ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΥΠΗΡΕΣΙΑ ΑΠΟΘΗΚΕΥΣΗΣ ΚΑΙ ΣΥΓΧΡΟΝΙΣΜΟΥ ΑΡΧΕΙΩΝ ΣΤΟ CLOUD

Διπλωματική Εργασία

Νόβας Ιωάννης

Επιβλέπων: Σταμούλης Γεώργιος

Βόλος 2021

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΥΠΗΡΕΣΙΑ ΑΠΟΘΗΚΕΥΣΗΣ ΚΑΙ ΣΥΓΧΡΟΝΙΣΜΟΥ ΑΡΧΕΙΩΝ ΣΤΟ CLOUD

Διπλωματική Εργασία

Νόβας Ιωάννης

Επιβλέπων: Σταμούλης Γεώργιος

Βόλος 2021

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

A CLOUD BASED SYNCHRONIZATION SERVICE

Diploma Thesis

Novas Ioannis

Supervisor: Stamoulis Georgios

Volos 2021

Εγκρίνεται από την Επιτροπή Εξέτασης:

Επιβλέπων **Σταμούλης Γεώργιος**

Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Μέλος **Βασιλακόπουλος Μιχαήλ**

Αναπληρωτής Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Ηλεκτρονικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Μέλος **Θάνος Γεώργιος**

ΕΔΙΠ, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών

Ηλεκτρονικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Ημερομηνία έγκρισης: 07-07-2021

ΕΥΧΑΡΙΣΤΙΕΣ

Πρώτα απ όλα θα ήθελα να ευχαριστήσω τους γονείς μου και τον αδερφό μου που μια ζωή τρέχουν για μένα μεταφορικά και κυριολεκτικά, με στηρίζουν και με εμπιστεύονται και ελπίζω η πορεία μου εώς τώρα να τους κάνει περήφανους. Ακόμη θα ήθελα να ευχαριστήσω μέσα από την καρδία μου τον παππού τον Γιάννη και την γιαγιά την Κούλα οι οποιοι αγωνίστηκαν και αγωνίζονται για να μπορέσω να σπουδάσω και να πετύχω τους στόχους μου, όπως επίσης και την θεία μου την Ντίνα που από μικρό παιδί μου μεταλαμπαδεύει γνώσεις και αξίες. Επίσης θα ήθελα να ευχαριστήσω τον παππού τον Μανώλη και την γιαγιά την Χρύσα που όποτε και αν χρειάστηκαν ήταν εκεί και με το παραπάνω.

Οπωσδήποτε θα ήθελα να ευχαριστήσω μέσα από την καρδιά μου τους φίλους μου, τον Μούλο, τον Τζήμο, τον Ψηλό, τον Τσακλίδη, τον Στάθη, τον Τάσο και τον Αλέκο, με τους οποίους μοιραστήκαμε τα πιο όμορφα, ξέγνοιαστα και ανεκτίμητα χρόνια, τα φοιτητικά.

Τέλος θα ήθελα να ευχαριστήσω τον κύριο Θάνο. Ήταν ο καθοδηγητής και συνεργάτης που με τις συμβουλές του και την υπομονή του κατάφερα να ολοκληρώσω την παρούσα εργασία.

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο Δηλών

Νόβας Ιωάννης
30.6.2021

ΠΕΡΙΛΗΨΗ

Τα τελευταία χρόνια έχει παρατηρηθεί αύξηση της εξ αποστάσεως εργασίας τόσο σε επίπεδο επιχειρήσεων όσο και σε ατομικό επίπεδο. Αυτό δημιουργεί την ανάγκη για εξ αποστάσεως αποθήκευση και διαμοιρασμό δεδομένων και το συγχρονισμό των αρχείων και των φακέλων των τελικών χρηστών. Για την υλοποίηση των παραπάνω λειτουργιών καθοριστικής σημασίας είναι η ύπαρξη υπηρεσιών συγχρονισμού αρχείων σε πραγματικό χρόνο. Σκοπός της παρούσας διπλωματικής εργασίας είναι η δημιουργία μιας υπηρεσίας που ικανοποιεί την παραπάνω απαίτηση.

Η παρούσα εργασία περιλαμβάνει μια εφαρμογή που εκτελείται στο τερματικό του χρήστη, μία υπηρεσία εξυπηρετητή (server side) και μία εφαρμογή διαδικτύου για την παροχή της υπηρεσίας μέσω web-browser. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής τόσο στην πλευρά του server όσο και του client είναι η Python. Για την εφαρμογή που παρέχει το web-interface χρησιμοποιήθηκε το micro web framework Flask.

ABSTRACT

In recent years there has been observed a significant increase in teleworking especially with the advent of the pandemic of the COVID-19 virus. This creates the need for cloud based storage and data sharing services as well as the synchronization of files and folders of the end users at runtime. The existence of robust and real time file synchronization services are crucial to implement those functionalities. The purpose of this thesis is to create a service which fulfills these requirements.

The present project includes an application which executes on the user's terminal , a server based service and a web-interface for accessing the service via a web browser. The programming language which was used for the development of the application is Python. We have used the Flask micro web framework for the web interface part of the project.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

HTML	Hypertext Markup Language
CSS	Cascading Style Sheet
JSON	JavaScript Object Notation
HTTP	Hypertext Transfer Protocol
App	Application
URL	Uniform Resource Locator
FIFO	First In First Out
AES	Advanced Encryption Standard
SHA	Secure Hash Algorithm
GUI	Graphical User Interface
WSGI	Web Server Gateway Interface
XML	Extensible Markup Language
JWT	Json Web Token

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΕΦΆΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

Η παρούσα διπλωματική εργασία υλοποιεί μία υπηρεσία συγχρονισμού αρχείων με αξιοποίηση του cloud [6] στα πρότυπα υπηρεσιών όπως είναι το Dropbox [17], το MEGA[18] και το Pithos του ΕΔΕΤ [19].

Η υπηρεσία αποτελείται από τρεις επιμέρους οντότητες λογισμικού οι οποίες συνεργάζονται ώστε να παρέχουν την υπηρεσία προς τον τελικό χρήστη. Η βασική υπηρεσία αποτελείται από έναν εξυπηρετητή που επιτυγχάνει την αποθήκευση και το συγχρονισμό των αρχείων μεταξύ των τερματικών συσκευών του χρήστη. Για να το πετύχει αυτό διατηρεί σε κάθε χρονική στιγμή ένα στιγμιότυπο των αρχείων του χρήστη, το οποίο και προσπαθεί να συγχρονίσει με όλες τις επιμέρους απομακρυσμένες συσκευές που είναι συνδεδεμένες υπό τον ίδιο λογαριασμό. Πιο συγκεκριμένα, ο εξυπηρετητής λαμβάνει πληροφορίες αναφορικά με τις αλλαγές που έχουν συμβεί στο τερματικό του χρήστη, αποθηκεύει αυτές τις αλλαγές αυτές τοπικά και συγχρονίζει άμεσα τις υπόλοιπες συσκευές με βάση τις συγκεκριμένες αλλαγές.

