγγραφή ο χρήστης μπορεί να κάνει «Επεξεργασία» , «Αντιγραφή και επικόλληση» και «Διαγραφή» της συγκεκριμένης εγγραφής (εικόνα 7.4). Στις πρώτες δύο ενέργειες εμφανίζεται ένα παράθυρο διαλόγου το οποίο περιέχει τις υπάρχουσες τιμές της εγγραφής σε editable πεδία ώστε να κάνει αλλαγές ο χρήστης (εικόνα 7.5).



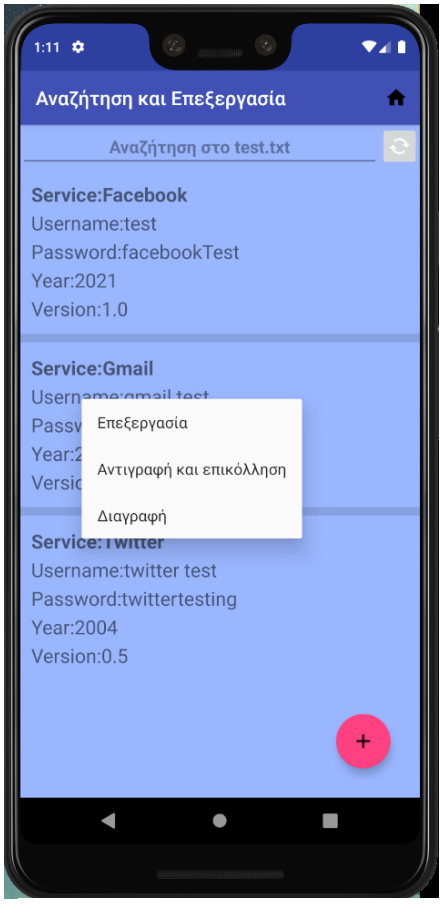
Εικόνα 7.1



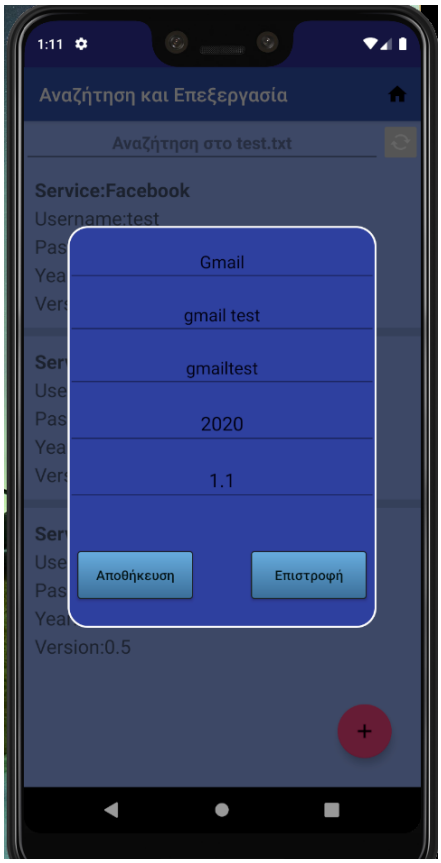
Εικόνα 7.2



Εικόνα 7.3



Εικόνα 7.4

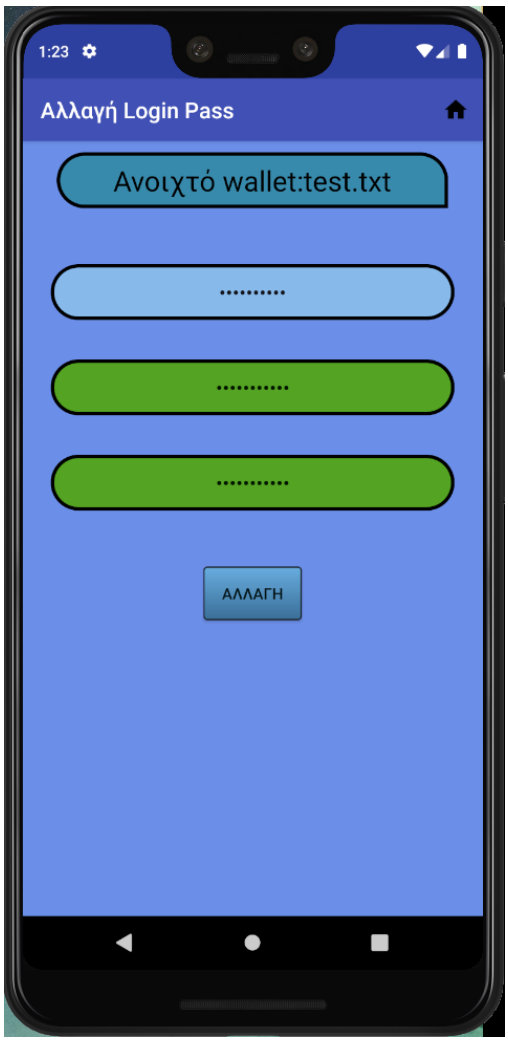


Εικόνα 7.5

5.8 Αλλαγή password στο wallet

Αυτή η δραστηριότητα είναι παρόμοια με την δραστηριότητα «Αλλαγή LoginPass» (εικόνα 8.1). Ο χρήστης πρέπει να εισάγει το ήδη υπάρχον password ενός ανοιχτού wallet και στην συνέχεια το νέο password μαζί με την επαλήθευσή του (εικόνα 8.2).

Εικόνα 8.1



Εικόνα 8.2

5.9 Μετονομασία wallet κωδικών

Ο χρήστης αφού επιλέξει από το υπο-μενού την «Μετονομασία» εμφανίζεται ένα dialog (εικόνα 9.1) όπου επιλέγει το wallet που επιθυμεί να μετονομάσει και μετά καταχωρεί τη νέα ονομασία και το password που έχει το συγκεκριμένο wallet προκειμένου να πραγματοποιηθεί η ενέργεια.

Εικόνα 9.1

5.10 Διαγραφή wallet κωδικών

Παρόμοια με την «Μετονομασία wallet κωδικών» η «Διαγραφή wallet κωδικών» εμφανίζει ένα dialog όπου ο χρήστης επιλέγει το wallet που επιθυμεί να διαγράψει και αφού καταχωρήσει το password του μπορεί να το διαγράψει (εικόνα 10.1).


9:04
Λειτουργίες
Δεν υπάρχει wallet ανοιχτό
test.txt
test3.txt
Εισαγωγή password
Διαγραφή
Επιστροφή
Αναζήτηση και Επεξεργασία

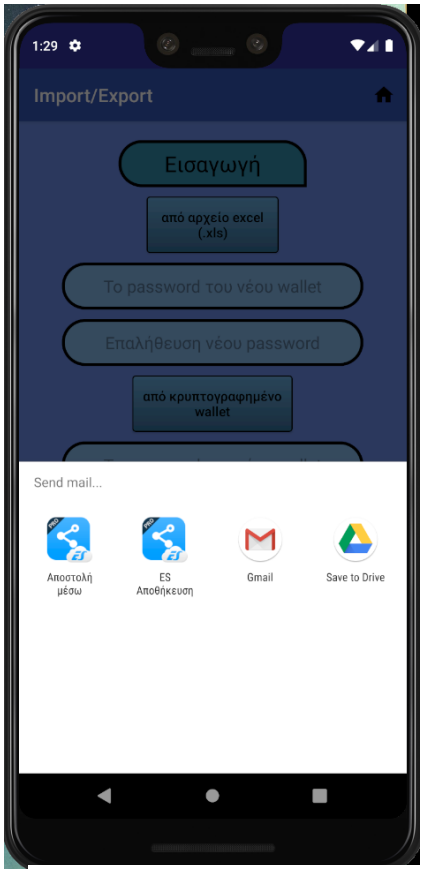
Εικόνα 10.1

5.11 Import/Export

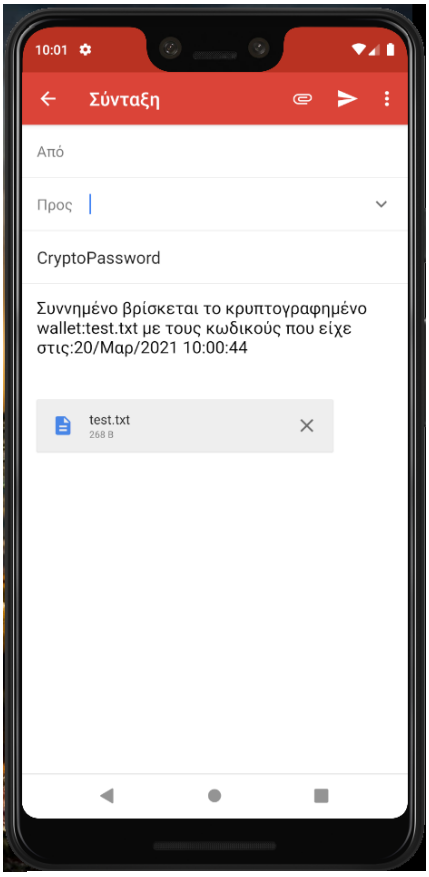
Η εφαρμογή υποστηρίζει την εισαγωγή και εξαγωγή κωδικών. Όταν, λοιπόν, ο χρήστης επιλέξει την συγκεκριμένη δραστηριότητα αν δεν έχει ξεκλειδώσει κάποιο wallet τότε είναι ενεργοποιημένα μόνο τα κουμπιά της εισαγωγής (εικόνα 11.1) ενώ σε αντίθετη περίπτωση είναι ενεργοποιημένα και τα κουμπιά της εξαγωγής (εικόνα 11.2). Δίνεται η δυνατότητα εισαγωγής από κρυπτογραφημένο wallet κωδικών με την προϋπόθεση να γνωρίζει ο χρήστης το κλειδί με το οποίο έχει κρυπτογραφηθεί το συγκεκριμένο wallet, έτσι ώστε να μπορέσει να το αποκρυπτογραφήσει και εισαγωγή κωδικών από αρχείο excel (.xls) όπου κατά την εισαγωγή τους θα δημιουργηθεί και ένα νέο wallet μέσα στην εφαρμογή. Παρόμοια, στην εξαγωγή μπορούμε να στείλουμε το wallet που έχουμε εκείνη την στιγμή ξεκλειδώσει είτε με την μορφή κρυπτογραφημένου wallet, είτε σαν excel αρχείο. Χρησιμοποιώντας κάποια επιλογή από την εξαγωγή τότε η εφαρμογή προτρέπει τον χρήστη να επιλέξει πως θέλει να στείλει το αρχείο (εικόνα 11.3). Αν ο χρήστης επιλέξει αποστολή μέσω mail τότε η εφαρμογή γράφει αυτόματα κάποια στοιχεία μέσα στο mail (εικόνα 11.4).

Εικόνα 11.1

Εικόνα 11.2



Εικόνα 11.3



Εικόνα 11.4

5.12 Προγραμματισμός των κλάσεων (Java)

Κάθε κουμπί και λειτουργία που εκτελούν οι παραπάνω δραστηριότητες στην πραγματικότητα εκτελούνται μέσω κώδικα σε κλάσεις Java. Παρακάτω λοιπόν θα παρουσιαστούν και σχολιαστούν οι κλάσεις και οι λειτουργίες τους.

Κλάση LoginPage

```
package com.cryptopassword.central;


import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.Toast;

import com.cryptopassword.helpers.DDException;
import com.cryptopassword.helpers.GlobalVariables;
import com.cryptopasswordentral.R;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.concurrent.Executor;
import java.util.concurrent.Executors;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.biometric.BiometricPrompt;
import androidx.core.app.ActivityCompat;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.helpers.FileHandler.getAbsoluteFile;
import static com.cryptopassword.helpers.FileHandler.readPinFile;
import static com.cryptopassword.helpers.FileHandler.writePinFile;
import static java.lang.Boolean.FALSE;
import static java.lang.Boolean.TRUE;


public class LoginPageextends AppCompatActivity {


private String password = "";
    private ImageViewimgDigit1;
```

```
    private ImageViewimgDigit2;
    private ImageViewimgDigit3;
    private ImageViewimgDigit4;
    private Boolean firstLogIn= TRUE;
    private String fingerprintSetted;
    private ImageButtonfingerprint_login;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.log_in);
ActivityCompat.requestPermissions(LoginPage.this,newString[]{Manifest.permission.W
RITE_EXTERNAL_STORAGE},1);
imgDigit1 = findViewById(R.id.digit1);
imgDigit2 = findViewById(R.id.digit2);
imgDigit3 = findViewById(R.id.digit3);
imgDigit4 = findViewById(R.id.digit4);
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(LoginPage.this);
fingerprintSetted=sharedPrefget.getString("FingerPrint","Empty");

Executor newExecutor = Executors.newSingleThreadExecutor();
        final BiometricPrompt.PromptInfopromptInfo = new
BiometricPrompt.PromptInfo.Builder()
                .setTitle(getResources().getString(R.string.fingerprintTitle))

.setSubtitle(getResources().getString(R.string.fingerprintSubTitle))
                .setNegativeButtonText("Cancel")
                .build();

        final BiometricPromptmyBiometricPrompt = new
BiometricPrompt(LoginPage.this, newExecutor, new
BiometricPrompt.AuthenticationCallback() {
@Override
public void onAuthenticationSucceeded(@NonNull
BiometricPrompt.AuthenticationResult result) {
super.onAuthenticationSucceeded(result);
SharedPreferencessharedPref = LoginPage.this.getPreferences(Context.MODE_PRIVATE);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("FingerPrint", "Setted");
editor.apply();
fingerprintResult("Ok");
}

@Override
public void onAuthenticationFailed() {
super.onAuthenticationFailed();
fingerprintResult("Error");
}

@Override
public void onAuthenticationError(int errorCode, @NonNull CharSequenceerrString) {
super.onAuthenticationError(errorCode, errString);
fingerprintResult(errString.toString());
}

        });
```

```
fingerprint_login= findViewById(R.id.fingerprint_login);
        if(fingerprintSetted.equals("Empty")){
fingerprint_login.setVisibility(View.INVISIBLE);
}else{
fingerprint_login.setVisibility(View.VISIBLE);
}
fingerprint_login.setOnClickListener(view -> {
myBiometricPrompt.authenticate(promptInfo);
});
}

public void numberPressed(View view) throws IOException, DDException {
        String pressed =(String) ((Button) view).getText();
        if(password.length()<5) {
if (pressed.equals(getResources().getString(R.string.deleteSymbol))
&&password.length() >0) {
password = password.substring(0, password.length() - 1);
} else if (!pressed.equals(getResources().getString(R.string.deleteSymbol))) {
password = password + pressed;
}
if (password.length() == 0)
{imgDigit1.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit2.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit3.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit4.setImageResource(R.drawable.round_radio_button_unchecked);
}
if (password.length() == 1)
{imgDigit1.setImageResource(R.drawable.round_radio_button_checked);
imgDigit2.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit3.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit4.setImageResource(R.drawable.round_radio_button_unchecked);
}
if (password.length() == 2)
{imgDigit1.setImageResource(R.drawable.round_radio_button_checked);
imgDigit2.setImageResource(R.drawable.round_radio_button_checked);
imgDigit3.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit4.setImageResource(R.drawable.round_radio_button_unchecked);
}
if (password.length() == 3)
{imgDigit1.setImageResource(R.drawable.round_radio_button_checked);
imgDigit2.setImageResource(R.drawable.round_radio_button_checked);
imgDigit3.setImageResource(R.drawable.round_radio_button_checked);
imgDigit4.setImageResource(R.drawable.round_radio_button_unchecked);
}
if (password.length() == 4)
{imgDigit1.setImageResource(R.drawable.round_radio_button_checked);
imgDigit2.setImageResource(R.drawable.round_radio_button_checked);
imgDigit3.setImageResource(R.drawable.round_radio_button_checked);
imgDigit4.setImageResource(R.drawable.round_radio_button_checked);
LogIn();
}
        }
```

```
    }


public void checkFirstLogIn() {
        File applicationPath = getAbsoluteFile("", this);
File path = new File(applicationPath.getAbsolutePath() + "/CryptoPassword/");
        if (!path.isDirectory()) {
Toast.makeText(LoginPage.this, R.string.createPIN, Toast.LENGTH_LONG).show();
firstLogIn= TRUE;
}else{
firstLogIn= FALSE;
}
        ((GlobalVariables) this.getApplication()).setFirstLogIn(firstLogIn);
}

public void LogIn() throws IOException, DDException {
if(firstLogIn){
Toast.makeText(LoginPage.this,
getResources().getString(R.string.PINcreated),Toast.LENGTH_SHORT).show();
writePinFile(password,this);
resetLogIn();
}else{
ArrayList<String>decryptedPin = readPinFile(this);
            if (decryptedPin != null &&password.equals(decryptedPin.get(0))){
Toast.makeText(LoginPage.this, R.string.rightPIN,Toast.LENGTH_SHORT).show();
((GlobalVariables) this.getApplication()).setPassword(password);
}else{
Toast.makeText(LoginPage.this, R.string.falsePIN,Toast.LENGTH_SHORT).show();
resetLogIn();
                return;
}
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(LoginPage.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(LoginPage.this, MainMenu.class);
startActivity(intent);
}
    }



public void resetLogIn(){
imgDigit1.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit2.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit3.setImageResource(R.drawable.round_radio_button_unchecked);
imgDigit4.setImageResource(R.drawable.round_radio_button_unchecked);
password = "";
firstLogIn= FALSE;
}



@Override
public void onRequestPermissionsResult(int requestCode, String permissions[],
@NonNull int[] grantResults) {
```

```
if (requestCode == 1) {
if (grantResults.length>0  &&grantResults[0] == PackageManager.PERMISSION_GRANTED)
{
checkFirstLogIn();
} else {
Toast.makeText(LoginPage.this, "Permission denied to read/write your External
storage", Toast.LENGTH_SHORT).show();
exitApplication(LoginPage.this);
}
        }
    }

private void fingerprintResult(String result){
        Thread thread = new Thread(){
public void run(){
runOnUiThread(() -> {
if(result.equals("Ok")){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(LoginPage.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(LoginPage.this, MainMenu.class);
startActivity(intent);
}else if(!result.equals("Error")){
Toast.makeText(LoginPage.this, result, Toast.LENGTH_SHORT).show();
}
                });
}
        };
thread.start();
}
    }
```