Η επικοινωνία μεταξύ του εξυπηρετητή και των τερματικών συσκευών γίνεται μέσω της εφαρμογής συγχρονισμού που εκτελείται τοπικά στον τερματικό υπολογιστή του χρήστη και είναι υλοποιημένη σε γλώσσα Python [9]. Προκειμένου να επιτευχθεί ο συγχρονισμός, κάθε τερματικό του χρήστη θα πρέπει να εκτελεί ένα στιγμιότυπο της συγκεκριμένης εφαρμογής με τα χαρακτηριστικά ταυτοποίησης του. Η εφαρμογή έχει την δυνατότητα να αποστείλει στον εξυπηρετητή ενημερώσεις σχετικά με αλλαγές που έχουν συμβεί τοπικά, αλλά και να δεχθεί από τον εξυπηρετητή ενημερώσεις για τις αλλαγές που έχουν γίνει σε απομακρυσμένες συσκευές υπό τον συνδεδεμένο λογαριασμό.

Υπό αυτό το πρίσμα, η εφαρμογή καθίσταται ιδανική για την συνεργασία απομακρυσμένων ομάδων χρηστών σε ένα κοινό αποθετήριο ή για την διατήρηση πολλαπλών τερματικών συσκευών, των οποίων τα αρχεία και οι φάκελοι είναι συγχρονισμένα υπό τον ίδιο λογαριασμό.

Εκτός, της εφαρμογής συγχρονισμού που διασφαλίζει την άμεση ενημέρωση του εξυπηρετητή με τις αλλαγές από τον χρήστη και στη συνέχεια την ενημέρωση όλων των

συνδεδεμένων τερματικών συσκευών, διατίθεται και ένα web-interface, ώστε ο χρήστης να έχει άμεση πρόσβαση στα αρχεία του από οποιοδήποτε υπολογιστή ή φορητή συσκευή.

Ο χρήστης διαθέτει τον προσωπικό του "φάκελο συγχρονισμού" μέσα στον οποίο περιλαμβάνει φακέλους, υποφακέλους και αρχεία στα οποία έχει πλήρη πρόσβαση από όποια συσκευή επιθυμεί. Ο "φάκελος συγχρονισμού" δημιουργείται τοπικά στον υπολογιστή του χρήστη όταν συνδεθεί μέσω της εφαρμογής συγχρονισμού από κάποια συσκευή. Ο χρήστης μπορεί να δημιουργήσει, να επεξεργαστεί, να μετονομάσει, να μετακινήσει και να διαγράψει οποιοδήποτε αντικείμενο βρίσκεται εντός του "φακέλου συγχρονισμού" και η αλλαγή αυτή θα συγχρονιστεί αυτόματα σε όλες τις συνδεδεμένες συσκευές του. Παράλληλα, ο χρήστης μπορεί να παρακολουθεί, να επεξεργάζεται και να λαμβάνει ή να μεταφορτώνει αρχεία μέσω του web-interface της εφαρμογής.

Παρόμοιες υπηρεσίες υπάρχουν αρκετές, όπως για παράδειγμα το Google Drive [20], το Box [21], το MEGA και την δημοφιλέστερη όλων το Dropbox. Σε αυτό το σημείο θα πρέπει να διακρίνουμε τις υπηρεσίες οι οποίες αξιοποιούν το cloud [6] σε δύο βασικές κατηγορίες. Υπάρχουν οι υπηρεσίες που δημιουργήθηκαν με γνώμονα τον συγχρονισμό των αρχείων μεταξύ διαφορετικών τερματικών συσκευών του ίδιου χρήστη και οι υπηρεσίες που φτιάχτηκαν με γνώμονα την δημιουργία αντιγράφων ασφαλείας. Οι πρώτες συνήθως προσφέρουν έναν ειδικό φάκελο και οτιδήποτε τοποθετείται μέσα σε αυτόν συγχρονίζεται με το διαδικτυακό χώρο αποθήκευσης καθώς επίσης και με τις άλλες συνδεδεμένες συσκευές. Συνεπώς επειδή αυτές οι υπηρεσίες έχουν σχεδιαστεί για συγχρονισμό δεν διαθέτουν μνήμη σε βάθος χρόνου αναφορικά με τα αντικείμενα που διαγράφονται. Περιέχουν ένα σύστημα τύπου κάδου απορριμάτων αλλά δεν παρέχουν την δυνατότητα ανάκτησης του ιστορικού μεταβολών σε βάθος χρόνου.

Οι υπηρεσίες της δεύτερης κατηγορίας προσφέρουν στο χρήστη ένα διαδικτυακό χώρο αποθήκευσης που δεν συνδέεται με κάποιο τοπικό χώρο αποθήκευσης και σκοπό έχει την διασφάλιση της ακεραιότητας και ιδιωτικότητας των αρχείων. Η πιο δημοφιλές υπηρεσία αυτής της κατηγορίας είναι το Blackblaze [22] το οποίο προσφέρει μεγάλο αποθηκευτικό χώρο στο cloud. Οι υπηρεσίες αυτές είναι κατάλληλες για την αποθήκευση αρχείων που περιέχουν ευαίσθητα προσωπικά δεδομένα, καθώς προσφέρουν την δυνατότητα στον χρήστη να δημιουργεί το δικό του κλειδί κρυπτογράφησης προκειμένου να

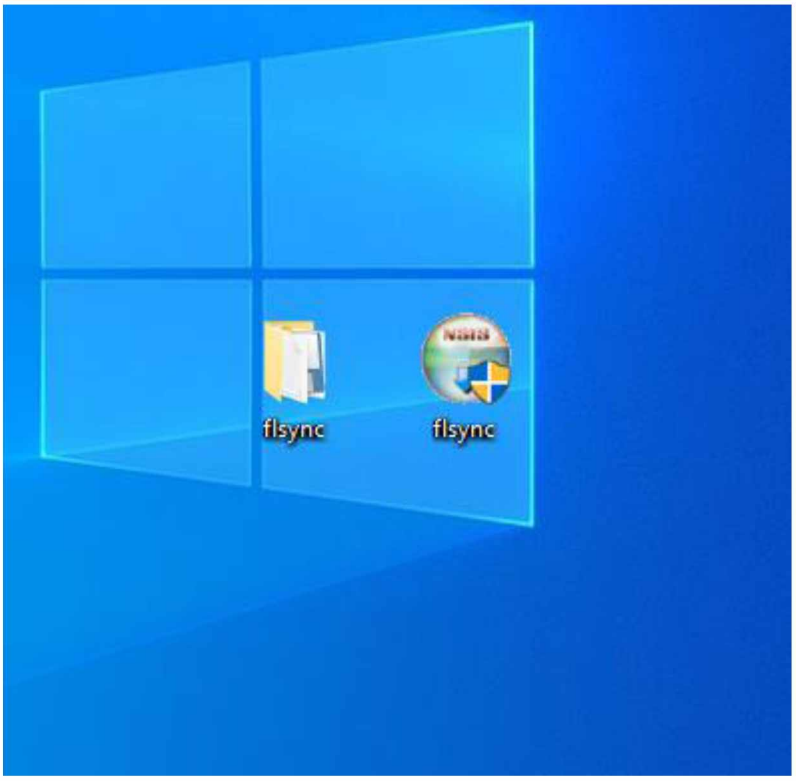
προστατεύονται τα αρχεία από μη εξουσιοδοτημένη ή κακόβουλη πρόσβαση στον διακομιστή αντιγράφων ασφαλείας. Το Dropbox και το Google Drive δεν προσφέρουν αυτή την υπηρεσία και ο πελάτης αν επιθυμεί μεγαλύτερη ασφάλεια θα πρέπει να χρησιμοποιεί ξεχωριστό λογισμικό κρυπτογράφησης προτού μεταφορτώσει τα αρχεία του. Επίσης οι υπηρεσίες ασφαλείας προσφέρουν περισσότερο χώρο αποθήκευσης συγκριτικά με τις υπηρεσίες συγχρονισμού. Η παρούσα υπηρεσία προσφέρει συγχρονισμό αρχείων σε όλες τις συνδεδεμένες συσκευές και πρόσβαση στο χώρο αποθήκευσης μέσω ενός web-interface.

ΚΕΦΆΛΑΙΟ 2

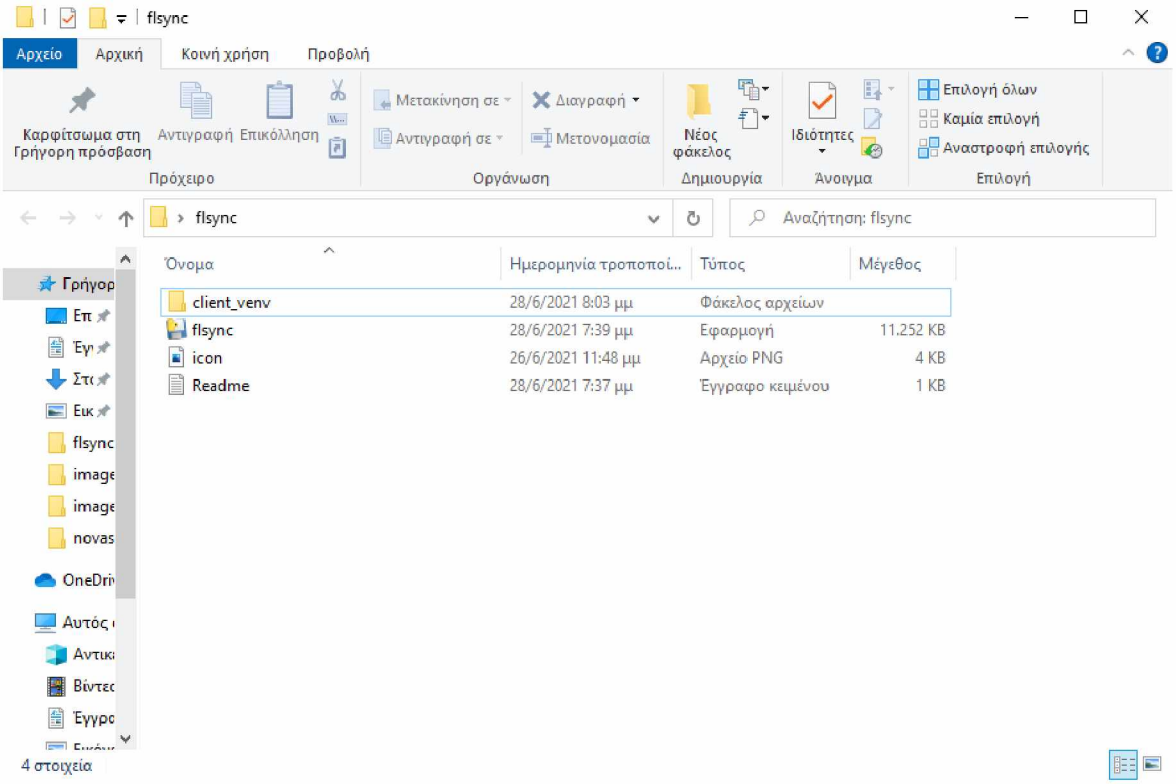
ΛΕΙΤΟΥΡΓΙΚΗ ΠΕΡΙΓΡΑΦΗ DESKTOP APPLICATION

2.1 Εγκατάσταση της εφαρμογής συγχρονισμού

Ο χρήστης κατεβάζει από την ιστοσελίδα της υπηρεσίας το αρχείο setup. Τρέχοντας το setup θα δημιουργηθεί στο σημείο που θα επιλέξει ένας φάκελος με όνομα flsync μέσα στον οποίο υπάρχει το εκτελέσιμο αρχείο και τα συνοδευτικά αρχεία για να λειτουργεί σωστά η εφαρμογή. Ακόμη υπάρχει και ένα αρχείο Readme.txt το οποίο περιέχει οδηγίες χρήσης της εφαρμογής.



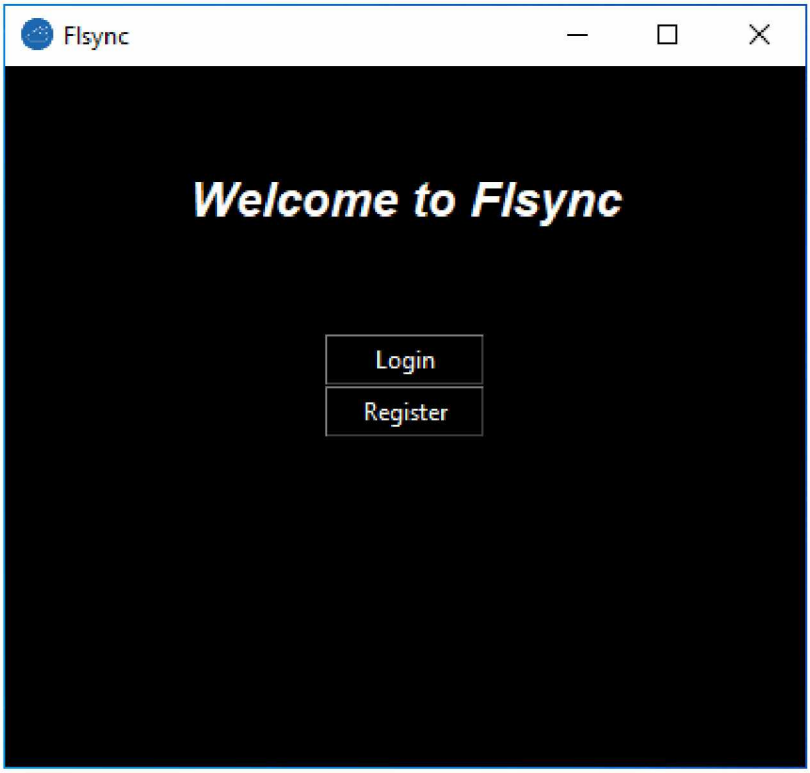
Εικόνα 2.1: Αρχείο setup και φάκελος εφαρμογής