Κλάση MainMenu

```
package com.cryptopassword.central;

import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.cryptopassword.helpers.DeleteWalletDialog;
import com.cryptopassword.helpers.GlobalVariables;
import com.cryptopassword.helpers.RenameWalletDialog;
import com.cryptopassword.helpers.SelectWalletDialog;
import com.cryptopasswordentral.R;

import java.util.ArrayList;
import java.util.Objects;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.central.ExitActivity.logOut;
import static com.cryptopassword.helpers.FileHandler.getAllFileNames;

public class MainMenuextends AppCompatActivity {

    Boolean firstLogIn;
TextViewwalletName;
Button changeWallet,showCredentials;

@Override
protected void onCreate(Bundle savedInstanceState){
super.onCreate(savedInstanceState);
setContentView(R.layout.main_menu);
firstLogIn= ((GlobalVariables) this.getApplication()).getFirstLogIn();
walletName= findViewById(R.id.walletName_txt);

showCredentials= findViewById(R.id.showCredentials);
showCredentials.setOnClickListener(view -> {
if(((GlobalVariables) this.getApplication()).getFileName() == null){
```

```
Toast.makeText(this, getResources().getString(R.string.walletNotOpened),
Toast.LENGTH_SHORT).show();
                return;
}
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "ShowCredentials");
editor.apply();
Intent intent = new Intent(MainMenu.this, ShowCredentials.class);
startActivity(intent);
});

changeWallet= findViewById(R.id.changeWallet);
changeWallet.setOnClickListener(view -> {
if(!walletsExists()){
Toast.makeText(this, getResources().getString(R.string.noWalletExists),
Toast.LENGTH_SHORT).show();
                return;
}
SelectWalletDialogselectWalletDialog=new
SelectWalletDialog(MainMenu.this,((GlobalVariables)
getApplication()).getFileName());
Objects.requireNonNull(selectWalletDialog.getWindow()).setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));
selectWalletDialog.show();
selectWalletDialog.setOnDismissListener(dialogInterface -> {
setButtonsVisibility();
});
});
setButtonsVisibility();
}

private void setButtonsVisibility(){
            Animation animation;
animation = AnimationUtils.loadAnimation(getApplicationContext(),R.anim.fade_in);
walletName.startAnimation(animation);
            if(firstLogIn) {
walletName.setText(getResources().getString(R.string.walletNotOpened));
}else{
if(((GlobalVariables) this.getApplication()).getFileName() != null){
walletName.setText(getResources().getString(R.string.nameWalletOpened)+":"+((Globa
lVariables) this.getApplication()).getFileName());
}else{
walletName.setText(getResources().getString(R.string.walletNotOpened));
}

            }

        }




private booleanwalletsExists(){
ArrayList<String>filesInFilder = getAllFileNames(MainMenu.this);
            int wallets=0;
```

```
            for(String s:filesInFilder) {
if(s.contains(".txt") && !s.equals("pin.txt")){
                    wallets++;
}
            }
if (wallets>0)
return true;
            else
                return false;
}

@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener()
                {
@Override
public void onClick(DialogInterface dialog, int which) {
exitApplication(MainMenu.this);
}

                })
                .setNegativeButton("No", null)
                .show();
}

@Override
public booleanonCreateOptionsMenu(Menu menu){
this.getMenuInflater().inflate(R.menu.menu, menu);
            return true;
}

@Override
public booleanonOptionsItemSelected(MenuItem item) {
        String id = item.getTitle().toString();
        if(id.equals(getResources().getString(R.string.logInChange))){
            ((GlobalVariables) this.getApplication()).setChangePassForLogIn(true);
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "PasswordChange");
editor.apply();
Intent intent = new Intent(MainMenu.this, PasswordChange.class);
startActivity(intent);
            return super.onOptionsItemSelected(item);
}else if(id.equals(getResources().getString(R.string.setFingerprintLogIn))){
final PackageManager pm = this.getPackageManager();
            if(pm.hasSystemFeature(PackageManager.FEATURE_FINGERPRINT)){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "FastLoginMenu");
editor.apply();
Intent intent = new Intent(MainMenu.this, FastLoginMenu.class);
startActivity(intent);
```

```
}else{
Toast.makeText(MainMenu.this,
MainMenu.this.getResources().getString(R.string.noFingerprintHW),
Toast.LENGTH_LONG).show();
}
        }else if(id.equals(getResources().getString(R.string.newWallet))){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "CredentialFormat");
editor.apply();
Intent intent = new Intent(MainMenu.this, CredentialFormat.class);
startActivity(intent);
}else if(id.equals(getResources().getString(R.string.importExport))){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "ImpExpMenu");
editor.apply();
Intent intent = new Intent(MainMenu.this, ImpExpMenu.class);
startActivity(intent);
}else if(id.equals(getResources().getString(R.string.generatePass))) {
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "Generator");
editor.apply();
Intent intent = new Intent(MainMenu.this, Generator.class);
startActivity(intent);
}else if(id.equals(getResources().getString(R.string.changeWalletPassword))){
if(!walletsExists()) {
Toast.makeText(MainMenu.this,
MainMenu.this.getResources().getString(R.string.walletNotOpenedDenyMenu),
Toast.LENGTH_LONG).show();
                return super.onOptionsItemSelected(item);
}if(((GlobalVariables) this.getApplication()).getFileName()==null){
Toast.makeText(MainMenu.this,
MainMenu.this.getResources().getString(R.string.walletNotOpenedDenyMenu2),
Toast.LENGTH_LONG).show();
                return super.onOptionsItemSelected(item);
}
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "PasswordChange");
editor.apply();
((GlobalVariables) this.getApplication()).setChangePassForLogIn(false);
Intent intent = new Intent(MainMenu.this, PasswordChange.class);
startActivity(intent);
}else if(id.equals(getResources().getString(R.string.renameWallet))) {
if(!walletsExists()) {
Toast.makeText(MainMenu.this,
MainMenu.this.getResources().getString(R.string.walletNotOpenedDenyMenu),
Toast.LENGTH_LONG).show();
                return super.onOptionsItemSelected(item);
}
RenameWalletDialogrenameWalletDialog=new
RenameWalletDialog(MainMenu.this,((GlobalVariables)
```

```
getApplication()).getFileName());
Objects.requireNonNull(renameWalletDialog.getWindow()).setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));
renameWalletDialog.show();
renameWalletDialog.setOnDismissListener(dialogInterface -> {
setButtonsVisibility();
});
}else if(id.equals(getResources().getString(R.string.deleteWallet))) {
if(!walletsExists()) {
Toast.makeText(MainMenu.this,
MainMenu.this.getResources().getString(R.string.walletNotOpenedDenyMenu),
Toast.LENGTH_LONG).show();
                return super.onOptionsItemSelected(item);
}
DeleteWalletDialogdeleteWalletDialog=new
DeleteWalletDialog(MainMenu.this,((GlobalVariables)
getApplication()).getFileName());
Objects.requireNonNull(deleteWalletDialog.getWindow()).setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));
deleteWalletDialog.show();
}else if(id.equals(getResources().getString(R.string.disconnect))) {
new AlertDialog.Builder(this)
                    .setIcon(R.drawable.log_off)
                    .setTitle(R.string.prosoxi)
                    .setMessage(R.string.exodosApoEfarmogi)
                    .setPositiveButton("Yes", new
DialogInterface.OnClickListener()
                    {
@Override
public void onClick(DialogInterface dialog, int which) {
exitApplication(MainMenu.this);
}

                    })
                    .setNegativeButton("No", null)
                    .show();
}
return super.onOptionsItemSelected(item);
}

@Override
protected void onStart(){
super.onStart();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
        if(sharedPrefget.getString("Navigation","Logout").equals("MainMenu")){
//irthe apo allo activity kai oxi apo lost focus stinefarmogi
}else{
Toast.makeText(MainMenu.this,
MainMenu.this.getResources().getString(R.string.disconnected),
Toast.LENGTH_LONG).show();
LogOut(MainMenu.this);
Intent intent = new Intent(MainMenu.this,LoginPage.class);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK| Intent.FLAG_ACTIVITY_CLEAR_TASK);
startActivity(intent);
}
    }
```

```
@Override
protected void onStop(){
super.onStop();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
        if(sharedPrefget.getString("Navigation","Logout").equals("MainMenu")){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(MainMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "Logout");
editor.apply();
}else{
//stamatise kai pige se allo activity kai oxi se lost focus stinefarmogi
}
    }
}
```

Κλάση CredentialFormat

```
package com.cryptopassword.central;

import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.Gravity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RelativeLayout;
import android.widget.Toast;

import com.cryptopassword.helpers.GlobalVariables;
import com.cryptopasswordentral.R;

import java.util.ArrayList;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.helpers.FileHandler.getAllFileNames;


public class CredentialFormatextends AppCompatActivity {

private int numberNewFields= 0;
    private Button addProperty,minusProperty;
    private ArrayList<String>newFields;
    private EditTextfileName,newPassword,newPasswordVerify;

@Override
```

```
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.credential_format);

addProperty= findViewById(R.id.addProperty);
addProperty.setOnClickListener(view ->addLine());

minusProperty= findViewById(R.id.minusProperty);
minusProperty.setVisibility(View.GONE);
minusProperty.setOnClickListener(view ->deleteLine());

newPassword= findViewById(R.id.newPassword);
newPassword.addTextChangedListener(new TextWatcher() {
public void afterTextChanged(Editable s) {showPasswordsVerifyColor();}
public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
public void onTextChanged(CharSequence s, int start, int before, int count) {}
        });

newPasswordVerify=  findViewById(R.id.newPasswordVerify);
newPasswordVerify.addTextChangedListener(new TextWatcher() {
public void afterTextChanged(Editable s) {showPasswordsVerifyColor();}
public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
public void onTextChanged(CharSequence s, int start, int before, int count) {}
        });

fileName= findViewById(R.id.fileName);
Button save = findViewById(R.id.saveFormat);
save.setOnClickListener(view -> {
            String fineNameWithExtension = fileName.getText() + ".txt";
            if(fileName.getText().toString().equals("")){
Toast.makeText(CredentialFormat.this,
R.string.fileNameEmpty,Toast.LENGTH_LONG).show();
                return;
}if(!fileName.getText().toString().matches("[0-9a-zA-z]+$")){
Toast.makeText(CredentialFormat.this,
R.string.fileNameRightName,Toast.LENGTH_LONG).show();
                return;
}
if(getAllFileNames(getApplicationContext()).contains(fineNameWithExtension)){
Toast.makeText(CredentialFormat.this,
R.string.fileNameExists,Toast.LENGTH_LONG).show();
                return;
}
if(newPassword.getText().toString().equals("")){
Toast.makeText(CredentialFormat.this,
R.string.newPasswordEmpty,Toast.LENGTH_LONG).show();
                return;
}
if(newPassword.getText().toString().length()<10){
Toast.makeText(CredentialFormat.this,
R.string.newPasswordSmall,Toast.LENGTH_LONG).show();
                return;
}if(!newPassword.getText().toString().equals(newPasswordVerify.getText().toString(
))){
Toast.makeText(CredentialFormat.this,
R.string.notMatchingPasswords,Toast.LENGTH_LONG).show();
                return;
}
```

```
if(newPassword.getText()!=null){
                ((GlobalVariables)
getApplication()).setPassword(newPassword.getText().toString());
}
EditText et;
newFields= new ArrayList<>();
            for (int i = 0; i<numberNewFields; i++) {
                et = findViewById(i + 1);
                if (et.getText() != null)
newFields.add(et.getText().toString());
}
            ((GlobalVariables)
getApplication()).setFileName((fineNameWithExtension));
((GlobalVariables) getApplication()).setNewFields(newFields);
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(CredentialFormat.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "InsertCredentials");
editor.apply();
Intent intent = new Intent(CredentialFormat.this, InsertCredentials.class);
startActivity(intent);
});
}

private void showPasswordsVerifyColor(){
if (newPassword.getText().toString().length() >0
&&newPasswordVerify.getText().toString().length() >0 &&
newPassword.getText().toString().equals(newPasswordVerify.getText().toString())) {
newPasswordVerify.setBackgroundResource(R.drawable.edit_txt_style_green);
newPassword.setBackgroundResource(R.drawable.edit_txt_style_green);
} else if(newPassword.getText().toString().length() >0
&&newPasswordVerify.getText().toString().length() >0 &&

!newPassword.getText().toString().equals(newPasswordVerify.getText().toString())){
newPasswordVerify.setBackgroundResource(R.drawable.edit_txt_style_red);
newPassword.setBackgroundResource(R.drawable.edit_txt_style_red);
}
    }

private void addLine() {
RelativeLayoutll = findViewById(R.id.cred_format_scrl);
EditText et = new EditText(this);
        float pixelsWidth =  350 *
CredentialFormat.this.getResources().getDisplayMetrics().density;
        float pixelsHeight =  50 *
CredentialFormat.this.getResources().getDisplayMetrics().density;
RelativeLayout.LayoutParams p = new
RelativeLayout.LayoutParams(Math.round(pixelsWidth), Math.round(pixelsHeight));
        if(numberNewFields== 0){
//p.addRule(RelativeLayout.BELOW, R.id.addPropertyLO);
minusProperty.setVisibility(View.VISIBLE);
}else{
p.addRule(RelativeLayout.BELOW, numberNewFields);
}
p.addRule(RelativeLayout.CENTER_HORIZONTAL);
p.setMargins(0, 30, 0, 0);
et.setHint(R.string.propertyName);
et.setGravity(Gravity.CENTER);
```

```
et.setLayoutParams(p);
et.setTextSize(25);
et.setBackgroundResource(R.drawable.edit_txt_style);
et.setId(numberNewFields+ 1);
ll.addView(et);
numberNewFields++;
        if(numberNewFields== 6) {
addProperty.setVisibility(View.GONE);
}
    }

public void deleteLine() {
if(numberNewFields>0){
RelativeLayoutll = findViewById(R.id.cred_format_scrl);
EditText et = findViewById(numberNewFields);
ll.removeView(et);
numberNewFields--;
            if(numberNewFields== 0) {
minusProperty.setVisibility(View.GONE);
}
if(numberNewFields<6) {
addProperty.setVisibility(View.VISIBLE);
}
        }
    }

@Override
public booleanonCreateOptionsMenu(Menu menu) {
this.getMenuInflater().inflate(R.menu.menuother, menu);
        return true;
}

@Override
public booleanonOptionsItemSelected(MenuItem item) {
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(CredentialFormat.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(CredentialFormat.this, MainMenu.class);
startActivity(intent);
        return super.onOptionsItemSelected(item);
}

@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener()
                {
@Override
public void onClick(DialogInterface dialog, int which) {
exitApplication(CredentialFormat.this);
}

                })
```

```
                .setNegativeButton("No", null)
                .show();
}

@Override
protected void onStart(){
super.onStart();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(this);

if(sharedPrefget.getString("Navigation","Logout").equals("CredentialFormat")){
//irthe apo allo activity kai oxi apo lost focus stinefarmogi
}else{
exitApplication(this);
}
    }

@Override
protected void onStop(){
super.onStop();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(this);

if(sharedPrefget.getString("Navigation","Logout").equals("CredentialFormat")){
exitApplication(this);
}else{
//stamatise kai pige se allo activity kai oxi se lost focus stin efarmogi
}
    }

}
```