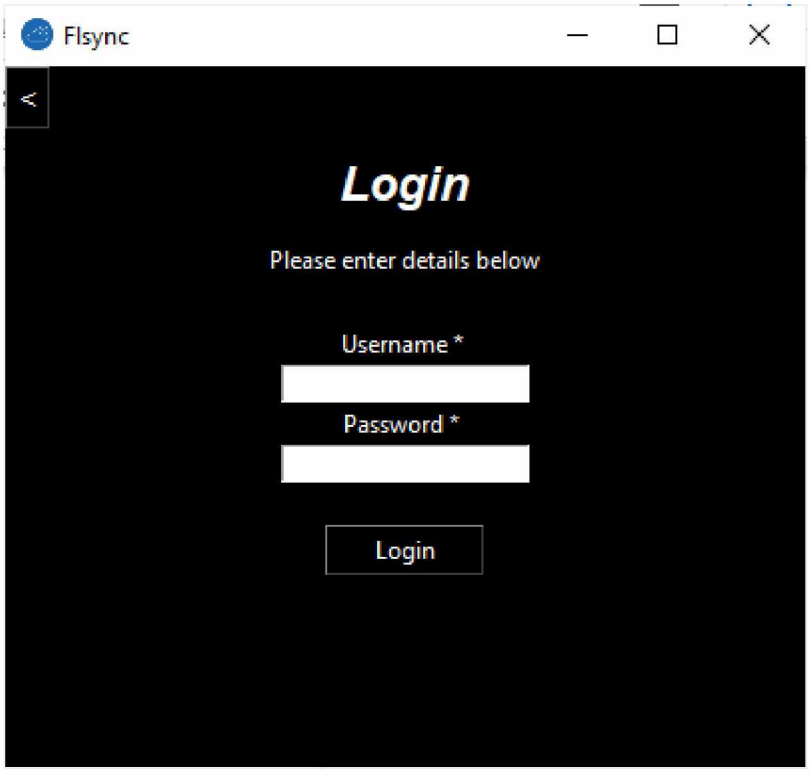
Εικόνα 2.2: Περιεχόμενα φακέλου flsync

2.2 Δημιουργία λογαριασμού ή ταυτοποίηση υφιστάμενου χρήστη

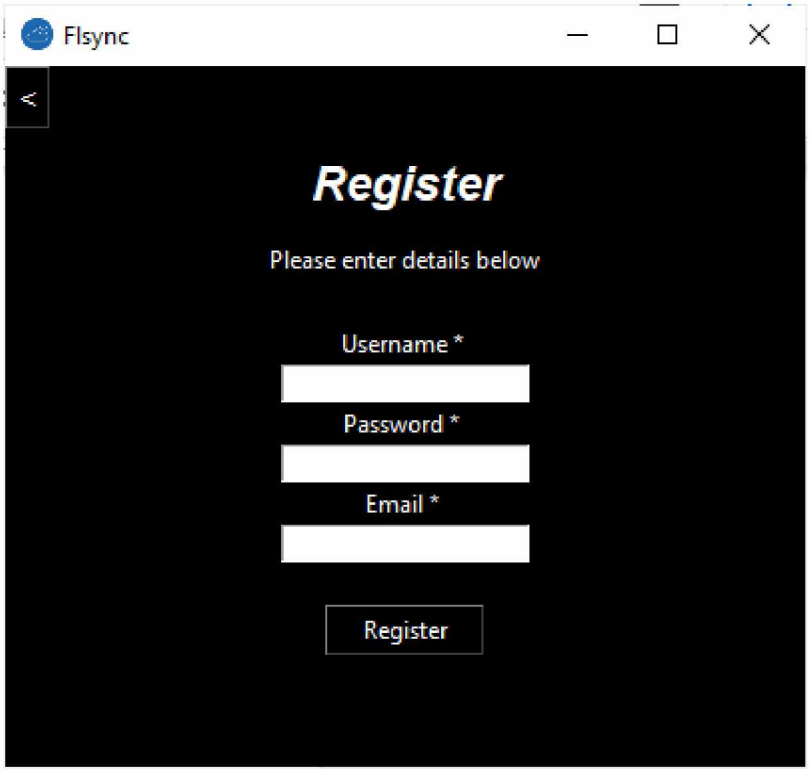
Ο χρήστης αφού ανοίξει την εφαρμογή καλείται είτε να κάνει νέο λογαριασμό μέσω της επιλογής “Register” είτε να συνδεθεί με τα διαπιστευτήρια του μέσω της επιλογής “Login”



Εικόνα 2.3: Αρχική σελίδα εφαρμογής



Εικόνα 2.4: Login

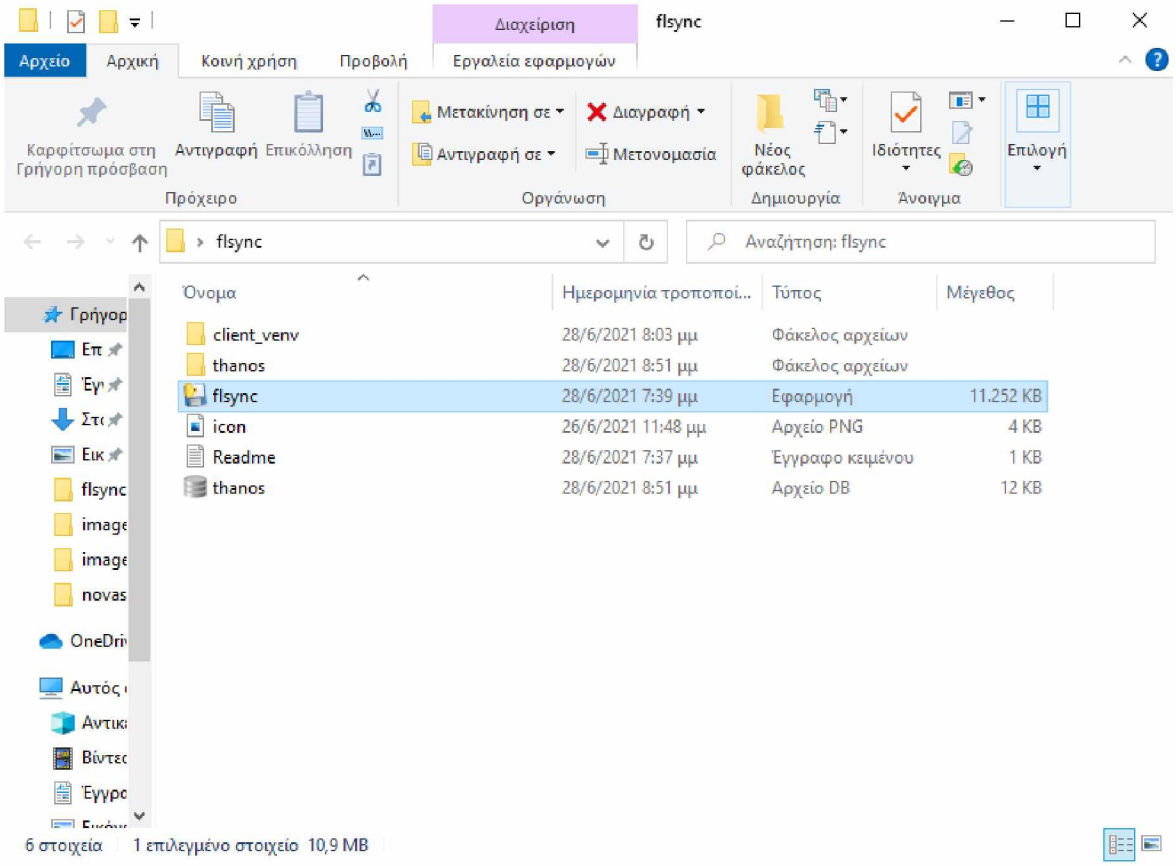


Εικόνα 2.5: Register

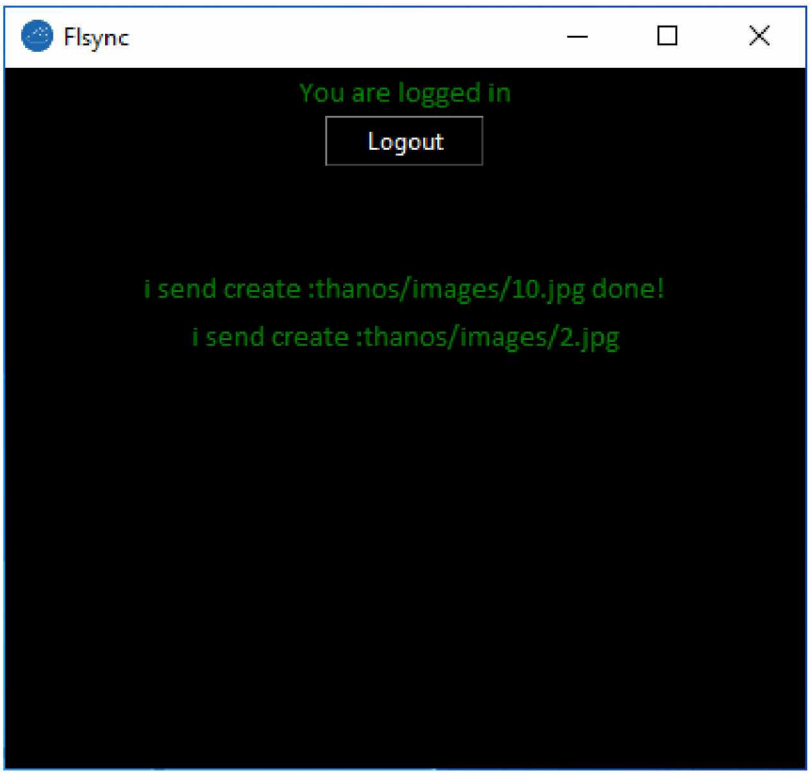
2.3 Φάκελος συγχρονισμού και μηνύματα ενημέρωσης

Αφού ο χρήστης δημιουργήσει με επιτυχία νέο λογαριασμό θα δημιουργηθεί ο φάκελος συγχρονισμού του, με όνομα φακέλου το username του στην υπηρεσία. Επιπλέον, θα δημιουργηθεί τοπικά και βάση δεδομένων με όνομα username.db η οποία είναι απαραίτητη για την ομαλή λειτουργία του συστήματος. Στην περίπτωση που ο χρήστης ταυτοποιηθεί στην υπηρεσία, θα δημιουργηθεί και πάλι ο φάκελος συγχρονισμού και η βάση δεδομένων εφόσον δεν υπάρχουν, ενώ εάν υπάρχουν θα ενημερωθούν.

Για οποιαδήποτε μεταβολή αρχείου ή φακέλου συμβαίνει μέσα στον φάκελο συγχρονισμού ενώ ο χρήστης είναι συνδεδεμένος στέλνεται η αντίστοιχη ενημέρωση στον εξυπηρετητή. Κατά την διαδικασία αποστολής η παραλαβής μιας ενημέρωσης εμφανίζεται ένα περιληπτικό μήνυμα ενημέρωσης στο παράθυρο του χρήστη όπου περιλαμβάνει τον τύπο της ενημέρωσης και το όνομα του αρχείου ή φακέλου που αφορά. Όταν ολοκληρωθεί η εκτέλεση της ενημέρωσης δίπλα από το μήνυμα ενημέρωσης εμφανίζεται το μήνυμα done! το οποίο μετά από λίγο παύει να εμφανίζεται.