Κλάση InsertCredentials

```
package com.cryptopassword.central;

import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.graphics.Typeface;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.Gravity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.cryptopassword.helpers.Credential;
import com.cryptopassword.helpers.DDException;
import com.cryptopassword.helpers.GlobalVariables;
```

```
import com.cryptopasswordentral.R;

import java.io.IOException;
import java.util.ArrayList;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.central.ExitActivity.logOut;
import static com.cryptopassword.helpers.FileHandler.getAllFileNames;
import static com.cryptopassword.helpers.FileHandler.writeDataFile;

public class InsertCredentialsextends AppCompatActivity {

private int mikos,synoloNewFieldsDatas;
    private EditTexttitlos,username,passwordText;
    private ProgressBarsimpleProgressBar;

@Override
protected void onCreate(Bundle savedInstanceState){
super.onCreate(savedInstanceState);
setContentView(R.layout.insert_credentials);
ArrayList<String>newFields;
newFields = ((GlobalVariables) this.getApplication()).getNewFields();
        final String password = ((GlobalVariables)
this.getApplication()).getPassword();
passwordText= findViewById(R.id.password);
titlos= findViewById(R.id.titlos);
username = findViewById(R.id.username);
simpleProgressBar= findViewById(R.id.simpleProgressBar);
simpleProgressBar.setProgressDrawable(getResources().getDrawable(R.drawable.progre
ssbar_color,null));

        if(newFields != null){
addLine(newFields);
}

passwordText.addTextChangedListener(new TextWatcher() {
public void afterTextChanged(Editable s) {
if (s.length()==0){
mikos=0;
simpleProgressBar.setProgress(passwordText.getText().length());
}else if(s.length()>mikos){
simpleProgressBar.setProgress(passwordText.getText().length()*4);
mikos=(s.length());
}else if(s.length()<mikos){
simpleProgressBar.setProgress(0);
simpleProgressBar.setProgress(passwordText.getText().length()*4);
mikos=(s.length());
}
            }
public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
public void onTextChanged(CharSequence s, int start, int before, int count) {}
        });

Button kataxwrisi = findViewById(R.id.kataxwrisi);
```

```
kataxwrisi.setOnClickListener(view -> {
ArrayList<String>listGiaCredential = readNewFields();
            if(rightFields(listGiaCredential)){
                Credential credential = new
Credential(titlos.getText().toString(),username.getText().toString(),
passwordText.getText().toString(),listGiaCredential);
                try {
writeDataFile(credential,password,((GlobalVariables)
getApplication()).getFileName(),InsertCredentials.this);
((GlobalVariables)getApplication()).setFirstLogIn(false);
Toast.makeText(InsertCredentials.this, R.string.insertSuccess,
Toast.LENGTH_SHORT).show();
clearFields();
} catch (IOException | DDException e) {
e.printStackTrace();
Toast.makeText(InsertCredentials.this, R.string.insertError,
Toast.LENGTH_SHORT).show();
}
            }
        });

}

private Boolean rightFields(ArrayList<String>listGiaCredential){
if(titlos.getText().toString().length() <1){
Toast.makeText(InsertCredentials.this, R.string.kenoSerivce,
Toast.LENGTH_SHORT).show();
            return false;
}
if(username.getText().toString().length() <1){
Toast.makeText(InsertCredentials.this, R.string.kenoUsername,
Toast.LENGTH_SHORT).show();
            return false;
}
if(passwordText.getText().toString().length() <1){
Toast.makeText(InsertCredentials.this, R.string.kenoPassword,
Toast.LENGTH_SHORT).show();
            return false;
}
        String proigoumenoField = null;
        for (String s : listGiaCredential){
if(s.isEmpty()){
Toast.makeText(InsertCredentials.this,
getResources().getString(R.string.kenoNewField)+ " " + proigoumenoField,
Toast.LENGTH_SHORT).show();
                return false;
}
proigoumenoField = s;
}
return true;
}

private void clearFields(){
passwordText.setText(null);
titlos.setText(null);
username.setText(null);
simpleProgressBar.setProgress(0);
EditText et;
```

```
        for(int i=0;i<synoloNewFieldsDatas;i=i+2){
            et = findViewById(i+2);
et.setText(null);
}
    }

private ArrayList<String>readNewFields(){
EditText et;
TextView tv;
ArrayList<String>telikiLista = new ArrayList<>();
        for(int i=0;i<synoloNewFieldsDatas;i=i+2){
            tv = findViewById(i+1);
et = findViewById(i+2);
telikiLista.add(tv.getText().toString());
telikiLista.add(et.getText().toString());
}
return telikiLista;
}

private void addLine(ArrayList<String> str){
RelativeLayoutll = findViewById(R.id.insert_cred_layout);
synoloNewFieldsDatas=0;
        for (String s:str){
RelativeLayout.LayoutParams p;
TextView tv = new TextView(this);
p = new RelativeLayout.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
            if(synoloNewFieldsDatas== 0){
p.addRule(RelativeLayout.BELOW, R.id.etPasswordLayout);
}else{
p.addRule(RelativeLayout.BELOW, synoloNewFieldsDatas);
}
p.setMargins(0, 100, 0, 0);
p.addRule(RelativeLayout.CENTER_HORIZONTAL);
tv.setTypeface(Typeface.DEFAULT_BOLD);
tv.setLayoutParams(p);
tv.setTextSize(20);
tv.setTextColor(getResources().getColor(R.color.dark,null));
tv.setText(s);
tv.setId(synoloNewFieldsDatas+1);
ll.addView(tv);

            float pixelsWidth =  350 *
InsertCredentials.this.getResources().getDisplayMetrics().density;
            float pixelsHeight =  50 *
InsertCredentials.this.getResources().getDisplayMetrics().density;
EditText et = new EditText(this);
p = new RelativeLayout.LayoutParams(Math.round(pixelsWidth),
Math.round(pixelsHeight));
p.addRule(RelativeLayout.BELOW, synoloNewFieldsDatas+1);
p.addRule(RelativeLayout.CENTER_HORIZONTAL);
et.setGravity(Gravity.CENTER);
et.setBackgroundResource(R.drawable.edit_txt_style);
et.setLayoutParams(p);
et.setId(synoloNewFieldsDatas+2);
ll.addView(et);
synoloNewFieldsDatas=synoloNewFieldsDatas+2;
}
```

```
    }

@Override
public booleanonCreateOptionsMenu(Menu menu) {
this.getMenuInflater().inflate(R.menu.menuother, menu);
        return true;
}

@Override
public booleanonOptionsItemSelected(MenuItem item) {
ArrayList<String>filesInFilder = getAllFileNames(InsertCredentials.this);
        if(!filesInFilder.contains(((GlobalVariables)
getApplication()).getFileName())){
            ((GlobalVariables) getApplication()).setFileName((null));
}
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(InsertCredentials.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(InsertCredentials.this, MainMenu.class);
startActivity(intent);
        return super.onOptionsItemSelected(item);
}

@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener()
                {
@Override
public void onClick(DialogInterface dialog, int which) {
exitApplication(InsertCredentials.this);
}

                })
                .setNegativeButton("No", null)
                .show();
}

@Override
protected void onStart(){
super.onStart();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(InsertCredentials.this);

if(sharedPrefget.getString("Navigation","Logout").equals("InsertCredentials")){
//irthe apo allo activity kai oxi apo lost focus stinefarmogi
}else{
Toast.makeText(InsertCredentials.this,
InsertCredentials.this.getResources().getString(R.string.disconnected),
Toast.LENGTH_LONG).show();
logOut(InsertCredentials.this);
Intent intent = new Intent(InsertCredentials.this,LoginPage.class);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK| Intent.FLAG_ACTIVITY_CLEAR_TASK);
```

```
startActivity(intent);
}
    }

@Override
protected void onStop(){
super.onStop();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(InsertCredentials.this);

if(sharedPrefget.getString("Navigation","Logout").equals("InsertCredentials")){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(InsertCredentials.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "Logout");
editor.apply();
}else{
//stamatise kai pige se allo activity kai oxi se lost focus stin efarmogi
}
    }

}
```

Κλάση ShowCredentials

```
package com.cryptopassword.central;

import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ImageButton;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.Toast;

import com.cryptopassword.helpers.Credential;
import com.cryptopassword.helpers.CredentialsAdapter;
import com.cryptopassword.helpers.DDException;
import com.cryptopassword.helpers.EditCredentialDialog;
import com.cryptopassword.helpers.GlobalVariables;
import com.cryptopasswordentral.R;
import com.google.android.material.floatingactionbutton.FloatingActionButton;

importjava.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Objects;
```

```
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.AppCompatAutoCompleteTextView;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.central.ExitActivity.logOut;
import static com.cryptopassword.helpers.FileHandler.readDataFile;
import static com.cryptopassword.helpers.FileHandler.writeAgainAllCredentials;
import static com.cryptopassword.helpers.FileHandler.writeDataFile;

public class ShowCredentialsextends AppCompatActivityimplements
EditCredentialDialog.DialogListener {

private ListViewlistView;
    private CredentialsAdaptercrAdapter;
    private AppCompatAutoCompleteTextViewautoTextView;
    private ArrayList<Credential>crList= new ArrayList<>();
    private ImageButtonrefreshCrList;
    private ArrayList<String>decryptedString, listNewFieldsDatas;
    private AdapterView.AdapterContextMenuInfoacmi;
    private FloatingActionButtoninsertCredentials;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
        final String password = ((GlobalVariables)
this.getApplication()).getPassword();
setContentView(R.layout.show_credentials);
RelativeLayoutrl = findViewById(R.id.show_credentials);

RelativeLayout.LayoutParams p;
p = new RelativeLayout.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
p.addRule(RelativeLayout.BELOW, R.id.searchLayout);

listView= new ListView(this);
listView.setLayoutParams(p);
listView.setDividerHeight(30);
rl.addView(listView);

decryptedString= readDataFile(password, ((GlobalVariables)
this.getApplication()).getFileName(),this);
        if(decryptedString!= null){
fillCredentialList(decryptedString);
crAdapter= new CredentialsAdapter(this, crList);
listView.setAdapter(crAdapter);
registerForContextMenu(listView);

autoTextView= findViewById(R.id.autoComplete);
autoTextView.setHint(getResources().getString(R.string.auto_comp_anazitisi)+"
"+((GlobalVariables) getApplication()).getFileName());
setAutoCompleteText();

refreshCrList= findViewById(R.id.refreshCrList);
refreshCrList.setOnClickListener(view -> {
decryptedString= readDataFile(password, ((GlobalVariables)
getApplication()).getFileName(),ShowCredentials.this);
```

```
            assert decryptedString!= null;
fillCredentialList(decryptedString);
crAdapter= new CredentialsAdapter(getApplicationContext(), crList);
listView.setAdapter(crAdapter);
});
}

insertCredentials= findViewById(R.id.insertCredentials);
insertCredentials.setOnClickListener(view -> {
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(ShowCredentials.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "InsertCredentials");
editor.apply();
Intent intent = new Intent(ShowCredentials.this, InsertCredentials.class);
startActivity(intent);
});
}


public void setAutoCompleteText() {
ArrayList<String> temp = new ArrayList<>();
        for (Credential c : crList) {
if (!temp.contains(c.getService()))
temp.add(c.getService());
            if (!temp.contains(c.getUsename()))
temp.add(c.getUsename());
            if (!temp.contains(c.getPassword()))
temp.add(c.getPassword());
            for(int i = 0;i<c.getProperties().size();i++){
if (!temp.contains(c.getProperties().get(i).split(":")[1])){
temp.add(c.getProperties().get(i).split(":")[1]);
}
            }
        }
ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
android.R.layout.select_dialog_item, temp);
autoTextView.setThreshold(1); //will start working from first character
autoTextView.setAdapter(adapter);
autoTextView.setOnItemClickListener((parent, view, position, id) -> {
decryptedString= readDataFile(((GlobalVariables)
ShowCredentials.this.getApplication()).getPassword(), ((GlobalVariables)
getApplication()).getFileName(),ShowCredentials.this);
fillCredentialList(Objects.requireNonNull(decryptedString));
crAdapter= new CredentialsAdapter(getApplicationContext(), crList);
listView.setAdapter(crAdapter);
String selected = (String) parent.getItemAtPosition(position);
ArrayList<Credential>tempCr = new ArrayList<>();
            for (Credential cr : crList) {
if(cr.getService().equals(selected) || cr.getUsename().equals(selected) ||
cr.getPassword().equals(selected)){
tempCr.add(cr);
}else{
for(int i = 0;i<cr.getProperties().size();i++) {
if (cr.getProperties().get(i).split(":")[1].equals(selected)) {
tempCr.add(cr);
                            break;
} else {
```

```
//nothing
}
                    }
                }
            }
crList.clear();
crList= tempCr;
crAdapter= new CredentialsAdapter(getApplicationContext(), crList);
listView.setAdapter(crAdapter);
});
}

@Override
public void setEditedCredential(Credential editedCredential, Credential
originalCredential,BooleangiaAntigrafi) throws IOException, DDException {
decryptedString= readDataFile(((GlobalVariables) getApplication()).getPassword(),
((GlobalVariables) getApplication()).getFileName(),this);
        assert decryptedString!= null;
fillCredentialList(decryptedString);
        if(!giaAntigrafi) {
for (int i = 0; i<crList.size(); i++) {
if (crList.get(i).equals(originalCredential)) {
crList.get(i).setService(editedCredential.getService());
crList.get(i).setUsename(editedCredential.getUsename());
crList.get(i).setPassword(editedCredential.getPassword());
crList.get(i).setProperties(editedCredential.getProperties());
}
            }
writeAgainAllCredentials(crList, ((GlobalVariables)
this.getApplication()).getPassword(), ((GlobalVariables)
this.getApplication()).getFileName(),this);
}else{
crList.add(editedCredential);
ArrayList<String>propertiesToAdd=new ArrayList<>();
            for(int i=0;i<editedCredential.getProperties().size();i++){

propertiesToAdd.add(editedCredential.getProperties().get(i).split(":")[0]);
propertiesToAdd.add(editedCredential.getProperties().get(i).split(":")[1]);
}
            Credential toAdd = new
Credential(editedCredential.getService(),editedCredential.getUsename(),editedCrede
ntial.getPassword(),propertiesToAdd);
writeDataFile(toAdd,((GlobalVariables)
this.getApplication()).getPassword(),((GlobalVariables)
getApplication()).getFileName(),ShowCredentials.this);
}
crAdapter= new CredentialsAdapter(this, crList);
listView.setAdapter(crAdapter);
setAutoCompleteText();

}

public void fillCredentialList(ArrayList<String>decryptedString) {
crList.clear();
Credential cr;
        for (String s : decryptedString) {
listNewFieldsDatas= new ArrayList<>();
String[] temp = s.split("\n");
```

```
            if (temp.length>3) {
listNewFieldsDatas.addAll(Arrays.asList(temp).subList(3, temp.length));
}
cr = new Credential(temp[0], temp[1], temp[2], listNewFieldsDatas);
crList.add(cr);
}
    }

@Override
public booleanonCreateOptionsMenu(Menu menu) {
this.getMenuInflater().inflate(R.menu.menuother, menu);
        return true;
}

@Override
public booleanonOptionsItemSelected(MenuItem item) {
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(ShowCredentials.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(ShowCredentials.this, MainMenu.class);
startActivity(intent);
        return super.onOptionsItemSelected(item);
}

@Override
public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfomenuInfo) {
acmi= (AdapterView.AdapterContextMenuInfo) menuInfo;
this.getMenuInflater().inflate(R.menu.listview_menu, menu);
}

@Override
public booleanonContextItemSelected(MenuItem item) {
        String pressedItem = item.getTitle().toString();
        if(pressedItem.equals(getResources().getString(R.string.edit))){
            Credential cr = (Credential)
listView.getItemAtPosition(acmi.position);
EditCredentialDialogeditCredentialDialog = new
EditCredentialDialog(ShowCredentials.this, cr, false);
Objects.requireNonNull(editCredentialDialog.getWindow()).setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));
editCredentialDialog.show();
}else if(pressedItem.equals(getResources().getString(R.string.copyPaste))){
            Credential cr = (Credential)
listView.getItemAtPosition(acmi.position);
EditCredentialDialogeditCredentialDialog = new
EditCredentialDialog(ShowCredentials.this, cr, true);
Objects.requireNonNull(editCredentialDialog.getWindow()).setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));
editCredentialDialog.show();
}else if(pressedItem.equals(getResources().getString(R.string.delete))){
            Credential cr = (Credential)
listView.getItemAtPosition(acmi.position);
            if(crList.size()==1){

Toast.makeText(getApplicationContext(),ShowCredentials.this.getResources().getStri
```

```
ng(R.string.deleteAll),Toast.LENGTH_LONG).show();
                return super.onContextItemSelected(item);
}
crList.remove(cr);
            try {
writeAgainAllCredentials(crList, ((GlobalVariables)
this.getApplication()).getPassword(), ((GlobalVariables)
this.getApplication()).getFileName(),this);
} catch (IOException | DDException e) {
e.printStackTrace();
}
crAdapter= new CredentialsAdapter(this, crList);
listView.setAdapter(crAdapter);
setAutoCompleteText();
}
return super.onContextItemSelected(item);
}

@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener()
                {
@Override
public void onClick(DialogInterface dialog, int which) {
exitApplication(ShowCredentials.this);
}

                })
                .setNegativeButton("No", null)
                .show();
}

@Override
protected void onStart(){
super.onStart();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(ShowCredentials.this);

if(sharedPrefget.getString("Navigation","Logout").equals("ShowCredentials")){
//irthe apo allo activity kai oxi apo lost focus stinefarmogi
}else{
Toast.makeText(ShowCredentials.this,
ShowCredentials.this.getResources().getString(R.string.disconnected),
Toast.LENGTH_LONG).show();
logOut(ShowCredentials.this);
Intent intent = new Intent(ShowCredentials.this,LoginPage.class);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK| Intent.FLAG_ACTIVITY_CLEAR_TASK);
startActivity(intent);
}
    }

@Override
protected void onStop(){
super.onStop();
```

```
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(ShowCredentials.this);

if(sharedPrefget.getString("Navigation","Logout").equals("ShowCredentials")){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(ShowCredentials.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "Logout");
editor.apply();
}else{
//stamatise kai pige se allo activity kai oxi se lost focus stinefarmogi
}
    }

}
```

Κλάση PasswordChange

```
package com.cryptopassword.central;

import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.text.Editable;
import android.text.InputFilter;
import android.text.InputType;
import android.text.TextWatcher;
import android.text.method.PasswordTransformationMethod;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.cryptopassword.helpers.Credential;
import com.cryptopassword.helpers.DDException;
import com.cryptopassword.helpers.GlobalVariables;
import com.cryptopasswordentral.R;
import com.google.android.material.textfield.TextInputLayout;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.central.ExitActivity.logOut;
import static com.cryptopassword.helpers.FileHandler.readDataFile;
import static com.cryptopassword.helpers.FileHandler.readPinFile;
import static com.cryptopassword.helpers.FileHandler.writeAgainAllCredentials;
import static com.cryptopassword.helpers.FileHandler.writePinFile;
```

```
public class PasswordChangeextends AppCompatActivity {

private EditTextoldpass,newPassword,newPasswordVerify;
    private TextInputLayoutnewpass_layout,verifypass_layout;
    private ArrayList<String>decryptedString,listNewFieldsDatas;
    private ArrayList<Credential>credList;
    private TextViewfilename;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.passwordchange);

filename =  findViewById(R.id.filename);
oldpass=  findViewById(R.id.oldpass);
newPassword=  findViewById(R.id.newpass);
newPasswordVerify=  findViewById(R.id.verifypass);

        if(((GlobalVariables) this.getApplication()).getChangePassForLogIn()){
filename.setText(getResources().getString(R.string.loginChange));
InputFilter[] fArray = new InputFilter[1];
fArray[0] = new InputFilter.LengthFilter(4);
oldpass.setHint(R.string.oldpass);
oldpass.setInputType(InputType.TYPE_CLASS_NUMBER);
oldpass.setTransformationMethod(PasswordTransformationMethod.getInstance());
oldpass.setFilters(fArray);
newPassword.setHint(R.string.newpass);
newPassword.setInputType(InputType.TYPE_CLASS_NUMBER);
newPassword.setTransformationMethod(PasswordTransformationMethod.getInstance());
newPassword.setFilters(fArray);
newPasswordVerify.setHint(R.string.verifypin);
newPasswordVerify.setInputType(InputType.TYPE_CLASS_NUMBER);
newPasswordVerify.setTransformationMethod(PasswordTransformationMethod.getInstance
());
newPasswordVerify.setFilters(fArray);
}else{
filename.setText(getResources().getString(R.string.nameWalletOpened)+":"+((GlobalV
ariables) this.getApplication()).getFileName());
oldpass.setHint(R.string.oldpassword);
oldpass.setInputType(InputType.TYPE_CLASS_TEXT);
oldpass.setTransformationMethod(PasswordTransformationMethod.getInstance());
newPassword.setHint(R.string.newpassword);
newPassword.setInputType(InputType.TYPE_CLASS_TEXT);
newPassword.setTransformationMethod(PasswordTransformationMethod.getInstance());
newPasswordVerify.setHint(R.string.verifypassword);
newPasswordVerify.setInputType(InputType.TYPE_CLASS_TEXT);
newPasswordVerify.setTransformationMethod(PasswordTransformationMethod.getInstance
());
}

newPassword= findViewById(R.id.newpass);
newPassword.addTextChangedListener(new TextWatcher() {
public void afterTextChanged(Editable s) {showPasswordsVerifyColor();}
public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
public void onTextChanged(CharSequence s, int start, int before, int count) {}
        });
```

```
newPasswordVerify=  findViewById(R.id.verifypass);
newPasswordVerify.addTextChangedListener(new TextWatcher() {
public void afterTextChanged(Editable s) {showPasswordsVerifyColor();}
public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
public void onTextChanged(CharSequence s, int start, int before, int count) {}
        });
}

private void showPasswordsVerifyColor(){
if (newPassword.getText().toString().length() >0
&&newPasswordVerify.getText().toString().length() >0 &&
newPassword.getText().toString().equals(newPasswordVerify.getText().toString())) {
newPasswordVerify.setBackgroundResource(R.drawable.edit_txt_style_green);
newPassword.setBackgroundResource(R.drawable.edit_txt_style_green);
} else if(newPassword.getText().toString().length() >0
&&newPasswordVerify.getText().toString().length() >0 &&

!newPassword.getText().toString().equals(newPasswordVerify.getText().toString())){
newPasswordVerify.setBackgroundResource(R.drawable.edit_txt_style_red);
newPassword.setBackgroundResource(R.drawable.edit_txt_style_red);
}
    }

public void toChange(View view) throws IOException, DDException {
if(swstaInputs()){
ArrayList<String>decryptedPin = readPinFile(this);
            if(decryptedPin != null &&
!oldpass.getText().toString().equals(decryptedPin.get(0))){
Toast.makeText(PasswordChange.this, R.string.pc4, Toast.LENGTH_LONG).show();
                return;
}
if (((GlobalVariables) this.getApplication()).getChangePassForLogIn()){
writePinFile(newPassword.getText().toString(),this);
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(PasswordChange.this);
String fingerprintSetted=sharedPrefget.getString("FingerPrint","Empty");
                if (!fingerprintSetted.equals("Empty")){
Toast.makeText(PasswordChange.this, R.string.deactivatedFingerprint,
Toast.LENGTH_LONG).show();
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(PasswordChange.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("FingerPrint", "Empty");
editor.apply();
}
Toast.makeText(PasswordChange.this, R.string.PINcreated,
Toast.LENGTH_LONG).show();
((GlobalVariables)
this.getApplication()).setPassword(newPassword.getText().toString());
}else{
if(decryptedString!=null){
fillCredentialList(decryptedString);
}
                ((GlobalVariables)
this.getApplication()).setPassword(newPassword.getText().toString());
writeAgainAllCredentials(credList, newPassword.getText().toString(),
((GlobalVariables) this.getApplication()).getFileName(),this);
Toast.makeText(PasswordChange.this, R.string.pc6, Toast.LENGTH_LONG).show();
```

```
}
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(PasswordChange.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(PasswordChange.this, MainMenu.class);
startActivity(intent);
}
    }

public void fillCredentialList(ArrayList<String>decryptedString) {
credList= new ArrayList<>();
Credential cr;
        for (String s : decryptedString) {
listNewFieldsDatas= new ArrayList<>();
String[] temp = s.split("\n");
            if (temp.length>3) {
listNewFieldsDatas.addAll(Arrays.asList(temp).subList(3, temp.length));
}
cr = new Credential(temp[0], temp[1], temp[2], listNewFieldsDatas);
credList.add(cr);
}
    }

public booleanswstaInputs(){
if (((GlobalVariables) this.getApplication()).getChangePassForLogIn()){
if(oldpass.getText().length()!=4 || newPassword.getText().length()!=4 ||
newPasswordVerify.getText().length()!=4){
Toast.makeText(PasswordChange.this, R.string.pc3, Toast.LENGTH_LONG).show();
                return false;
}
ArrayList<String>decryptedPin = readPinFile(this);
            if (decryptedPin != null &&
!oldpass.getText().toString().equals(decryptedPin.get(0))){
Toast.makeText(PasswordChange.this, R.string.pc4, Toast.LENGTH_LONG).show();
}
//            if(!oldpass.getText().toString().equals(((GlobalVariables)
this.getApplication()).getPassword())){
//                Toast.makeText(PasswordChange.this, R.string.pc4,
Toast.LENGTH_LONG).show();
//                return false;
//            }
if(!newPassword.getText().toString().equals(newPasswordVerify.getText().toString()
)){
Toast.makeText(PasswordChange.this, R.string.pc1, Toast.LENGTH_LONG).show();
                return false;
}
if(newPassword.getText().toString().equals(oldpass.getText().toString())){
Toast.makeText(PasswordChange.this, R.string.pc5, Toast.LENGTH_LONG).show();
                return false;
}
return true;
}else{
if(oldpass.getText().length()<10|| newPassword.getText().length()<10 ||
newPasswordVerify.getText().length()<10){
Toast.makeText(PasswordChange.this, R.string.minPassword,
Toast.LENGTH_LONG).show();
```

```
                return false;
}
decryptedString= readDataFile(oldpass.getText().toString(),((GlobalVariables)
this.getApplication()).getFileName(),this);
            if(decryptedString== null){

Toast.makeText(this,this.getResources().getString(R.string.falsePassword),
Toast.LENGTH_SHORT).show();
                return false;
}
if(!newPassword.getText().toString().equals(newPasswordVerify.getText().toString()
)){
Toast.makeText(PasswordChange.this, R.string.notMatchingPasswords,
Toast.LENGTH_LONG).show();
                return false;
}
if(newPassword.getText().toString().equals(oldpass.getText().toString())){
Toast.makeText(PasswordChange.this, R.string.newEqualOldPassword,
Toast.LENGTH_LONG).show();
                return false;
}
return true;
}
    }

@Override
public booleanonCreateOptionsMenu(Menu menu) {
this.getMenuInflater().inflate(R.menu.menuother, menu);
        return true;
}

@Override
public booleanonOptionsItemSelected(MenuItem item) {
SharedPreferencessharedPref = PreferenceManager.getDefaultSharedPreferences(this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(PasswordChange.this, MainMenu.class);
startActivity(intent);
        return super.onOptionsItemSelected(item);
}

@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener()
                {
@Override
public void onClick(DialogInterface dialog, int which) {
exitApplication(PasswordChange.this);
}

                })
                .setNegativeButton("No", null)
                .show();
```

```
}

@Override
protected void onStart(){
super.onStart();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(PasswordChange.this);

if(sharedPrefget.getString("Navigation","Logout").equals("PasswordChange")){
//irthe apo allo activity kai oxi apo lost focus stinefarmogi
}else{
Toast.makeText(PasswordChange.this,
PasswordChange.this.getResources().getString(R.string.disconnected),
Toast.LENGTH_LONG).show();
logOut(PasswordChange.this);
Intent intent = new Intent(PasswordChange.this,LoginPage.class);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK| Intent.FLAG_ACTIVITY_CLEAR_TASK);
startActivity(intent);
}
    }

@Override
protected void onStop(){
super.onStop();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(PasswordChange.this);

if(sharedPrefget.getString("Navigation","Logout").equals("PasswordChange")){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(PasswordChange.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "Logout");
editor.apply();
}else{
//stamatise kai pige se allo activity kai oxi se lost focus stinefarmogi
}
    }

}
```

Κλάση ImpExpMenu

```
package com.cryptopassword.central;

import android.content.ActivityNotFoundException;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.provider.OpenableColumns;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Button;
import android.widget.EditText;
```

```
import android.widget.Toast;

import com.cryptopassword.helpers.Credential;
import com.cryptopassword.helpers.DDException;
import com.cryptopassword.helpers.GlobalVariables;
import com.cryptopasswordentral.R;

import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;

import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.content.FileProvider;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.helpers.FileHandler.decryptString;
import static com.cryptopassword.helpers.FileHandler.destroyFile;
import static com.cryptopassword.helpers.FileHandler.getAbsoluteFile;
import static com.cryptopassword.helpers.FileHandler.getAllFileNames;
import static com.cryptopassword.helpers.FileHandler.readDataFile;
import static com.cryptopassword.helpers.FileHandler.writeAgainAllCredentials;
import static com.cryptopassword.helpers.FileHandler.writeXLSFile;

public class ImpExpMenuextends AppCompatActivity {

private static final int OPEN_REQUEST_CODE_TXT = 41;
    private static final int OPEN_REQUEST_CODE_EXCEL = 42;
    private Button exportExcel, exportWallet,importExcel,importWallet,
deleteExcel;
    private EditTextpinForImport,newPinForImport,verifynewPinForImport;
    private ArrayList<String>decryptedString;
    private ArrayList<Credential>crList= new ArrayList<>();
String excelFile;

@Override
protected void onCreate(Bundle savedInstanceState){
super.onCreate(savedInstanceState);
setContentView(R.layout.impexp_menu);
verifynewPinForImport= findViewById(R.id.verifynewPinForImport);
exportExcel= findViewById(R.id.exportExcelMail);
exportWallet= findViewById(R.id.exportWalletMail);
importWallet= findViewById(R.id.importWallet);
```

```
importExcel= findViewById(R.id.importExcel);
pinForImport= findViewById(R.id.pinForImport);
newPinForImport= findViewById(R.id.newPinForImport);
deleteExcel= findViewById(R.id.deleteExcelMail);
        if (((GlobalVariables) this.getApplication()).getFileName()!=null)
excelFile= ((GlobalVariables)
this.getApplication()).getFileName().substring(0,((GlobalVariables)
this.getApplication()).getFileName().length()-4)+".xls";
setDeleteButtonText();
setExportButtons();

deleteExcel.setOnClickListener(view -> {
try {
destroyFile(((GlobalVariables)
getApplication()).getFileName().substring(0,((GlobalVariables)
getApplication()).getFileName().length()-4)+".xls",ImpExpMenu.this);
setDeleteButtonText();
} catch (DDException e) {
e.printStackTrace();
}
        });

exportWallet.setOnClickListener(view -> {
try {
sendFile("wallet");
} catch (IOException e) {
e.printStackTrace();
}
        });

exportExcel.setOnClickListener(view -> {
try {
sendFile("xls");
} catch (IOException e) {
e.printStackTrace();
}
        });
importWallet.setOnClickListener(view -> {
importFile("wallet");
});

importExcel.setOnClickListener(view -> {
if(swstaInputs())
importFile("xls");
});


newPinForImport.addTextChangedListener(new TextWatcher() {
public void afterTextChanged(Editable s) {showPasswordsVerifyColor();}
public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
public void onTextChanged(CharSequence s, int start, int before, int count) {}
        });

verifynewPinForImport.addTextChangedListener(new TextWatcher() {
public void afterTextChanged(Editable s) {showPasswordsVerifyColor();}
public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
public void onTextChanged(CharSequence s, int start, int before, int count) {}
        });
```

```
}

private void showPasswordsVerifyColor(){
if (newPinForImport.getText().toString().length() >0
&&verifynewPinForImport.getText().toString().length() >0 &&
newPinForImport.getText().toString().equals(verifynewPinForImport.getText().toStri
ng())) {
verifynewPinForImport.setBackgroundResource(R.drawable.edit_txt_style_green);
newPinForImport.setBackgroundResource(R.drawable.edit_txt_style_green);
} else if(newPinForImport.getText().toString().length() >0
&&verifynewPinForImport.getText().toString().length() >0 &&

!newPinForImport.getText().toString().equals(verifynewPinForImport.getText().toStr
ing())){
verifynewPinForImport.setBackgroundResource(R.drawable.edit_txt_style_red);
newPinForImport.setBackgroundResource(R.drawable.edit_txt_style_red);
}
    }

public booleanswstaInputs(){
if(newPinForImport.getText().length()<10 ||
verifynewPinForImport.getText().length()<10){
Toast.makeText(ImpExpMenu.this, R.string.minPassword, Toast.LENGTH_LONG).show();
            return false;
}
if(!newPinForImport.getText().toString().equals(verifynewPinForImport.getText().to
String())){
Toast.makeText(ImpExpMenu.this, R.string.notMatchingPasswords,
Toast.LENGTH_LONG).show();
            return false;
}
return true;
}

private void setDeleteButtonText(){
if(getAllFileNames(ImpExpMenu.this).contains(excelFile)){
deleteExcel.setText(getResources().getString(R.string.deleteExcelFile)+excelFile);
}else{
deleteExcel.setText(getResources().getString(R.string.noDeleteExcelFile));
}
    }

private void setExportButtons(){
if(((GlobalVariables) this.getApplication()).getFileName()==null) {
exportExcel.setEnabled(false);
exportExcel.setBackgroundColor(getResources().getColor(R.color.mycolor3,null));
deleteExcel.setEnabled(false);
deleteExcel.setBackgroundColor(getResources().getColor(R.color.mycolor3,null));
exportWallet.setEnabled(false);
exportWallet.setBackgroundColor(getResources().getColor(R.color.mycolor3,null));
}else{
exportExcel.setEnabled(true);
exportExcel.setBackground(getResources().getDrawable(R.drawable.general_btn,null))
;
deleteExcel.setEnabled(true);
deleteExcel.setBackground(getResources().getDrawable(R.drawable.general_btn,null))
;
exportWallet.setEnabled(true);
```

```
exportWallet.setBackground(getResources().getDrawable(R.drawable.general_btn,null)
);
}
    }



private void createExcelFile() throws IOException {
decryptedString= readDataFile(((GlobalVariables)
this.getApplication()).getPassword(), ((GlobalVariables)
this.getApplication()).getFileName(),this);
ArrayList<Credential>crList = new ArrayList<>();
ArrayList<String>listNewFieldsDatas;
Credential cr;
        for (String s : decryptedString) {
listNewFieldsDatas = new ArrayList<>();
String[] temp = s.split("\n");
            if (temp.length>3) {
listNewFieldsDatas.addAll(Arrays.asList(temp).subList(3, temp.length));
}
cr = new Credential(temp[0], temp[1], temp[2], listNewFieldsDatas);
crList.add(cr);
}
writeXLSFile(excelFile,crList,ImpExpMenu.this);
}

private void sendFile(String fileType) throws IOException {
        Date c = Calendar.getInstance().getTime();
SimpleDateFormat df = new SimpleDateFormat("dd/MMM/yyyyhh:mm:ss",
Locale.getDefault());
String formattedDate = df.format(c);
File applicationPath = getAbsoluteFile("", ImpExpMenu.this);
File file = null;
        if(fileType.equals("wallet")) {
            file = new File(applicationPath, "/CryptoPassword/" +
((GlobalVariables) this.getApplication()).getFileName());
            if (!file.exists()) {
return;
}
        }else if(fileType.equals("xls")){
createExcelFile();
file = new File(applicationPath, "/CryptoPassword/" + excelFile);
            if (!file.exists()) {
return;
}
        }
        Intent i = new Intent(Intent.ACTION_SEND);
Uri uri = FileProvider.getUriForFile(ImpExpMenu.this,
ImpExpMenu.this.getApplicationContext().getPackageName() + ".provider", file);
i.addFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);
i.setType("vnd.android.cursor.dir/email");
i.putExtra(Intent.EXTRA_SUBJECT,
getResources().getString(R.string.encrWalletMailSubject));
        if(fileType.equals("wallet")) {
i.putExtra(Intent.EXTRA_TEXT,
getResources().getString(R.string.encrWalletMailBody1) + ((GlobalVariables)
this.getApplication()).getFileName() +
" " + getResources().getString(R.string.excelMailBody2) + formattedDate);
```

```
}else if(fileType.equals("xls")){
i.putExtra(Intent.EXTRA_TEXT, getResources().getString(R.string.excelMailBody1) +
excelFile+ " " + getResources().getString(R.string.excelMailBody2) +
formattedDate);
}
i.putExtra(Intent.EXTRA_STREAM, uri);
        try {
setDeleteButtonText();
startActivity(Intent.createChooser(i, "Send mail..."));
} catch (ActivityNotFoundException ex) {
Toast.makeText(ImpExpMenu.this, "There are no email clients installed.",
Toast.LENGTH_SHORT).show();
}
    }

private void importFile(String fileType){
if(fileType.equals("wallet")){
            Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
intent.addCategory(Intent.CATEGORY_OPENABLE);
intent.setType("text/plain");
startActivityForResult(intent, OPEN_REQUEST_CODE_TXT);
}else if(fileType.equals("xls")){
            Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
intent.addCategory(Intent.CATEGORY_OPENABLE);
intent.setType("application/vnd.ms-excel");
startActivityForResult(intent, OPEN_REQUEST_CODE_EXCEL);
}
    }

public void onActivityResult(int requestCode, int resultCode,IntentresultData) {
super.onActivityResult(requestCode, resultCode, resultData);
Uri currentUri;
        if (requestCode == OPEN_REQUEST_CODE_TXT) {
if (resultData != null) {
currentUri = resultData.getData();
Cursor returnCursor = getContentResolver().query(currentUri, null, null, null,
null);
                int nameIndex =
returnCursor.getColumnIndex(OpenableColumns.DISPLAY_NAME);
returnCursor.moveToFirst();
String importFileName = returnCursor.getString(nameIndex);
                try {
                    String content = readFileContent(currentUri);
decryptedString= decryptString(pinForImport.getText().toString(),content);
                    if(decryptedString!=null)
fillCredentialList(decryptedString);
                    if(crList.size()>0){
writeAgainAllCredentials(crList, pinForImport.getText().toString(),
importFileName,this);
Toast.makeText(this,
this.getResources().getString(R.string.walletImported)+importFileName,
Toast.LENGTH_SHORT).show();
((GlobalVariables) this.getApplication()).setFileName(importFileName);
((GlobalVariables)getApplication()).setFirstLogIn(false);
((GlobalVariables)
getApplication()).setPassword(pinForImport.getText().toString());
pinForImport.setText("");
}
```

```
                } catch (IOException | DDException e) {
e.printStackTrace();
}
            }
        }
if (requestCode == OPEN_REQUEST_CODE_EXCEL) {
if (resultData != null) {
currentUri = resultData.getData();
Cursor returnCursor = getContentResolver().query(currentUri, null, null, null,
null);
                int nameIndex =
returnCursor.getColumnIndex(OpenableColumns.DISPLAY_NAME);
returnCursor.moveToFirst();
String importFileName = returnCursor.getString(nameIndex);
                if(importFileName.contains(".xls")) {
try {
InputStream inputStream = getContentResolver().openInputStream(currentUri);
                        if (inputStream != null) {
                            Workbook workbook = new HSSFWorkbook(inputStream);
Sheet datatypeSheet = workbook.getSheetAt(0);
                            int rowNum=0;
ArrayList<Credential>crList= new ArrayList<>();
ArrayList<String>newFields= new ArrayList<>();
ArrayList<String>newFieldsData= new ArrayList<>();
ArrayList<String>newFDTelikiLista= new ArrayList<>();
                            for (Row currentRow : datatypeSheet) {
                                String service = null,username = null,password =
null;
                                int cellNum=0;
                                for (Cell currentCell : currentRow) {
if(rowNum==0){
if(cellNum==0 && !currentCell.getStringCellValue().equals("Υπηρεσία") &&
!currentCell.getStringCellValue().equals("Service")){
Toast.makeText(this, this.getResources().getString(R.string.firstColumnError),
Toast.LENGTH_SHORT).show();
                                                return;
}else if(cellNum==1 &&
!currentCell.getStringCellValue().equals(getResources().getString(R.string.usernam
e))){
Toast.makeText(this, this.getResources().getString(R.string.secondColumnError),
Toast.LENGTH_SHORT).show();
                                                return;
}else if(cellNum==2 &&
!currentCell.getStringCellValue().equals(getResources().getString(R.string.passwor
d))){
Toast.makeText(this, this.getResources().getString(R.string.thirdColumnError),
Toast.LENGTH_SHORT).show();
                                                return;
}else if(cellNum>2){
newFields.add(currentCell.getStringCellValue());
}
                                    }else{
if(cellNum==0){
                                                service = currentCell.toString();
}else if(cellNum==1){
                                                username = currentCell.toString();
}else if(cellNum==2){
                                                password = currentCell.toString();
```

```
}else if(cellNum>2){
newFieldsData.add(currentCell.toString());
}
                                    }
cellNum++;
}
if(rowNum!=0){
for(int i=0;i<newFields.size();i++){
newFDTelikiLista.add(newFields.get(i)+":"+newFieldsData.get(i));
}
crList.add(new Credential(service,username,password,newFDTelikiLista));
newFieldsData= new ArrayList<>();
newFDTelikiLista= new ArrayList<>();
}
rowNum++;
}
                            String fileName = importFileName.substring(0,
importFileName.length() - 4) + ".txt";
writeAgainAllCredentials(crList,newPinForImport.getText().toString(),
fileName,ImpExpMenu.this);
Toast.makeText(this,
this.getResources().getString(R.string.walletImported)+importFileName,
Toast.LENGTH_SHORT).show();
((GlobalVariables) this.getApplication()).setFileName(fileName);
((GlobalVariables)getApplication()).setFirstLogIn(false);
((GlobalVariables)
getApplication()).setPassword(newPinForImport.getText().toString());
newPinForImport.setText("");
newPinForImport.setBackgroundResource(R.drawable.edit_txt_style);
verifynewPinForImport.setText("");
verifynewPinForImport.setBackgroundResource(R.drawable.edit_txt_style);
}
                    } catch (IOException | DDException e) {
e.printStackTrace();
}
                }
            }
        }

    }

private String readFileContent(Uri uri) throws IOException {
InputStreaminputStream = getContentResolver().openInputStream(uri);
BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));
StringBuilder stringBuilder = new StringBuilder();
        int c;
        while ((c = reader.read()) != -1) {
//eprepenadiavasoxaraktiraxaraktiragiati to string eixe mesa to /r kai
stamatagenadaiavazei me to readLine
char character = (char) c;
stringBuilder.append(character);
}
inputStream.close();
        return stringBuilder.toString();
}

private void fillCredentialList(ArrayList<String>decryptedString) {
crList.clear();
```

```
Credential cr;
        for (String s : decryptedString) {
ArrayList<String>listNewFieldsDatas = new ArrayList<>();
String[] temp = s.split("\n");
            if (temp.length>3) {
listNewFieldsDatas.addAll(Arrays.asList(temp).subList(3, temp.length));
}
cr = new Credential(temp[0], temp[1], temp[2], listNewFieldsDatas);
crList.add(cr);
}
    }

@Override
public booleanonCreateOptionsMenu(Menu menu) {
this.getMenuInflater().inflate(R.menu.menuother, menu);
        return true;
}

@Override
public booleanonOptionsItemSelected(MenuItem item) {
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(ImpExpMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(ImpExpMenu.this, MainMenu.class);
startActivity(intent);
        return super.onOptionsItemSelected(item);
}

@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", (dialog, which) -
>exitApplication(ImpExpMenu.this))
                .setNegativeButton("No", null)
                .show();
}

}
```