Εικόνα 2.6: Δημιουργία φακέλου και βάσης για τον χρήστη με username thanos



Εικόνα 2.7: Παράθυρο εμφάνισης ενημερώσεων

ΚΕΦΆΛΑΙΟ 3

ΑΝΑΛΥΤΙΚΗ ΤΕΧΝΙΚΗ ΠΕΡΙΓΡΑΦΗ

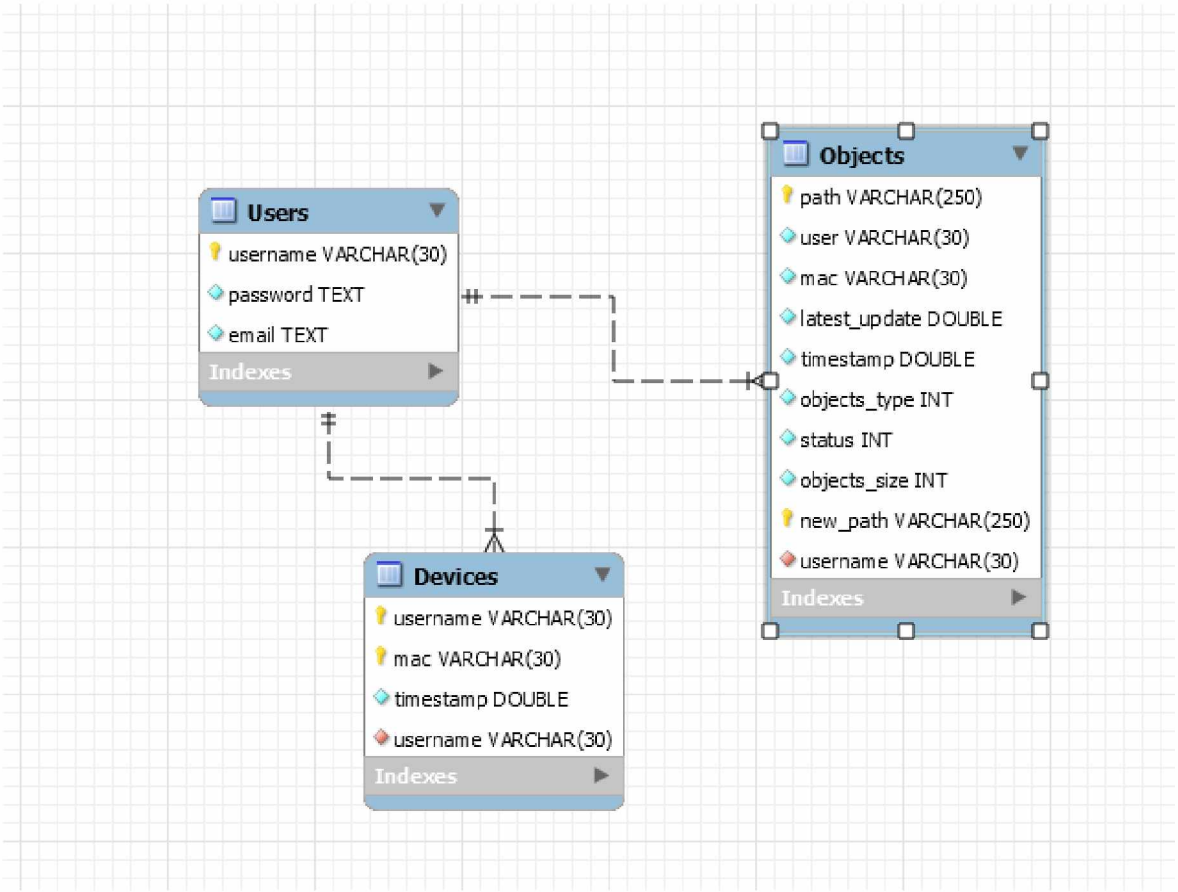
3.1 Τεχνική περιγραφή του server

3.1.1 Εισαγωγή

Ο server αποτελείται από τρία βασικά νήματα τα οποία αποτελούν τον κορμό του. Το πρώτο δημιουργεί ένα socket επικοινωνίας στο port 8001. Για κάθε client που συνδέεται σε αυτο'δημιουργεί ένα νέο νήμα [7] το οποίο χρησιμοποιεί το κανάλι επικοινωνίας, που δημιουργήθηκε για να επιτευχθεί η ταυτοποίηση του χρήστη. Το δεύτερο νήμα δημιουργεί ένα socket επικοινωνίας στο port 8002 το οποίο δέχεται συνδέσεις από χρήστες που έχουν ταυτοποιηθεί και δημιουργεί ένα νέο νήμα στο οποίο ο server μέσω του καναλιού επικοινωνίας στέλνει ενημερώσεις στον client. Το τρίτο νήμα χρησιμοποιεί το port 8003 με σκοπό την αποστολή ενημερώσεων από τον client στον server.

3.1.2 Αποθήκευση αρχείων στον server και τη βάση δεδομένων

Για κάθε χρήστη υφίσταται στον server ένας φάκελος με το username του, όπου περιλαμβάνει όλο το filesystem του. Μονάχα τα αρχεία που έχουν μεταφορτωθεί στον εξυπηρετητή με επιτυχία είναι διαθέσιμα και προσβάσιμα από κάθε συσκευή του χρήστη. Ο εξυπηρετητής διαθέτει σχεσιακή βάση δεδομένων MySQL [8] στην οποία αποθηκεύει πληροφορίες σχετικά με τους χρήστες, τις συσκευές τους αλλά και τα αρχεία τους. Μια ενημέρωση που παραλήφθηκε θεωρείται ότι εκτελέστηκε επιτυχημένα εφόσον ενημερώθηκε ο εξυπηρετητής και η βάση δεδομένων. Στο παρακάτω σχεσιακό μοντέλο απεικονίζονται οι σχέσεις των λειτουργικών οντοτήτων και ο τρόπος που συνδέονται αυτές μεταξύ τους.



Εικόνα 3.1: Σχεσιακό μοντέλο Βάσης δεδομένων του server

Σε αυτό το σημείο αξίζει να εξηγήσουμε μερικές παραμέτρους των σχέσεων καθώς θα χρειαστούν για την κατανόηση των λειτουργιών του server. Η οντότητα Users περιγράφει τον χρήστη της υπηρεσίας μέσω των πεδίων username, password και email. Για την οντότητα Devices, το username και το mac αποτελούν το κλειδί της. Επειδή την ίδια συσκευή είναι πιθανό να την χρησιμοποιούν δύο χρήστες το MAC address μόνο του δεν επαρκεί. Επίσης το timestamp συμβολίζει την στιγμή που ενημερώθηκε τελευταία φορά η συσκευή. Μεγάλη προσοχή αξίζει να δοθεί στην ανάλυση της οντότητας Objects. Τα πεδία path και new_path αποτελούν το μοναδικό κλειδί της οντότητας και αυτό καθώς θεωρήθηκε απαραίτητο για τις περιπτώσεις μετονομασίας, όπου το path σε αυτήν την περίπτωση θεωρείται το παλιό όνομα και το new_path προφανώς το νέο όνομα. Το path περιέχει το πλήρες μονοπάτι από τον φάκελο συγχρονισμού και μετά. Το user δείχνει σε ποιόν χρήστη ανήκει η καταχώρηση και το mac δείχνει ποιά συσκευή του ενημέρωσε τελευταία φορά το αντικείμενο. Το latest_udpate αποθηκεύει το πότε ενημερώθηκε το

αντικείμενο τελευταία φορά. Από την άλλη το timestamp αφορά την ενημέρωση του server, συμβολίζει δηλαδή την χρονική στιγμή που ο server έλαβε και ολοκλήρωσε την αλλαγή. Για κάθε ενημέρωση που δέχεται ο server δημιουργεί ένα timestamp προκειμένου να σειριοποιήσει τις αλλαγές με στόχο να στέλνονται στους clients με την σειρά που τις δέχεται αυτός. Οι παράμετροι objects_type και objects_size εκφράζουν αυτό που περιγραφουν το όνομά τους, συγκεκριμένα το objects_type παίρνει την τιμή 0 για αρχεία και 1 για φακέλους. Τέλος το status συμβολίζει την κατάσταση του αντικειμένου. Δηλαδή αν το αντικείμενο δεν υφίσταται πλέον στον server έχει την τιμή 0, αν υφίσταται έχει την τιμή 1 και αν έχει υποστεί μετονομασία έχει την τιμή 2.

3.1.3 Υποδοχή και προώθηση ενημερώσεων, server

Οι ενέργειες που γίνονται ανάλογα τον τύπο της ενημέρωσης είναι οι εξής:

- Μήνυμα Δημιουργίας

  Αρχικά ελέγχεται αν το αρχείο έχει δημιουργηθεί ήδη από κάποια άλλη συσκευή, εξετάζοντας την ύπαρξη καταχώρησης στην βάση. Αν ισχύει η παραπάνω υπόθεση επιστρέφει μήνυμα αναφοράς “conflict”. Διαφορετικά δημιουργεί το αντικείμενο, μεταφορτώνει τα απαραίτητα bytes αν χρειάζεται, δημιουργεί την αντίστοιχη καταχώρηση στην βάση και ενημερώνει τον client πως ολοκλήρωσε με επιτυχία.

- Μήνυμα Ενημέρωσης

  Και σε αυτήν την περίπτωση αρχικά ελέγχεται αν προκύπτει σύγκρουση αρχείων, εξετάζοντας αν το πεδίο old_latest_update του μηνύματος είναι ίδιο με το τρέχον latest_update της βάσης, αν δηλαδή η αλλαγή που εντόπισε και έστειλε ο client έχει πραγματοποιηθεί στο υπάρχον αρχείο του server. Σε περίπτωση σύγκρουσης στέλνει το μήνυμα αναφοράς “conflict” ενώ σε διαφορετική περίπτωση προχωράει στην ενημέρωση του αρχείου, της βάσης και στην αποστολή επιβεβαίωσης στον client.

- Μήνυμα Μετονομασίας

Στην περίπτωση της μετονομασίας το μόνο που ελέγχεται είναι αν υπάρχει το αντικείμενο που πρόκειται να μετονομαστεί, αν δηλαδή δεν το έχει διαγράψει κάποιος άλλος προτού γίνει η μετονομασία. Διαφορετικά μετονομάζει το αντικείμενο και τις κατάλληλες καταχωρήσεις στην βάση. Συγκεκριμένα ως προς την βάση, μετονομάζει την αρχική καταχώρηση και δημιουργεί μια νέα με path αυτό του παλιού path του αντικειμένου, με new_path αυτό του του νέου path, με status ίσο με 2 και με timestamp αυτό της μετονομασίας.

- Μήνυμα Διαγραφής

  Αν το αντικείμενο δεν υπάρχει τότε το αγνοεί. Αν το αντικείμενο που διαγράφηκε στην συσκευή έχει τροποποιηθεί στον server πριν την διαγραφή του τότε αν είναι φάκελος του στέλνει μήνυμα αναφοράς “recreate” διαφορετικά “ignore”. Αλλιώς διαγράφει το αντικείμενο, ενημερώνει το status της καταχώρησης σε 0 και στέλνει μήνυμα αναφοράς “completed”.

Αφού ολοκληρωθεί η διαδικασία υποδοχής και επεξεργασίας ακολουθεί η διαδικασία προώθησης στις υπόλοιπες ενεργές συσκευές. Για κάθε ενεργή συσκευή υπάρχει μια ουρά τύπου FIFO στην οποία προστίθενται οι ενημερώσεις που προέρχονται από τις διαφορετικές συσκευές του χρήστη. Από αυτην την ουρά λαμβάνει ενημερώσεις το νήμα αποστολής ενημερώσεων που διαθέτει ο server για κάθε client. Επομένως μετά το πέρας της διαδικασίας και επεξεργασίας προστίθενται στις κατάλληλες ουρές η νέα ενημέρωση, γίνεται η αποστολή της και αφού ολοκληρωθεί με επιτυχία ενημερώνεται το timestamp της συσκευής στην βάση του server.

3.2 Τεχνική περιγραφή του client

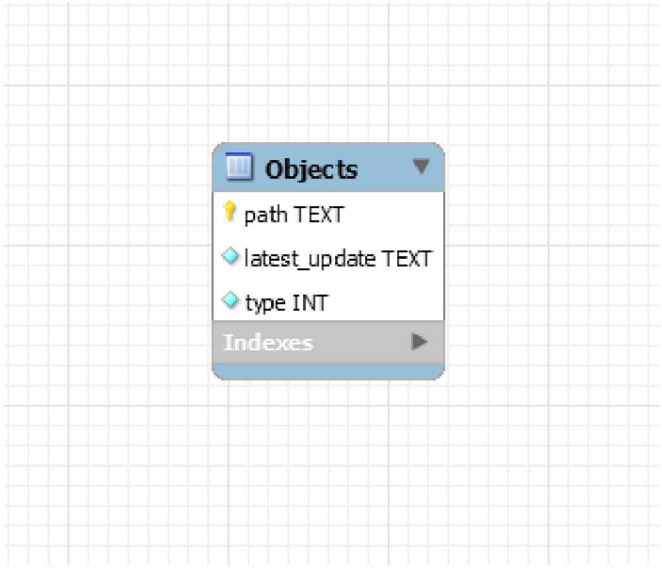
3.2.1 Βασικός σκελετός του client

Ο client αφου συνδεθεί στον socket με port 8001 μέσω του βασικού του νήματος και ταυτοποιηθεί μέσω των credentials του ξεκινά τα τρία κύρια νήματα του. Το πρώτο είναι το νήμα υποδοχής ενημερώσεων, όπου συνδέεται στον server στο port 8002, το δεύτερο είναι το νήμα αποστολής ενημερώσεων, όπου αντίστοιχα συνδέεται στον server στο port 8003 μέσω του οποίου θα στέλνει τις ενημερώσεις στον server. Το τρίτο νήμα

είναι αυτή που παρακολουθεί τον φάκελο συγχρονισμού και προωθεί νέες ενημερώσεις στο νήμα αποστολής ενημερώσεων. Η παρακολούθηση του φακέλου συγχρονισμού πραγματοποιείται με τη βοήθεια της τεχνολογίας watchdog. Το watchdog είναι ένα Python module που εντοπίζει αλλαγές συστήματος σε ένα προκαθορισμένο φάκελο, στην περίπτωσή μας ο φάκελος αυτός είναι ο φάκελος συγχρονισμού.

3.2.2 Βάση δεδομένων του client

Ο client όπως είναι λογικό μπορεί να κάνει αλλαγές στο filesystem που υπόκειται στον φάκελο συγχρονισμού ακόμη και όταν είναι offline, δηλαδή όταν δεν τρέχει την εφαρμογή client στα τερματικά του και αυτό οδηγεί στην ανάγκη καταγραφής των πληροφοριών των αρχείων όσο είναι offline. Η ανάγκη αυτή ικανοποιείται με την χρήση της SQLite η οποία είναι μια βιβλιοθήκη σε C που παρέχει μια απλή σχεσιακή βάση δεδομένων η οποία δεν απαιτεί διακομιστή και εξυπηρετεί την τοπική αποθήκευση δεδομένων [10]. Ο client αποθηκεύει στην βάση τα ονόματα των αρχείων, τον τύπο τους (φάκελος ή αρχείο) και το timestamp της τελευταίας καταγεγραμμένης αλλαγής του. Ακολουθεί το σχεσιακό μοντέλο του.