Κλάση Generator

```
package com.cryptopassword.central;

import android.content.ClipData;
import android.content.ClipboardManager;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
```

```
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import com.cryptopasswordentral.R;

import java.util.Random;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.central.ExitActivity.logOut;

/**
 * @author ddamanakis
 * @version 1.0
 */

public class Generator extends AppCompatActivity {

private SeekBarseekBar;
    private TextViewtextProgress,showpassword;
    private CheckBoxaz,AZ,nums,symbols;
    private Button generate,proxeiro;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.generator);

seekBar= findViewById(R.id.seekBar);
textProgress= findViewById(R.id.textProgress);
showpassword=findViewById(R.id.textView5);
az= findViewById(R.id.az);
AZ = findViewById(R.id.AZ);
nums= findViewById(R.id.zero);
symbols = findViewById(R.id.symbols);
generate = findViewById(R.id.generate);
proxeiro= findViewById(R.id.copy);
textProgress.setText(6+" bit");
az.setChecked(true);
seekBar.setMax(26);

seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
@Override
public void onStopTrackingTouch(SeekBarseekBar) {}
@Override
public void onStartTrackingTouch(SeekBarseekBar) {}
@Override
public void onProgressChanged(SeekBarseekBar, int progress,booleanfromUser) {
textProgress.setText(progress+6+" bit");
}
        });
```

```
}

/* Δημιουργία του password ανάλογα με τις επιλογές που έχει κάνει οχρήστης */
public void generatepass(View view){
if (!az.isChecked() && !AZ.isChecked() && !nums.isChecked() &&
!symbols.isChecked()) {
Toast.makeText(Generator.this, R.string.g1, Toast.LENGTH_SHORT).show();
}
else {
int passwordSize = (seekBar.getProgress() + 6);
            char[] chars = "abcdefghijklmnopqrstuvwxyz".toCharArray();
            char[] Cchars = "QWERTYUIOPLKJHGFDSAZXCVBNM".toCharArray();
            char[] numsar = "0123456789".toCharArray();
            char[] symbolsar = "!@#$%^&*()_".toCharArray();
StringBuilder sb = new StringBuilder();
Random random = new Random();
        if (az.isChecked() &&AZ.isChecked() &&nums.isChecked()
&&symbols.isChecked()){
char[] temp =
"abcdefghijklmnopqrstuvwxyzQWERTYUIOPLKJHGFDSAZXCVBNM0123456789!@#$%^&*()_".toChar
Array();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
            }else if(az.isChecked() &&AZ.isChecked() &&nums.isChecked()) {
char[] temp =
"abcdefghijklmnopqrstuvwxyzQWERTYUIOPLKJHGFDSAZXCVBNM0123456789".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
             }else if(az.isChecked() &&AZ.isChecked()  &&symbols.isChecked()) {
char[] temp =
"abcdefghijklmnopqrstuvwxyzQWERTYUIOPLKJHGFDSAZXCVBNM0!@#$%^&*()_".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
            }else if(az.isChecked() &&nums.isChecked() &&symbols.isChecked()){
char[] temp = "abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()_".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }else if( AZ.isChecked() &&nums.isChecked() &&symbols.isChecked()){
char[] temp = "QWERTYUIOPLKJHGFDSAZXCVBNM0123456789!@#$%^&*()_".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }else if(az.isChecked() &&AZ.isChecked() ){
char[] temp =
"abcdefghijklmnopqrstuvwxyzQWERTYUIOPLKJHGFDSAZXCVBNM".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }else if(az.isChecked() &&nums.isChecked()){
char[] temp = "abcdefghijklmnopqrstuvwxyz0123456789".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }else if (az.isChecked()  &&symbols.isChecked()){
```

```
char[] temp = "abcdefghijklmnopqrstuvwxyz!@#$%^&*()_".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }else if(AZ.isChecked() &&nums.isChecked() ){
char[] temp = "QWERTYUIOPLKJHGFDSAZXCVBNM0123456789".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }else if(AZ.isChecked() &&symbols.isChecked()){
char[] temp = "QWERTYUIOPLKJHGFDSAZXCVBNM!@#$%^&*()_".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }else if(nums.isChecked() &&symbols.isChecked()){
char[] temp = "0123456789!@#$%^&*()_".toCharArray();
            for (int i = 0; i<passwordSize; i++) {
char c = temp[random.nextInt(temp.length)];
sb.append(c);}
        }
else if (az.isChecked()){
for (int i = 0; i<passwordSize; i++) {
char c = chars[random.nextInt(chars.length)];
sb.append(c);
}
            }else if(AZ.isChecked()){
for (int i = 0; i<passwordSize; i++) {
char c = Cchars[random.nextInt(Cchars.length)];
sb.append(c);
}
            }else if(nums.isChecked()){
for (int i = 0; i<passwordSize; i++) {
char c = numsar[random.nextInt(numsar.length)];
sb.append(c);
}
            }else if(symbols.isChecked()){
for (int i = 0; i<passwordSize; i++) {
char c = symbolsar[random.nextInt(symbolsar.length)];
sb.append(c);
}
            }
            String output = sb.toString();
showpassword.setText(output);
}
    }

/* Αντιγραφή του παραγόμενου password στο πρόχειρο τη ςυσκευής */
public void antigrafiprox(View view){
if (showpassword.getText().toString().isEmpty()) {
Toast.makeText(Generator.this,R.string.g2, Toast.LENGTH_SHORT).show();
}else {
ClipboardManager clipboard = (ClipboardManager)
getSystemService(Context.CLIPBOARD_SERVICE);
ClipData clip = ClipData.newPlainText("", showpassword.getText().toString());
clipboard.setPrimaryClip(clip);
Toast.makeText(Generator.this, R.string.g3, Toast.LENGTH_SHORT).show();
}
    }
```

```
/* Δημιουργία menu */
@Override
public booleanonCreateOptionsMenu(Menu menu) {
this.getMenuInflater().inflate(R.menu.menuother, menu);
        return true;
}

/* Navigation σεεπόμενο activity μέσωτου menu */
@Override
public booleanonOptionsItemSelected(MenuItem item) {
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(Generator.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(Generator.this, MainMenu.class);
startActivity(intent);
        return super.onOptionsItemSelected(item);
}

/* Dialog για έξοδο από την εφαρμογή όταν ο χρήστης πατάει το κουμπί πίσω */
@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener()
                {
@Override
public void onClick(DialogInterface dialog, int which) {
exitApplication(Generator.this);
}

                })
                .setNegativeButton("No", null)
                .show();
}

/* Έλεγχο όταν ξεκινάει το activity για το αν ο χρήστης ήρθε από προηγούμενο
activityόποτε τον αφήνει να συνεχίσει ή ήρθε με άλλο τρόπο οπότε και τον
αποσυνδέει */
@Override
protected void onStart(){
super.onStart();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(Generator.this);
        if (!sharedPrefget.getString("Navigation", "Logout").equals("Generator"))
{
Toast.makeText(Generator.this,
Generator.this.getResources().getString(R.string.disconnected),
Toast.LENGTH_LONG).show();
logOut(Generator.this);
Intent intent = new Intent(Generator.this,LoginPage.class);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK| Intent.FLAG_ACTIVITY_CLEAR_TASK);
startActivity(intent);
}
```

```
    }

/* Έλεγχο πως πήγε σε άλλο activity και όχι lost focus */
@Override
protected void onStop(){
super.onStop();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(Generator.this);
        if(sharedPrefget.getString("Navigation","Logout").equals("Generator")){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(Generator.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "Logout");
editor.apply();
}
    }

}
```

Κλάση FastLoginMenu

```
package com.cryptopassword.central;


import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.cryptopasswordentral.R;

import java.util.ArrayList;
import java.util.concurrent.Executor;
import java.util.concurrent.Executors;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.biometric.BiometricPrompt;
import androidx.preference.PreferenceManager;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.central.ExitActivity.logOut;
import static com.cryptopassword.helpers.FileHandler.readPinFile;

/**
 * Η κλάση χρησιμοποιείται με σκοπό ο χρήστης να ενεργοποιήσει/απενεργοποιήσει το
fast login μέσω δακτυλικού αποτυπώματος στην εφαρμογή
 * @author ddamanakis
 * @version 1.0
 */

public class FastLoginMenuextends AppCompatActivity {
```

```
private Button deleteFingerPrint,setFingerPrint;
    private EditTextpasswordForFinegerprint;
    private String fingerprintSetted;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.fast_log_in);

deleteFingerPrint= findViewById(R.id.deleteFingerPrint);
setFingerPrint= findViewById(R.id.setFingerPrint);
passwordForFinegerprint= findViewById(R.id.passwordForFinegerprint);
setButtons();

/* prompt για χρήση δακτυλικού αποτυπώματος */
Executor newExecutor = Executors.newSingleThreadExecutor();
        final BiometricPrompt.PromptInfopromptInfo = new
BiometricPrompt.PromptInfo.Builder()
                .setTitle(getResources().getString(R.string.fingerprintTitle))

.setSubtitle(getResources().getString(R.string.fingerprintSubTitle))
                .setNegativeButtonText("Cancel")
                .build();

/* Ενέργειες ανάλογα με το αποτέλεσμα της χρήσης δακτυλικού αποτυπώματος */
final BiometricPromptmyBiometricPrompt = new BiometricPrompt(FastLoginMenu.this,
newExecutor, new BiometricPrompt.AuthenticationCallback() {
@Override
public void onAuthenticationSucceeded(@NonNull
BiometricPrompt.AuthenticationResult result) {
super.onAuthenticationSucceeded(result);
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(FastLoginMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("FingerPrint", "Setted");
editor.apply();
showResult("OK");
}

@Override
public void onAuthenticationFailed() {
super.onAuthenticationFailed();
showResult("Error");
}

@Override
public void onAuthenticationError(int errorCode, @NonNull CharSequenceerrString) {
super.onAuthenticationError(errorCode, errString);
showResult(errString.toString());
}

        });

/* Ανάλογα με το Pin που έδωσε ο χρήστης αν είναι σωστό ενεργοποιείται
η λειτουργία του δακτυλικού αποτυπώματος αλλιώς εμφανίζεται το ανάλογο μήνυμα */
setFingerPrint.setOnClickListener(view -> {
if(passwordForFinegerprint.getText().toString().length()!=4){
```

```
Toast.makeText(FastLoginMenu.this, R.string.pc3,Toast.LENGTH_SHORT).show();
}
ArrayList<String>decryptedPin = readPinFile(this);
            if (decryptedPin != null
&&passwordForFinegerprint.getText().toString().equals(decryptedPin.get(0))){
Toast.makeText(FastLoginMenu.this, R.string.rightPIN,Toast.LENGTH_SHORT).show();
myBiometricPrompt.authenticate(promptInfo);
}else{
Toast.makeText(FastLoginMenu.this, R.string.falsePIN,Toast.LENGTH_SHORT).show();
}
        });

/* Ανάλογα με το Pin που έδωσε ο χρήστης αν είναι σωστό απενεργοποιείται
η λειτουργία του δακτυλικού αποτυπώματος αλλιώς εμφανίζεται το ανάλογο μήνυμα */
deleteFingerPrint.setOnClickListener(view -> {
if(passwordForFinegerprint.getText().toString().length()!=4){
Toast.makeText(FastLoginMenu.this, R.string.pc3,Toast.LENGTH_SHORT).show();
}
ArrayList<String>decryptedPin = readPinFile(this);
            if (decryptedPin != null
&&passwordForFinegerprint.getText().toString().equals(decryptedPin.get(0))){
Toast.makeText(FastLoginMenu.this, R.string.rightPIN,Toast.LENGTH_SHORT).show();
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(FastLoginMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("FingerPrint", "Empty");
editor.apply();
setButtons();
}else{
Toast.makeText(FastLoginMenu.this, R.string.falsePIN,Toast.LENGTH_SHORT).show();
}
        });

}

/* Ανάλογαμε την κατάσταση που βρίσκεται το fast login
ενεργοποιούνται/απενεργοποιούνται τα κατάλληλα κουμπιά  */
private void setButtons(){
passwordForFinegerprint.setText("");
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(FastLoginMenu.this);
fingerprintSetted=sharedPrefget.getString("FingerPrint","Empty");
        if(fingerprintSetted.equals("Setted")){
deleteFingerPrint.setEnabled(true);
deleteFingerPrint.setBackground(getResources().getDrawable(R.drawable.general_btn,
null));
setFingerPrint.setEnabled(false);
setFingerPrint.setBackgroundColor(getResources().getColor(R.color.mycolor3,null));
}else if(fingerprintSetted.equals("Empty")){
setFingerPrint.setEnabled(true);
setFingerPrint.setBackground(getResources().getDrawable(R.drawable.general_btn,nul
l));
deleteFingerPrint.setEnabled(false);
deleteFingerPrint.setBackgroundColor(getResources().getColor(R.color.mycolor3,null
));
}
    }
```

```
/* Εμφάνιση αποτελέσματος και περεταίρω λειτουργίες ανάλογα
με το αποτέλεσμα από το scan του δακτυλικού αποτυπώματος */
private void showResult(String result){
        Thread thread = new Thread(){
public void run(){
runOnUiThread(() -> {
if(!result.equals("Ok")) {
Toast.makeText(FastLoginMenu.this, result, Toast.LENGTH_SHORT).show();
setButtons();
}
                });
}
        };
thread.start();
}

/* Δημιουργία menu */
@Override
public booleanonCreateOptionsMenu(Menu menu) {
this.getMenuInflater().inflate(R.menu.menuother, menu);
        return true;
}

/* Navigation σε επόμενο activity μέσω του menu */
@Override
public booleanonOptionsItemSelected(MenuItem item) {
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(FastLoginMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "MainMenu");
editor.apply();
Intent intent = new Intent(FastLoginMenu.this, MainMenu.class);
startActivity(intent);
        return super.onOptionsItemSelected(item);
}

/* Dialog για έξοδο από την εφαρμογή όταν ο χρήστης πατάει το κουμπί πίσω */
@Override
public void onBackPressed() {
new AlertDialog.Builder(this)
                .setIcon(R.drawable.log_off)
                .setTitle(R.string.prosoxi)
                .setMessage(R.string.exodosApoEfarmogi)
                .setPositiveButton("Yes", (dialog, which) -
>exitApplication(FastLoginMenu.this))
                .setNegativeButton("No", null)
                .show();
}

/* Έλεγχο όταν ξεκινάει το activity για το αν ο χρήστης ήρθε από προηγούμενο
activityόποτε τον αφήνει να συνεχίσει ή ήρθε με άλλο τρόπο οπότε και τον
αποσυνδέει */
@Override
protected void onStart(){
super.onStart();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(this);
        if (!sharedPrefget.getString("Navigation",
```

```
"Logout").equals("FastLoginMenu")) {
Toast.makeText(FastLoginMenu.this,
FastLoginMenu.this.getResources().getString(R.string.disconnected),
Toast.LENGTH_LONG).show();
logOut(FastLoginMenu.this);
Intent intent = new Intent(FastLoginMenu.this,LoginPage.class);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK| Intent.FLAG_ACTIVITY_CLEAR_TASK);
startActivity(intent);
}
    }

/* Έλεγχο πω ςπήγε σε άλλο activity και όχι lost focus */
@Override
protected void onStop(){
super.onStop();
SharedPreferencessharedPrefget =
PreferenceManager.getDefaultSharedPreferences(this);

if(sharedPrefget.getString("Navigation","Logout").equals("FastLoginMenu")){
SharedPreferencessharedPref =
PreferenceManager.getDefaultSharedPreferences(FastLoginMenu.this);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("Navigation", "Logout");
editor.apply();
}
    }

}
```

Κλάση ExitActivity

```
package com.cryptopassword.central;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;

import com.cryptopassword.helpers.GlobalVariables;

/**
 * Η κλάση χρησιμοποιείται με σκοπό να καθαρίσει η εφαρμογή και οι μεταβλητές της
από την μνήμη της συσκευής και να πραγματοποιηθεί έξοδος
 * @author ddamanakis
 * @version 1.0
 */

public class ExitActivityextends Activity{

@Override protected void onCreate(Bundle savedInstanceState){
super.onCreate(savedInstanceState);
finishAndRemoveTask();
}

public static void exitApplication(Context context){
        Intent intent = new Intent(context, ExitActivity.class);
```

```
intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK| Intent.FLAG_ACTIVITY_CLEAR_TASK|
Intent.FLAG_ACTIVITY_NO_ANIMATION| Intent.FLAG_ACTIVITY_EXCLUDE_FROM_RECENTS);
((GlobalVariables) ((Activity) context).getApplication()).setFileName(null);
context.startActivity(intent);
}

public static void logOut(Context context){
        ((GlobalVariables) ((Activity)
context).getApplication()).setFileName(null);
((GlobalVariables) ((Activity) context).getApplication()).setPassword(null);
}
}
```

Κλάση Credential

```
package com.cryptopassword.helpers;

import java.util.ArrayList;

public class Credential {

private String service;
    private String usename;
    private String password;
    private ArrayList<String>properties;

    public Credential(String
service,Stringusename,Stringpassword,ArrayList<String> properties) {
this.service= service ;
this.usename= usename;
this.password= password ;
this.properties= properties;
}

public booleanequals(Credential cr){
booleanflag = false;
        if (cr.getService().equals(service) &&cr.getUsename().equals(usename)
&&cr.getPassword().equals(password)){
                flag = true;
}
for(int i=0;i<cr.getProperties().size();i++){
if(!cr.getProperties().get(i).equals(this.properties.get(i))){
                flag=false;
}
        }
return flag;
}

public String getService() {
return service;
}

public void setService(String service) {
this.service= service;
}

public String getUsename() {
```

```
return usename;
}

public void setUsename(String usename) {
this.usename= usename;
}

public String getPassword() {
return password;
}

public void setPassword(String password) {
this.password= password;
}

public ArrayList<String>getProperties() {
return properties;
}

public void setProperties(ArrayList<String> properties) {
this.properties= properties;
}
}
```

Κλάση CredentialsAdapter

```
package com.cryptopassword.helpers;

import android.content.Context;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;

import java.util.ArrayList;
import java.util.List;

import androidx.annotation.NonNull;
public class CredentialsAdapterextends BaseAdapter {

private Context context;
    private List<Credential>credentialsList;

    public CredentialsAdapter(@NonNull Context context,  ArrayList<Credential>
list) {
this.context= context;
credentialsList= list;
}

@NonNull
    @Override
public View getView(int position, View convertView, ViewGroup parent) {
        Credential cr = credentialsList.get(position);
View v = new CredentialsAdapterView(this.context,cr);
        return v;
}

@Override
```

```
public int getCount() {
if(credentialsList.size()<=0){
return 1;
}
return credentialsList.size();
}

@Override
public Object getItem(int i) {
return credentialsList.get(i);
}

@Override
public long getItemId(int i) {
return i;
}
}
```

Κλάση CredentialsAdapterView

```
package com.cryptopassword.helpers;

import android.content.Context;
import android.graphics.Typeface;
import android.view.Gravity;
import android.widget.LinearLayout;
import android.widget.RelativeLayout;
import android.widget.TextView;

public class CredentialsAdapterViewextends RelativeLayout {
public CredentialsAdapterView(Context context,Credential credential){
super(context);
LinearLayout.LayoutParams params;
params = new LinearLayout.LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT);
params.setMargins(40,40,40,40);
LinearLayoutpanelLL = new LinearLayout(context);
panelLL.setOrientation(LinearLayout.VERTICAL);
panelLL.setGravity(Gravity.CENTER);

TextView service = new TextView( context );
service.setTextSize(20);
service.setTypeface(Typeface.DEFAULT, Typeface.BOLD);
service.setText("Service:"+credential.getService());
service.setPadding(0,5,0,5);
panelLL.addView(service);

TextView username = new TextView( context );
username.setTextSize(20);
username.setText("Username:"+credential.getUsename());
username.setPadding(0,5,0,5);
panelLL.addView(username);

TextView password = new TextView( context );
password.setTextSize(20);
password.setText("Password:"+credential.getPassword());
password.setPadding(0,5,0,5);
```

```
panelLL.addView(password);

        for(int i=0;i<credential.getProperties().size();i++){
TextView property = new TextView( context );
property.setTextSize(20);
property.setText(credential.getProperties().get(i));
property.setId(i+1);
property.setPadding(0,5,0,5);
panelLL.addView(property);
}

addView(panelLL, params);
}
}
```

Κλάση Cryptography

```
package com.cryptopassword.helpers;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.RandomAccessFile;
import java.nio.charset.StandardCharsets;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.Base64;
import java.util.Random;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

/**
 * Η κλάση παρέχει βασικές λειτουργίες κρυπτογράφησης. Περιλαμβάνει δημιουργία
κλειδιού, κρυπτογράφηση πίνακα και αρχείου, καταστροφή
 * αρχείου στο δίσκο, κλπ. Οι λειτουργίες της κρυπτογράφησης και της
αποκρυπτογράφησης αρχείου γίνονται με μεθόδους που εκτελούνται στο
 * σύνολο του αρχείου συνολικά (αν χωρά στη μνήμη ολόκληρο) και με μεθόδους που
εκτελούν τη λειτουργία block με block.
 */
publicclassCryptography
{
/**
     * Μετατρέπει ένα UTF-8 string σε 128 bits (16 bytes) κλειδί για χρήση σε
αλγόριθμους συμμετρικής κρυπτογράφησης. Το string μετατρέπεται
     * σε πίνακα bytes και στην συνέχεια εφαρμόζεται πάνω του ο hash αλγόριθμος
SHA-1. Από τον πίνακα που παράγεται επιστρέφονται τα πρώτα
     * 16 bytes.
     * @param PasswordTo string (συνθηματικό) που θα μετατραπεί σε κλειδί.
```

```
     * @return Το 128 bit κλειδί.
     * @throws DDException Λάθος στις παραμέτρους των βιβλιοθηκών. Δεν πρέπει να
συμβεί ποτέ.
     */
publicstaticbyte[] Password2Key (StringPassword) throwsDDException
    {
try{
byte[] Key = Password.getBytes (StandardCharsets.UTF_8);
MessageDigestsha = MessageDigest.getInstance("SHA-1");
Key = sha.digest (Key);
Key = Arrays.copyOf(Key, 16);
returnKey;
}
catch(NoSuchAlgorithmExceptionex){
thrownewDDException ("Password2Key", DDException.GenImpossibleException, Password,
ex);
}
    }

/**
     * Κρυπτογραφεί / Αποκρυπτογραφεί ένα πίνακα με bytes εφαρμόζοντας τον
αλγόριθμο XOR με ένα UTF-8 password. Το συνθηματικό μετατρέπεται
     * σε πίνακα από bytes. Δεν μπορεί να διαπιστωθεί αν το κλειδί είναι σωστό και
αν η αποκρυπτογράφηση πραγματοποιήθηκε κανονικά.
     * @param Mat Ο πίνακας προς κρυπτογράφηση / αποκρυπτογράφηση.
     * @param Password Το συνθηματικό κρυπτογράφησης / αποκρυπτογράφησης.
     * @return Ο πίνακας με το αποτέλεσμα της λειτουργίας.
     * @throws DDException Λάθος στις παραμέτρους των βιβλιοθηκών. Δεν πρέπει να
συμβεί ποτέ.
     */
publicstaticbyte[] XORCrypt(byte[] Mat, StringPassword) throwsDDException
    {
byte[] Key = null;
Key = Password.getBytes (StandardCharsets.UTF_8);
intj = 0;
byte[] Res = newbyte[Mat.length];
        for (inti = 0; i <Mat.length; i++){
Res[i] = (byte) (Mat[i] ^ Key[j++]);
if(j == Key.length)
                j = 0;
}
returnRes;
}

/**
     * Κρυπτογραφεί ένα πίνακα από bytes με τον αλγόριθμο AES και χρήση των
βιβλιοθηκών της Java. Οι παράμετροι της κλάσης "Cipher" είναι:
     * "AES/CBC/PKCS5Padding". Ο πίνακας IvSpec για το CBC είναι στατικά ορισμένος
εσωτερικά και πρέπει να θεωρείται... γνωστός. Το
     * συνθηματικό μετατρέπεται σε κλειδί με την χρήση της {@link
#Password2Key(java.lang.String)}. Κάθε κλήση της συνάρτησης θεωρείται νέα
     * κρυπτογράφηση και όχι συνέχεια της προηγούμενης. Λόγω του PKCS5 padding ο
πίνακας που δημιουργείται έχει μέγεθος ίσο με το επόμενο
     * πολλαπλάσιο του 16 από το μέγεθος του πίνακα προς κρυπτογράφηση.
     * @param PlainMat Ο πίνακας που θα κρυπτογραφηθεί.
     * @param PassTo συνθηματικό της κρυπτογράφησης.
     * @return Ο κρυπτογραφημένος πίνακας.
     * @throws DDException Λάθος στις παραμέτρους των βιβλιοθηκών. Δεν πρέπει να
```

```
συμβεί ποτέ.
     */
publicstaticStringAESEncrypt(StringPlainMat, StringPass) throwsDDException
    {
byte[] Iv = {45, 26, 11, 120, 32, 0, 1, 12, 1, 65, 9, 10, 34, 45, 2, 2};
IvParameterSpecIvSpec = newIvParameterSpec (Iv);
byte[] Key;
Cipher c;
try
{
Key = Password2Key (Pass);
c = Cipher.getInstance("AES/CBC/PKCS5Padding");
SecretKeySpec k = newSecretKeySpec (Key, "AES");
c.init (Cipher.ENCRYPT_MODE, k, IvSpec);
returnBase64.getEncoder().encodeToString(c.doFinal(PlainMat.getBytes(StandardChars
ets.UTF_8)));
}
catch(NoSuchPaddingException | NoSuchAlgorithmException | InvalidKeyException |
InvalidAlgorithmParameterException | IllegalBlockSizeException |
BadPaddingExceptionex)
        {
thrownewDDException ("AESEncrypt", DDException.GenImpossibleException, Pass, ex);
}
    }

/**
     * Αποκρυπτογραφεί ένα πίνακα από bytes με τον αλγόριθμο AES και χρήση των
βιβλιοθηκών της Java. Οι παράμετροι της κλάσης "Cipher"
     * είναι: "AES/CBC/PKCS5Padding". Ο πίνακας IvSpec για το CBC είναι στατικά
ορισμένος εσωτερικά και πρέπει να θεωρείται... γνωστός. Το
     * συνθηματικό μετατρέπεται σε κλειδί με την χρήση της {@link
#Password2Key(java.lang.String)}. Κάθε κλήση της συνάρτησης θεωρείται νέα
     * αποκρυπτογράφηση και όχι συνέχεια της προηγούμενης. Λόγω του PKCS5 padding
η μέθοδος μπορεί να διαπιστώσει αν η αποκρυπτογράφηση
     * πραγματοποιήθηκε σωστά.
     * @param CipherMatΟ πίνακας που θα αποκρυπτογραφηθεί.
     * @param PassΤο συνθηματικό της αποκρυπτογράφησης.
     * @return Ο πίνακας με τα αρχικά δεδομένα.
     * @throws DDException Σε περίπτωση λάθους συνθηματικού ή κρυπτογραφημένου
πίνακα και σε περίπτωση λάθους παραμέτρων στις βιβλιοθήκες
     * (δεν πρέπει να συμβεί ποτέ)
     */
publicstaticStringAESDecrypt(StringCipherMat, StringPass) throwsDDException
    {
byte[] Iv = {45, 26, 11, 120, 32, 0, 1, 12, 1, 65, 9, 10, 34, 45, 2, 2};
IvParameterSpecIvSpec = newIvParameterSpec (Iv);
byte[] Key;
Cipher c;
try
{
Key = Password2Key(Pass);
c = Cipher.getInstance("AES/CBC/PKCS5Padding");
SecretKeySpec k = newSecretKeySpec (Key, "AES");
c.init (Cipher.DECRYPT_MODE, k, IvSpec);
returnnewString(c.doFinal(Base64.getDecoder().decode(CipherMat)));
}
catch(NoSuchPaddingException | NoSuchAlgorithmException | InvalidKeyException
|InvalidAlgorithmParameterExceptionex)
```

```
        {
throwneDDException ("AESDecrypt", DDException.GenImpossibleException, Pass, ex);
} catch(IllegalBlockSizeExceptionex)
        {
throw new DDException ("IllegalBlockSizeException.",
DDException.CryptInvalidPassword, Pass, ex);
} catch(BadPaddingExceptionex)
        {
throw new DDException ("BadPaddingException.", DDException.CryptInvalidPassword,
Pass, ex);
}
    }


/**
     * Σβήνει ένα αρχείο αφού τροποποιήσει αρκετές φορές τα περιεχόμενά του ώστε
να μην μπορούν να ανακτηθούν σε περίπτωση αναίρεσης της
     * διαγραφής με ειδικά εργαλεία. Το μέγεθος του αρχείου πρέπει να είναι τέτοιο
που να επιτρέπει την δημιουργία ενός buffer στη
     * μνήμη ίδιου μεγέθους.
     * @param FnΤο όνομα του αρχείου που θα διαγραφεί.
     * @throws DDException Σε περίπτωση που το αρχείο δεν υπάρχει ή είναι αδύνατη
η τροποποίησή του.
*/
public static void DestroySmallFile(String Fn) throws DDException
    {
try
{
            File F = new File (Fn);
            if (!F.exists ())
throw new DDException ("File not Found", DDException.CryptCannotReadFile, Fn);
RandomAccessFile Rf = new RandomAccessFile (Fn, "rws");
Random Rnd = new Random ();
            byte[] Mat = new byte[(int) Rf.length ()];
            for (int i = 0; i<Mat.length; i++)
                Mat[i] = (byte) 0;
Rf.seek (0);
Rf.write (Mat);
            for (int i = 0; i<Mat.length; i++)
                Mat[i] = (byte) -1;
Rf.seek (0);
Rf.write (Mat);
            for (int i = 0; i<5; i++)
            {
Rnd.nextBytes (Mat);
Rf.seek (0);
Rf.write (Mat);
}
Rf.close ();
F.delete ();
}
catch (IOException e)
        {
throw new DDException ("Cannot Destroy File", DDException.CryptCannotWriteFile,
Fn);
}

    }
```

```
/**
     * Κρυπτογραφεί ένα αρχείο και γράφει το κρυπτόγραμμασε ένα νέο αρχείο. Το
αρχείο που κρυπτογραφείται δεν τροποποιείται. Το αρχείο
     * διαβάζεται τμηματικά και μπορεί να έχει οποιοδήποτε μέγεθος. Η
κρυπτογράφηση του κάθε τμήματος πραγματοποιείται σε δύο στάδια. Πρώτα
     * εφαρμόζεται στα δεδομένα ο αλγόριθμος AES και την συνέχεια ο αλγόριθμος XOR
με την χρήση της {@link #XORCrypt}. Για κάθε ένα από τα
     * στάδια χρησιμοποιείται διαφορετικό συνθηματικό.
     * @param SouΤο όνομα του προς κρυπτογράφηση αρχείου.
     * @param DestΤο όνομα του κρυπτογράμματος.
     * @param Pass1 Το συνθηματικό (κλειδί) για τον AES.
     * @param Pass2 Το συνθηματικό (κλειδί) για τον XOR.
     * @throws DDException Σε περίπτωση που δεν μπορεί να διαβαστεί το πηγαίο
αρχείο ή να γραφεί το κρυπτόγραμμα και σε περίπτωση που
     * δημιουργηθούν εξαιρέσεις στις μεθόδους που καλούνται.
*/
public static void EncryptFile(String Sou, String Dest, String Pass1, String
Pass2) throws DDException
    {
byte[] UnEncrypted;
        byte[] XORed;
        byte[] Encrypted;
        byte[] Iv = {45, 26, 11, 120, 32, 0, 1, 12, 1, 65, 9, 10, 34, 45, 2, 2};
IvParameterSpecIvSpec = new IvParameterSpec (Iv);
        byte[] Key;
Cipher c;
        int Br;
        try
{
            Key = Password2Key (Pass1);
c = Cipher.getInstance("AES/CBC/PKCS5Padding");
SecretKeySpec k = new SecretKeySpec (Key, "AES");
c.init (Cipher.ENCRYPT_MODE, k, IvSpec);
}
catch (NoSuchPaddingException | NoSuchAlgorithmException | InvalidKeyException
|InvalidAlgorithmParameterException e)
        {
throw new DDException ("File Encrypt", DDException.GenImpossibleException, Pass1,
e);
}

UnEncrypted = new byte[524288];

File S = new File (Sou);
        if (!S.canRead ())
throw new DDException ("Cannot Open Unencrypted File",
DDException.CryptCannotReadFile, Sou);
File D = new File (Dest);
        try
{
if (!D.createNewFile ())
throw new DDException ("Error Creating Encrypted File",
DDException.CryptCannotCreateFile, Dest);
}
catch (IOException ex)
```

```
        {
throw new DDException ("Error Creating Encrypted File",
DDException.CryptCannotWriteFile, Dest);
}
try (FileInputStreamFis = new FileInputStream (S);
FileOutputStreamFos = new FileOutputStream (D))
        {
while ((Br = Fis.read (UnEncrypted)) == 524288)
            {
                Encrypted = c.update (UnEncrypted);
XORed = XORCrypt(Encrypted, Pass2);
Fos.write (XORed);
}
if (Br >0)
                Encrypted = c.doFinal (UnEncrypted, 0, Br);
            else
Encrypted = c.doFinal ();
XORed = XORCrypt(Encrypted, Pass2);
Fos.write (XORed);
}
catch (IOException | IllegalBlockSizeException | BadPaddingException e)
        {
throw new DDException ("AESEncrypt", DDException.GenImpossibleException, Pass1,
e);
}
    }


/**
     * Αποκρυπτογραφεί ένα αρχείο το οποίο δημιουργήθηκε με την {@link
#EncryptFile }. Δημιουργεί νέο αρχείο χωρίς να τροποποιεί το αρχικό
     * (κρυπτόγραμμα).
     * @param SouTo αρχείο που θα αποκρυπτογραφηθεί.
     * @param DestTo αποκρυπτογραφημένο αρχείο που θα δημιουργηθεί.
     * @param Pass1 Το συνθηματικό (κλειδί) για τον AES.
     * @param Pass2 Το συνθηματικό για τον XOR.
     * @throws DDException Σε περίπτωση που δεν μπορεί να γίνει ανάγνωση / εγγραφή
των αρχείων και σε περίπτωση που τα συνθηματικά είναι
     * λανθασμένα ή το κρυπτογραφημένο αρχείο είναι κατεστραμένο.
*/
publicstaticvoidDecryptFile(StringSou, StringDest, StringPass1, StringPass2)
throwsDDException
    {
byte[] UnEncrypted;
byte[] XORed;
byte[] Encrypted;
byte[] Iv = {45, 26, 11, 120, 32, 0, 1, 12, 1, 65, 9, 10, 34, 45, 2, 2};
IvParameterSpecIvSpec = newIvParameterSpec (Iv);
byte[] Key;
Cipherc;
intBr;
intBlk;
inti;
try
{
Key = Password2Key(Pass1);
c = Cipher.getInstance("AES/CBC/PKCS5Padding");
SecretKeySpeck = newSecretKeySpec (Key, "AES");
```

```
c.init (Cipher.DECRYPT_MODE, k, IvSpec);
}
catch(NoSuchPaddingException | NoSuchAlgorithmException | InvalidKeyException
|InvalidAlgorithmParameterExceptione)
        {
thrownewDDException ("FileDecrypt", DDException.GenImpossibleException, Pass1, e);
}

XORed = newbyte[524288];

FileS = newFile (Sou);
if(!S.canRead ())
thrownewDDException ("CannotOpenEncryptedFile", DDException.CryptCannotReadFile,
Sou);
Blk = (int) ((S.length () - 1) / 524288);
FileD = newFile (Dest);
try
{
if(!D.createNewFile ())
thrownewDDException ("ErrorCreatingUnEncryptedFile",
DDException.CryptCannotCreateFile, Dest);
}
catch(IOExceptionex)
        {
thrownewDDException ("ErrorCreatingUnEncryptedFile",
DDException.CryptCannotWriteFile, Dest);
}
try(FileInputStreamFis = newFileInputStream (S);
FileOutputStreamFos = newFileOutputStream (D))
        {
for (i = 1; i <= Blk; i++)
            {
Fis.read (XORed);
Encrypted = XORCrypt(XORed, Pass2);
UnEncrypted = c.update (Encrypted);
Fos.write (UnEncrypted);
}
Br = Fis.read (XORed);
Encrypted = XORCrypt(XORed, Pass2);
UnEncrypted = c.doFinal (Encrypted, 0, Br);
Fos.write (UnEncrypted);
}
catch(IOException e)
        {
thrownewDDException ("DecryptFile IO Error", DDException.CryptCannotWriteFile,
Dest, e);
}
catch(IllegalBlockSizeException | BadPaddingException e)
        {
thrownewDDException ("InvalidPasswordorData", DDException.CryptInvalidPassword,
Dest, e);
}
    }

/**
     * Σβήνει ένα αρχείο αφού αντικαταστήσει τα περιεχόμενα του αρκετές φορές με
διάφορες τιμές ώστε να μην μπορεί να διαβαστεί αν ανακτηθεί
     * με ειδικά εργαλεία.Το μέγεθος του αρχείου μπορεί να μεγαλώσει λίγο (σε
```

```
πολλαπλάσιο του 512K) πριν διαγραφεί οριστικά. Οι εγγραφές που
     * γίνονται σε κάθε byte είναι μόλις 7 προκειμένου να μην δημιουργούνται
προβλήματα σε SSD δίσκους και ταμπλέτες.
     * @param FnΤο όνομα του αρχείου που θα καταστραφεί.
     * @throws DDException Σε περίπτωση που το αρχείο δεν υπάρχει ή δεν μπορεί να
γίνει τροποποίηση των περιεχομένων του.
*/
public static void DestroyFile(String Fn) throws DDException
    {
final int BlockS = 512 * 1024;
        try
{
            File F = new File (Fn);
            if (!F.exists ())
throw new DDException ("File not Found", DDException.CryptCannotReadFile, Fn);
RandomAccessFile Rf = new RandomAccessFile (Fn, "rws");
Random Rnd = new Random ();
            byte[] Mat = new byte[BlockS];
            long Fs = Rf.length ();
            int Blks = (int) (Fs / BlockS + 1);
            for (int B = 0; B <Blks; B++)
            {
for (int i = 0; i<BlockS; i++)
                    Mat[i] = (byte) 0;
Rf.seek (B * BlockS);
Rf.write (Mat);
                for (int i = 0; i<BlockS; i++)
                    Mat[i] = (byte) -1;
Rf.seek (B * BlockS);
Rf.write (Mat);
                for (int i = 0; i<5; i++)
                {
Rnd.nextBytes (Mat);
Rf.seek (B * BlockS);
Rf.write (Mat);
}
            }
Rf.close ();
F.delete ();
}
catch (IOException e)
        {
throw new DDException ("Cannot Destroy File", DDException.CryptCannotWriteFile,
Fn);
}
    }

public static class IOUtil
    {

public static byte[] readFile(File file) throws IOException
        {
RandomAccessFile f = new RandomAccessFile (file, "r");
            try
{
int length = (int) f.length ();
                byte[] data = new byte[length];
f.readFully (data);
```

```
                return data;
}
finally
{
f.close ();
}
        }

public static void writeFile(File file, byte[] data) throws IOException
        {
RandomAccessFile f = new RandomAccessFile (file, "rw");
            try
{
f.write (data);
}
finally
{
f.close ();
}
        }
    }
}
```

Κλάση DDException

```
packagecom.cryptopassword.helpers;

/**
 * Η κλάση DDException είναι η γενική κλάση για τις εξαιρέσεις όλων των έργων. Η
κλάση κληρονομεί την κλάση {@link DDException}
 * και ορίζει νέα attributes. Θα μπορεί να κληρονομηθεί από άλλες κλάσεις αν
χρειάζεται (π.χ. για να υλοποιηθούν κάποια interfaces με
 * κωδικούς λαθών), αλλά αυτό δεν θα είναι απαραίτητο. Υλοποιείτο interface {@link
ExceptionConsts}.

public class DDExceptionextends Exception implements ExceptionConsts
{
/**
     * Οκωδικόςλάθοςτου Exception. Πρέπει να πάρει τιμές από αυτές που ορίζονται
στα σχετικά interfaces.
     */
privateintErrorCode;

/**
     * Δεδομένα σχετικά με το πρόβλημα που δημιουργήθηκε. Μπορεί να περιέχει και
προγραμματιστικές πληροφορίες, π.χ. τις τιμές κάποιων
     * μεταβλητών κλπ.
     */
privateStringData;

/**
     * Το Exception του συστήματος που προκάλεσε τη δημιουργία του συγκεκριμένου
Exception (αν υπάρχει τέτοιο).
     */
privateExceptionSysException;

/**
     * Ο κατασκευαστής του Exception και ο μοναδικός τρόπος να οριστούν τιμές στα
```

```
πεδία του (δεν υπάρχουν setters).
     * @param messageΤο μήνυμα για τον κατασκευαστή της κλάσης {@link DDException}
που κληρονομείται.
     * @param errCodeΟ κωδικός σφάλματος. Οι κωδικοί πρέπει να παίρνουν τιμές από
το interface {@link ExceptionConsts} ή από αντίστοιχο.
     * @param dataΤα δεδομένα που αφορούν το σφάλμα σε String.
     * @param sysExcΑν το exception δημιουργείται για να ενημερώσει για κάποιο
άλλο exception που δημιουργήθηκε από το σύστημα.
     * (π.χ. ένα SQLException κατά την εκτέλεση εντολών σχετικών με τη βάση
δεδομένων) εδώ καταχωρείται το αρχικό exception. Αν δεν υπάρχει
     * τέτοια περίπτωση η παράμετρος έχει τιμή null.
     */
publicDDException(Stringmessage, interrCode, Stringdata, ExceptionsysExc)
    {
super(message);
ErrorCode= errCode;
Data= data;
SysException= sysExc;
}

/**
     * Βοηθητικός κατασκευαστής για την δημιουργία αντικειμένου χωρίς την
παράμετρο SysException. Καλεί τον βασικό κατασκευαστή περνώντας
     * του null στην σχετική παράμετρο.
     * @param messageΤο μήνυμα για τον κατασκευαστή της κλάσης Exception που
κληρονομείται.
     * @param errCodeΟ κωδικός σφάλματος. Οι κωδικοί πρέπει να πέρνουν τιμές από
τοinterface {@link ExceptionConsts} ή από αντίστοιχο.
     * @param dataΤα δεδομένα που αφορούν το σφάλμα σε String.
*/
public DDException(String message, int errCode, String data)
    {
this(message, errCode, data, (Exception) null);
}

/**
     * Επιστρέφει τον κωδικό λάθους που έχει καταγραφεί στο exception.
* @return Ο κωδικός λάθους.
     */
publicintgetErrorCode()
    {
returnErrorCode;
}

/**
     * Επιστρέφει τα δεδομένα που αφορούν το σφάλμα που δημιουργήθηκε.
     * @return Η αναλυτική περιγραφή.
     */
publicStringgetData()
    {
returnData;
}

/**
     * Επιστρέφει το Exception Συστήματος που δημιούργησε το τρέχων exception ή
null αν δεν έχει οριστεί τέτοιο.
* @return Το System Exception.
     */
```

```
public Exception getSysException()
    {
return SysException;
}

}
```

Κλάση DeleteWalletDialog

```
package com.cryptopassword.helpers;

import android.app.Activity;
import android.app.Dialog;
import android.content.Context;
import android.content.ContextWrapper;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.RelativeLayout;
import android.widget.Toast;

import com.cryptopasswordentral.R;

import java.io.File;
import java.util.ArrayList;

import static com.cryptopassword.central.ExitActivity.exitApplication;
import static com.cryptopassword.helpers.FileHandler.getAbsoluteFile;
import static com.cryptopassword.helpers.FileHandler.getAllFileNames;
import static com.cryptopassword.helpers.FileHandler.readDataFile;

public class DeleteWalletDialogextends Dialog implements View.OnClickListener {

private Activity activity;
    private Button delete, back;
    private String filename;
    private ArrayList<String>decryptedString;
    private int openedFileId;

    public DeleteWalletDialog(Activity a, String filename) {
super(a);
this.activity= a;
this.filename=filename;
}

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
requestWindowFeature(Window.FEATURE_NO_TITLE);
setContentView(R.layout.del_wallet_dialog);
delete = findViewById(R.id.delete);
```

```
back = findViewById(R.id.back);
delete.setOnClickListener(this);
back.setOnClickListener(this);

RadioGroup group = findViewById(R.id.wallet_group);
RadioButton button;
        int id=0;
        for(String s:getAllFileNames(getContext())) {
if(s.contains(".txt") && !s.equals("pin.txt")) {
                id++;
button = new RadioButton(getContext());
button.setText(s);
button.setTextSize(20);
button.setId(id);
RelativeLayout.LayoutParams p;
p = new RelativeLayout.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
p.addRule(RelativeLayout.BELOW, R.id.pinSelWallet);
button.setLayoutParams(p);
                if (s.equals(filename)) {
button.setChecked(true);
button.setTextColor(Color.GREEN);
openedFileId= id;
}
group.addView(button);
}
        }
    }

@Override
public void onClick(View v) {
switch (v.getId()) {
case R.id.delete:
EditText password = findViewById(R.id.pinSelWallet);
                if(password.getText().toString().equals("")){
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.passwordEmpty),
Toast.LENGTH_SHORT).show();
                    return;
}
RadioGroup group = findViewById(R.id.wallet_group);
View radioButton = group.findViewById(group.getCheckedRadioButtonId());
RadioButton r = (RadioButton) group.getChildAt(group.indexOfChild(radioButton));
decryptedString=
readDataFile(password.getText().toString(),r.getText().toString(),getContext());
                if(decryptedString!= null){
                    File applicationPath = getAbsoluteFile("", getContext());
File file = new File(applicationPath, "/CryptoPassword/"+r.getText().toString());
                    if(file.delete()){
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.walletDeleted)+":"+r.getText().toSt
ring(), Toast.LENGTH_SHORT).show();
                        if(getAllFileNames(getContext()).size()==0){
                            File directory = new File(applicationPath,
"/CryptoPassword/");
directory.delete();
}
if(((GlobalVariables) activity.getApplication()).getFileName() != null &&
```

```
((GlobalVariables)
activity.getApplication()).getFileName().equals(r.getText().toString())){
exitApplication(unwrap(getContext()));
}
                    }else{
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.walletNotDeleted)+":"+r.getText().t
oString(), Toast.LENGTH_SHORT).show();
}
                }else{
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.falsePassword),
Toast.LENGTH_SHORT).show();
}
case R.id.back:
                dismiss();
                break;
}
    }

private static Activity unwrap(Context context) {
while (!(context instanceofActivity) && context instanceofContextWrapper) {
            context = ((ContextWrapper) context).getBaseContext();
}
return (Activity) context;
}

}
```

Κλάση EditCredentialDialog

```
package com.cryptopassword.helpers;

import android.app.Activity;
import android.app.Dialog;
import android.content.Context;
import android.content.ContextWrapper;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RelativeLayout;
import android.widget.Toast;

import com.cryptopassword.central.ShowCredentials;
import com.cryptopasswordentral.R;

import java.io.IOException;
import java.util.ArrayList;

public class EditCredentialDialogextends Dialog implements
android.view.View.OnClickListener {

private Activity activity;
```

```
    private Dialog dialog;
    private Button save, back;
    private EditTextservice,username,password;
    private Credential credential;
    private DialogListenerlistener;
    private Boolean giaAntigrafi;

    public EditCredentialDialog(Activity a, Credential cr, Boolean giaAntigrafi) {
super(a);
this.activity= a;
this.credential= cr;
this.giaAntigrafi=giaAntigrafi;
}

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
requestWindowFeature(Window.FEATURE_NO_TITLE);
setContentView(R.layout.edit_cred_dialog);
save = findViewById(R.id.save);
back = findViewById(R.id.back);
save.setOnClickListener(this);
back.setOnClickListener(this);
service =  findViewById(R.id.service);
username =  findViewById(R.id.username);
password =  findViewById(R.id.password);

service.setText(credential.getService());
username.setText(credential.getUsename());
password.setText(credential.getPassword());

RelativeLayoutll = findViewById(R.id.sel_wallet_dialog_layout);
        int createdTextViews = 0;
        for (int i=0;credential.getProperties()!=null
&&i<credential.getProperties().size();i++){
RelativeLayout.LayoutParams p;
EditText et = new EditText(getContext());
p = new RelativeLayout.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
            if (createdTextViews == 0) {
p.addRule(RelativeLayout.BELOW, R.id.password);
} else {
p.addRule(RelativeLayout.BELOW, createdTextViews);
}
p.setMargins(0, 40, 0, 0);
et.setLayoutParams(p);
et.setTextSize(20);
et.setGravity(Gravity.CENTER);
et.setHint(credential.getProperties().get(i).split(":")[0]);
et.setText(credential.getProperties().get(i).split(":")[1]);
et.setId(createdTextViews + 1);
ll.addView(et);
createdTextViews++;
}
listener = unwrap(getContext());
}

@Override
```

```
public void onClick(View v) {
switch (v.getId()) {
case R.id.save:
EditText et;
ArrayList<String>newDatas = new ArrayList<>();
                for(int i=1;credential.getProperties()!=null
&&i<credential.getProperties().size()+1;i++){
                    et=findViewById(i);
newDatas.add(et.getHint().toString()+":"+et.getText().toString());
}
if(service.getText().toString().isEmpty()||username.getText().toString().isEmpty()
|| password.getText().toString().isEmpty()){
showErrorToast();
                    break;
}
                Credential editedCr = new
Credential(service.getText().toString(),username.getText().toString(),password.get
Text().toString(),newDatas);
                try {
listener.setEditedCredential(editedCr,credential,giaAntigrafi);
} catch (IOException | DDException e) {
e.printStackTrace();
}
break;
            case R.id.back:
                dismiss();
                break;
            default:
break;
}
        dismiss();
}

private static DialogListenerunwrap(Context context) {
while (!(context instanceofDialogListener) && context instanceofContextWrapper) {
            context = ((ContextWrapper) context).getBaseContext();
}
return (DialogListener) context;
}

public interface DialogListener{
void setEditedCredential(Credential
editedCredential,CredentialoriginalCredential,BooleangiaAntigrafi) throws
IOException, DDException;
}

public void showErrorToast() {
Toast.makeText(getContext(),R.string.kena, Toast.LENGTH_SHORT).show();
}
}
```

Κλάση FileHandler

```
package com.cryptopassword.helpers;


import android.content.Context;
```

```
import android.os.Environment;
import android.provider.Settings;
import android.util.Log;

import com.cryptopasswordentral.R;

import org.apache.poi.hssf.usermodel.HSSFCell;
import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.RandomAccessFile;
import java.util.ArrayList;
import java.util.Random;

import static com.cryptopassword.helpers.Cryptography.AESDecrypt;
import static com.cryptopassword.helpers.Cryptography.AESEncrypt;

public class FileHandler {

public static void writeDataFile(Credential
cr,Stringpassword,StringfileName,Context context) throws IOException, DDException
{
        File applicationPath = getAbsoluteFile("", context);
File cryproPath = new File(applicationPath.getAbsolutePath() +
"/CryptoPassword/");
        if (!cryproPath.exists())
if(!cryproPath.mkdir()){
Log.i("----","Den eftiakse ton katalogo");
}
        File file = new File(applicationPath, "/CryptoPassword/"+fileName);
        if (!file.exists()) {
            File dir = new File(applicationPath.getAbsolutePath() +
"/CryptoPassword/");
File file2 = new File(dir, fileName);
FileOutputStreamfos = new FileOutputStream(file2);
String temp = cr.getService() + "\n" + cr.getUsename() + "\n" + cr.getPassword() +
"\n";
            for(int i=0;i<cr.getProperties().size();i=i+2){
                temp = temp + cr.getProperties().get(i) + ":" +
cr.getProperties().get(i+1) + "\n";
}
fos.write(AESEncrypt(temp,password).getBytes());
fos.close();
} else {
FileOutputStreamfos = new FileOutputStream(file,true);
String temp = cr.getService() + "\n" + cr.getUsename() + "\n" + cr.getPassword() +
"\n";
            for(int i=0;i<cr.getProperties().size();i=i+2){
                temp = temp + cr.getProperties().get(i) + ":" +
cr.getProperties().get(i+1) + "\n";
}
            String divider = "--";
fos.write(divider.getBytes());
```

```
fos.write(AESEncrypt(temp,password).getBytes());
fos.close();
}
    }

public static void writePinFile(String pin,Context context) throws IOException,
DDException {
        File applicationPath = getAbsoluteFile("", context);
File cryproPath = new File(applicationPath.getAbsolutePath() +
"/CryptoPassword/");
        if (!cryproPath.exists())
if(!cryproPath.mkdir()){
Log.i("----","Den eftiakse ton katalogo");
}
        File file = new File(applicationPath, "/CryptoPassword/pin.txt");
        if (!file.exists()) {
            File dir = new File(applicationPath.getAbsolutePath() +
"/CryptoPassword/");
File file2 = new File(dir, "pin.txt");
FileOutputStreamfos = new FileOutputStream(file2);
fos.write(AESEncrypt(pin,
Settings.Secure.getString(context.getContentResolver(),Settings.Secure.ANDROID_ID)
).getBytes());
fos.close();
}else{
file.delete();
File dir = new File(applicationPath.getAbsolutePath() + "/CryptoPassword/");
File file2 = new File(dir, "pin.txt");
FileOutputStreamfos = new FileOutputStream(file2);
fos.write(AESEncrypt(pin,
Settings.Secure.getString(context.getContentResolver(),Settings.Secure.ANDROID_ID)
).getBytes());
fos.close();
}
    }

public static ArrayList<String>readPinFile(Context context) {
try{
            File applicationPath = getAbsoluteFile("", context);
ArrayList<String> result = new ArrayList<>();
File file2 = new File(applicationPath, "/CryptoPassword/pin.txt");
FileInputStreamfileInputStream = null;
            byte[] bFile = new byte[(int) file2.length()];
fileInputStream = new FileInputStream(file2);
fileInputStream.read(bFile);
String[] temp = new String(bFile).split("--");
            for(int i =0;i<temp.length;i++){

result.add(AESDecrypt(temp[i],Settings.Secure.getString(context.getContentResolver
(),Settings.Secure.ANDROID_ID)));
}
if(result.get(0).equals(""))
return null;
            return result;
}catch (IOException | DDException e) {
e.printStackTrace();
            return null;
}
```

```
    }

public static void writeXLSFile(String
fileName,ArrayList<Credential>crList,Context context) throws IOException {
        File applicationPath = getAbsoluteFile("", context);
HSSFWorkbookwb = new HSSFWorkbook();
HSSFSheet sheet = wb.createSheet(fileName.substring(0,fileName.length()-4)) ;
//iterating r number of rows
int rowNum=0;
        int columnNum=0;
        for(Credential cr:crList){
HSSFRow row = sheet.createRow(rowNum);
HSSFCell cell = row.createCell(columnNum);
            if(rowNum==0){
cell.setCellValue(context.getString(R.string.service));
columnNum++;
cell = row.createCell(columnNum);
cell.setCellValue(context.getString(R.string.username));
columnNum++;
cell = row.createCell(columnNum);
cell.setCellValue(context.getString(R.string.password));
                for(int i = 0;i<cr.getProperties().size();i++) {
columnNum++;
cell = row.createCell(columnNum);
cell.setCellValue(cr.getProperties().get(i).split(":")[0]);
}
rowNum++;
row = sheet.createRow(rowNum);
}
columnNum=0;
cell = row.createCell(columnNum);
cell.setCellValue(cr.getService());
columnNum++;
cell = row.createCell(columnNum);
cell.setCellValue(cr.getUsename());
columnNum++;
cell = row.createCell(columnNum);
cell.setCellValue(cr.getPassword());
            for(int i = 0;i<cr.getProperties().size();i++) {
columnNum++;
cell = row.createCell(columnNum);
cell.setCellValue(cr.getProperties().get(i).split(":")[1]);
}
columnNum=0;
rowNum++;
}
FileOutputStreamfileOut = new FileOutputStream(new File(applicationPath,
"/CryptoPassword/"+fileName));
wb.write(fileOut);
fileOut.close();
fileOut.flush();
}

public static void
writeAgainAllCredentials(ArrayList<Credential>credList,Stringpassword,StringfileNa
me,Context context) throws IOException, DDException {
        File applicationPath = getAbsoluteFile("", context);
File cryproPath = new File(applicationPath.getAbsolutePath() +
```

```
"/CryptoPassword/");
        if (!cryproPath.exists())
if(!cryproPath.mkdir()){
Log.i("----","Error on creating new dir");
}
        File file = new File(applicationPath, "/CryptoPassword/"+fileName);
file.delete();
File file2 = new File(applicationPath, "/CryptoPassword/"+fileName);
FileOutputStreamfos = new FileOutputStream(file2);
        int x=0;
        for(Credential cr:credList) {
            String temp = cr.getService() + "\n" + cr.getUsename() + "\n" +
cr.getPassword() + "\n";
            for (int i = 0; i<cr.getProperties().size(); i++) {
                temp = temp + cr.getProperties().get(i) + "\n";
}
if(x!=0){
                String divider = "--";
fos.write(divider.getBytes());
}
fos.write(AESEncrypt(temp, password).getBytes());
x++;
}
fos.close();

}

public static ArrayList<String>readDataFile(String password,Stringfilename,Context
context) {
try{
            File applicationPath = getAbsoluteFile("", context);
ArrayList<String> result = new ArrayList<>();
File file2 = new File(applicationPath, "/CryptoPassword/"+filename);
FileInputStreamfileInputStream = null;
            byte[] bFile = new byte[(int) file2.length()];
fileInputStream = new FileInputStream(file2);
fileInputStream.read(bFile);
String[] temp = new String(bFile).split("--"); //evala to iso giatiexana bytes kai
eskage to decrypt
for(int i =0;i<temp.length;i++){
result.add(new String(AESDecrypt(temp[i],password)));
}
if(result.get(0).equals(""))
return null;
            return result;
}catch (IOException | DDException e) {
e.printStackTrace();
            return null;
}
    }


public static ArrayList<String>decryptString(String
password,StringencryptedString) {
try{
ArrayList<String> result = new ArrayList<>();
String[] temp = encryptedString.split("--");
for(int i =0;i<temp.length;i++){
```

```
result.add(new String(AESDecrypt(temp[i],password)));
}
if(result.get(0).equals(""))
return null;
            return result;
}catch (DDException e) {
e.printStackTrace();
            return null;
}
    }

public static ArrayList<String>getAllFileNames(Context context){
ArrayList<String> result = new ArrayList<>();
File applicationPath = getAbsoluteFile("", context);        //get all filenames
from CryptoPassword directory
File folder = new File(applicationPath.getAbsolutePath() + "/CryptoPassword/");
        if(folder.exists()) {
            File[] listOfFiles = folder.listFiles();
            for (File f : listOfFiles) {
if (f.isFile()) {
result.add(f.getName());
}
            }
        }
return result;
}

public static File getAbsoluteFile(String relativePath, Context context) {
if (Environment.MEDIA_MOUNTED.equals(Environment.getExternalStorageState())) {
return new File(context.getExternalFilesDir(null), relativePath);
} else {
return new File(context.getFilesDir(), relativePath);
}
    }


public static void destroyFile(String Fn,Context context) throws DDException
    {
try
{
            File applicationPath = getAbsoluteFile("", context);
File F = new File(applicationPath, "/CryptoPassword/"+Fn);
            if (!F.exists ())
throw new DDException ("File not Found", DDException.CryptCannotReadFile, Fn);
RandomAccessFile Rf = new RandomAccessFile (F, "rws");
Random Rnd = new Random ();
            byte[] Mat = new byte[(int) Rf.length ()];
            for (int i = 0; i<Mat.length; i++)
                Mat[i] = (byte) 0;
Rf.seek (0);
Rf.write (Mat);
            for (int i = 0; i<Mat.length; i++)
                Mat[i] = (byte) -1;
Rf.seek (0);
Rf.write (Mat);
            for (int i = 0; i<5; i++)
            {
Rnd.nextBytes (Mat);
```

```
Rf.seek (0);
Rf.write (Mat);
}
Rf.close ();
F.delete ();
}
catch (IOException e)
        {
throw new DDException ("Cannot Destroy File", DDException.CryptCannotWriteFile,
Fn);
}

    }
}
```

Κλάση GlobalVariables

```
package com.cryptopassword.helpers;

import android.app.Application;

import java.util.ArrayList;

public class GlobalVariablesextends Application {

private String password,fileName;
    private Boolean firstLogIn,changePassForLogIn;
    private ArrayList<String>newFields;

    public ArrayList<String>getNewFields() {
return newFields;
}

public void setNewFields(ArrayList<String>newFields) {
this.newFields= newFields;
}

public Boolean getFirstLogIn() {
return firstLogIn;
}

public void setFirstLogIn(Boolean firstLogIn) {
this.firstLogIn= firstLogIn;
}

public String getPassword() {
return password;
}

public void setPassword(String password) {
this.password= password;
}

public String getFileName() {
return fileName;
}
```

```
public void setFileName(String fileName) {
this.fileName= fileName;
}

public Boolean getChangePassForLogIn() {return changePassForLogIn;}

public void setChangePassForLogIn(Boolean changePassForLogIn)
{this.changePassForLogIn= changePassForLogIn;}
}
```

Κλάση RenameWalletDialog

```
package com.cryptopassword.helpers;

import android.app.Activity;
import android.app.Dialog;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.RelativeLayout;
import android.widget.Toast;

import com.cryptopasswordentral.R;

import java.io.File;
import java.util.ArrayList;

import static com.cryptopassword.helpers.FileHandler.getAbsoluteFile;
import static com.cryptopassword.helpers.FileHandler.getAllFileNames;
import static com.cryptopassword.helpers.FileHandler.readDataFile;

public class RenameWalletDialogextends Dialog implements View.OnClickListener {

private Activity activity;
    private Button rename, back;
    private String filename;
    private ArrayList<String>decryptedString;
    private int openedFileId;

    public RenameWalletDialog(Activity a, String filename) {
super(a);
this.activity= a;
this.filename=filename;
}

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
requestWindowFeature(Window.FEATURE_NO_TITLE);
setContentView(R.layout.ren_wallet_dialog);
rename = findViewById(R.id.rename);
```

```
back = findViewById(R.id.back);
rename.setOnClickListener(this);
back.setOnClickListener(this);

RadioGroup group = findViewById(R.id.wallet_group);
RadioButton button;
        int id=0;
        for(String s:getAllFileNames(getContext())) {
if(s.contains(".txt") && !s.equals("pin.txt")) {
                id++;
button = new RadioButton(getContext());
button.setText(s);
button.setTextSize(20);
button.setId(id);
RelativeLayout.LayoutParams p;
p = new RelativeLayout.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
p.addRule(RelativeLayout.BELOW, R.id.pinSelWallet);
button.setLayoutParams(p);
                if (s.equals(filename)) {
button.setChecked(true);
button.setTextColor(Color.GREEN);
openedFileId= id;
}
group.addView(button);
}
        }
    }

@Override
public void onClick(View v) {
switch (v.getId()) {
case R.id.rename:
EditText password = findViewById(R.id.pinSelWallet);
                if(password.getText().toString().equals("")){
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.passwordEmpty),
Toast.LENGTH_SHORT).show();
                    return;
}
EditTextnameNewWallet = findViewById(R.id.nameNewWallet);
                if(!nameNewWallet.getText().toString().matches("[0-9a-zA-z]+$")){
Toast.makeText(getContext(), R.string.fileNameRightName,Toast.LENGTH_LONG).show();
                    return;
}
RadioGroup group = findViewById(R.id.wallet_group);
View radioButton = group.findViewById(group.getCheckedRadioButtonId());
RadioButton r = (RadioButton) group.getChildAt(group.indexOfChild(radioButton));
decryptedString=
readDataFile(password.getText().toString(),r.getText().toString(),getContext());
                if(decryptedString!= null){
                    File applicationPath = getAbsoluteFile("", getContext());
File file = new File(applicationPath, "/CryptoPassword/"+r.getText().toString());
String fineNameWithExtension = nameNewWallet.getText() + ".txt";
File toRenameFile = new File(applicationPath,
"/CryptoPassword/"+fineNameWithExtension);
                    if(file.renameTo(toRenameFile)){
Toast.makeText(getContext(),
```

```
getContext().getResources().getString(R.string.walletRenamed)+":"+r.getText().toSt
ring(), Toast.LENGTH_SHORT).show();
                        if(((GlobalVariables)
activity.getApplication()).getFileName()!=null && ((GlobalVariables)
activity.getApplication()).getFileName().equals(r.getText().toString())){
                            ((GlobalVariables)
activity.getApplication()).setFileName(fineNameWithExtension);
}
                    }else{
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.walletNotRenamed)+":"+r.getText().t
oString(), Toast.LENGTH_SHORT).show();
}
                }else{
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.falsePassword),
Toast.LENGTH_SHORT).show();
}
case R.id.back:
                dismiss();
                break;
}
    }
}
```

Κλάση SelectWalletDialog

```
package com.cryptopassword.helpers;

import android.app.Activity;
import android.app.Dialog;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.RelativeLayout;
import android.widget.Toast;

import com.cryptopasswordentral.R;

import java.io.File;
import java.util.ArrayList;

import static com.cryptopassword.helpers.FileHandler.getAbsoluteFile;
import static com.cryptopassword.helpers.FileHandler.getAllFileNames;
import static com.cryptopassword.helpers.FileHandler.readDataFile;

public class RenameWalletDialogextends Dialog implements View.OnClickListener {

private Activity activity;
    private Button rename, back;
    private String filename;
```

```
    private ArrayList<String>decryptedString;
    private int openedFileId;

    public RenameWalletDialog(Activity a, String filename) {
super(a);
this.activity= a;
this.filename=filename;
}

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
requestWindowFeature(Window.FEATURE_NO_TITLE);
setContentView(R.layout.ren_wallet_dialog);
rename = findViewById(R.id.rename);
back = findViewById(R.id.back);
rename.setOnClickListener(this);
back.setOnClickListener(this);

RadioGroup group = findViewById(R.id.wallet_group);
RadioButton button;
        int id=0;
        for(String s:getAllFileNames(getContext())) {
if(s.contains(".txt") && !s.equals("pin.txt")) {
                id++;
button = new RadioButton(getContext());
button.setText(s);
button.setTextSize(20);
button.setId(id);
RelativeLayout.LayoutParams p;
p = new RelativeLayout.LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
p.addRule(RelativeLayout.BELOW, R.id.pinSelWallet);
button.setLayoutParams(p);
                if (s.equals(filename)) {
button.setChecked(true);
button.setTextColor(Color.GREEN);
openedFileId= id;
}
group.addView(button);
}
        }
    }

@Override
public void onClick(View v) {
switch (v.getId()) {
case R.id.rename:
EditText password = findViewById(R.id.pinSelWallet);
                if(password.getText().toString().equals("")){
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.passwordEmpty),
Toast.LENGTH_SHORT).show();
                    return;
}
EditTextnameNewWallet = findViewById(R.id.nameNewWallet);
                if(!nameNewWallet.getText().toString().matches("[0-9a-zA-z]+$")){
Toast.makeText(getContext(), R.string.fileNameRightName,Toast.LENGTH_LONG).show();
```

```
                    return;
}
RadioGroup group = findViewById(R.id.wallet_group);
View radioButton = group.findViewById(group.getCheckedRadioButtonId());
RadioButton r = (RadioButton) group.getChildAt(group.indexOfChild(radioButton));
decryptedString=
readDataFile(password.getText().toString(),r.getText().toString(),getContext());
                if(decryptedString!= null){
                    File applicationPath = getAbsoluteFile("", getContext());
File file = new File(applicationPath, "/CryptoPassword/"+r.getText().toString());
String fineNameWithExtension = nameNewWallet.getText() + ".txt";
File toRenameFile = new File(applicationPath,
"/CryptoPassword/"+fineNameWithExtension);
                    if(file.renameTo(toRenameFile)){
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.walletRenamed)+":"+r.getText().toSt
ring(), Toast.LENGTH_SHORT).show();
                        if(((GlobalVariables)
activity.getApplication()).getFileName()!=null && ((GlobalVariables)
activity.getApplication()).getFileName().equals(r.getText().toString())){
                            ((GlobalVariables)
activity.getApplication()).setFileName(fineNameWithExtension);
}
                    }else{
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.walletNotRenamed)+":"+r.getText().t
oString(), Toast.LENGTH_SHORT).show();
}
                }else{
Toast.makeText(getContext(),
getContext().getResources().getString(R.string.falsePassword),
Toast.LENGTH_SHORT).show();
}
case R.id.back:
                dismiss();
                break;
}
    }
}
```

ΚΕΦΑΛΑΙΟ 6ο Βιβλιογραφία

[1] Android (operating system), Wikipedia, 26 July 2021,
https://en.wikipedia.org/wiki/Android_(operating_system)

[2] Ιστορία εκδόσεων του Android, Βικιπαιδεια, 17 Μαρτίου 2020,
https://el.wikipedia.org/wiki/%CE%99%CF%83%CF%84%CE%BF%CF%81%CE%AF%CE%B1_%CE%B5%CE%BA%CE%B4%CF%8C%CF%83%CE%B5%CF%89%CE%BD_%CF%84%CE%BF%CF%85_Android.

[3] Κρυπτογραφία, Βικιπαιδεια, 27 Ιουνίου 2021,
https://el.wikipedia.org/wiki/%CE%9A%CF%81%CF%85%CF%80%CF%84%CE%BF%CE%B3%CF%81%CE%B1%CF%86%CE%AF%CE%B1

[4] Αραμπατζής Αναστάσιος, Κρυπτογράφηση και Αποκρυπτογράφηση,
Homodigitalis, https://www.homodigitalis.gr/posts/4305.

[5] History of cryptography, Wikipedia, 6 July 2021,
https://en.wikipedia.org/wiki/History_of_cryptography

[6] Advanced Encryption Standard, Wikipedia, 1 July 2021,
https://en.wikipedia.org/wiki/Advanced_Encryption_Standard