Εικόνα 3.2: Σχεσιακό μοντέλο Βάσης δεδομένων του client

3.2.3 Υποδοχή και εκτέλεση ενημερώσεων client

Όπως έχει αναφερθεί ο client διαθέτει ένα νήμα υποδοχής ενημερώσεων το οποίο εξυπηρετεί και στην εκτέλεση τους. Όταν ο client πραγματοποιεί τις ενημερώσεις που

δέχεται από τον server ενημερώνει ή δημιουργεί την αντίστοιχη καταχώρηση στην βάση δεδομένων του με latest_update ίσο με -1 έτσι ώστε το watchdog να καταλαβαίνει ότι οι αλλαγές προέρχονται από τον server. Πιο συγκεκριμένα ανάλογα με την ενημέρωση συμβαίνουν τα εξής :

- Μήνυμα δημιουργίας

  Ελέγχει αν η ενημέρωση έχει εκτελεστεί στο παρελθόν αλλά δεν πρόλαβε να ενημερώσει τον server, ελέγχοντας αν το αρχείο υπάρχει και αν ταυτίζονται τα timestamp τελευταίας ενημέρωσης της βάσης, του μηνύματος και του αρχείου. Σε αυτή την περίπτωση αγνοεί το μήνυμα και στέλνει μήνυμα αναφοράς “completed” στον server. Ελέγχει αν το αρχείο έχει αλλάξει τοπικά αλλά δεν έχει ενημερωθεί ο server. Σε αυτή την περίπτωση έχουμε σύγκρουση αρχείων, οπότε δημιουργεί ένα συμπληρωματικό conflict αρχείο και η διαδικασία από εκεί και πέρα είναι ίδια με την περίπτωση που δεν ισχύει καμία από τις δύο παραπάνω περιπτώσεις. Συγκεκριμένα, θέτει το latest_update στην καταχώρηση της βάσης με -1 ή την δημιουργεί, στέλνει στον server το μήνυμα αναφοράς “send bytes”, υποδέχεται τα bytes και ενημερώνει το αρχείο, ενημερώνει το timestamp τελευταίας αλλαγής του αρχείου και το latest_update στην καταχώρηση της βάσης και στέλνει μήνυμα αναφοράς “completed”.

- Μήνυμα ενημέρωσης

  Ελέγχει αν υπάρχει σύγκρουση αρχείων, θέτει το latest_update ίσο με -1 στην βάση δεδομένων, ζητά bytes αν χρειάζεται, ενημερώνει latest_update αρχείου και βάσης και στέλνει μήνυμα αναφοράς “completed”.

- Μήνυμα μετονομασίας

  Στις περιπτώσεις όπου το παλιό όνομα δεν υπάρχει ή το νέο όνομα υπάρχει αγνοεί το μήνυμα και στέλνει μήνυμα αναφοράς “completed”. Διαφορετικά ενημερώνει το latest_update της βάσης με -1 του παλιού ονόματος πραγματοποιεί την μετονομασία στο αντικείμενο και στην βάση και επαναφέρει το latest_update στην βάση.

- Μήνυμα διαγραφής

Αν το αντικείμενο δεν υπάρχει στην συσκευή το διαγράφει από την βάση, αν υπάρχει σε αυτή, και στέλνει μήνυμα αναφοράς "completed". Στην αντίθετη περίπτωση ελέγχει αν το περιεχόμενο του αρχείου που διαγράφηκε στον server είναι ίδιο με αυτό που υπάρχει. Αυτό πραγματοποιείται με τον έλεγχο του latest_update του μηνύματος, δηλαδή με το latest_update του αρχείου στον server. Αν ναι τότε πραγματοποιεί την διαγραφή διαφορετικά το αρχείο έχει υποστεί αλλαγές και δεν έχει προλάβει να ενημερωθεί ο server οπότε αγνοεί την διαγραφή και στέλνει μήνυμα αναφοράς "completed". Σε διαφορετική περίπτωση ενημερώνει το timestamp της καταχώρησης της βάσης με -1, διαγράφει το αντικείμενο, διαγράφει την καταχώρηση στην βάση και στέλνει μήνυμα αναφοράς "completed".

3.2.4 Προώθηση ενημερώσεων από watchdog σε νήμα αποστολής

Η διαδικασία αποστολής ενημερώσεων από τον client στον server γίνεται με σειριακό τρόπο. Συγκεκριμένα, η αποστολή μιας ενημέρωσης ξεκινά όταν ολοκληρωθεί πλήρως η διαδικασία αποστολής της προηγούμενης. Αυτό μπορεί να πάρει κάποια δευτερόλεπτα ανάλογα με την αλλαγή που σημειώθηκε. Για παράδειγμα το "ανέβασμα" ενός βίντεο θα διαρκέσει αρκετά δευτερόλεπτα μέσα στα οποία μπορεί να συμβούν νεότερες αλλαγές στο filesystem του φακέλου συγχρονισμού. Το watchdog παρέχει μια ουρά αναμονής των συμβάντων που συνδέονται με τις αλλαγές προκειμένου να μην χαθεί καμία αλλαγή όσο ελέγχουμε μια προηγούμενη. Η διαδικασία του εντοπισμού μια αλλαγής και της αποστολή της στον server χωρίζεται σε δύο διαφορετικά νήματα προκειμένου το νήμα του watchdog να μην "κολλάει", να μην υπερφορτωθεί η λίστα του watchdog και να μην υπάρχει ο κίνδυνος να χαθεί κάποια ενημέρωση. Η προώθηση των ενημερώσεων από το ένα νήμα στο άλλο πραγματοποιείται μέσω μια ουράς τύπου FIFO όπου το watchdog ελέγχει την ενημέρωση, ελέγχει δηλαδή αν προήλθε από αλλαγή που έστειλε ο server ή αν πρόκειται για αλλαγή που έκανε ο χρήστης στο τοπικό αρχείο, και αν ανήκει στην δεύτερη περίπτωση προσθέτει στην ουρά το αντίστοιχο μήνυμα ενημέρωσης. Με την σειρά του το νήμα αποστολής αναμένει για νέα αντικείμενα στην ουρά και στέλνει την ενημέρωση στον server. Με αυτόν τον τρόπο εξοικονομείται χρόνος καθώς όσο στέλνει το νήμα αποστολής την μια ενημέρωση το watchdog ετοιμάζει την επόμενη αποστολή. Αν αυτές τις δύο

ενέργειες πραγματοποιούνταν απο ένα νήμα θα μεσολαβούσα πάντα χρόνος ελέγχου μεταξύ των αποστολών.

3.2.5 Εντοπισμός και αποστολή ενημερώσεων, client

Το watchdog είναι υπεύθυνο για τον εντοπισμό και την αξιολόγηση των αλλαγων στον φάκελο συγχρονισμού, αν δηλαδή η αλλαγή προήλθε από ενημέρωση που έκανε ο server ή από τη συσκευή.

Συγκεκριμένα στις συναρτήσεις διαχείρισης των events συμβαίνουν τα εξής:

- on_created

  Αν πρόκειται για προσωρινό αρχείο το αγνοεί, όπως επίσης αν πρόκειται για αρχείο που έχει δημιουργηθεί μετά από ενημέρωση του server, δηλαδή αν υπάρχει καταχώρηση στην βάση. Διαφορετικά δημιουργεί μήνυμα ενημέρωσης και το προσθέτει στην ουρά ενημερώσεων.

- on_modified:

  Όπως και στην προηγούμενη περίπτωση τα προσωρινά αρχεία αγνοούνται. Επίσης οι αλλαγές φακέλων και αρχείων που έχουν αλλάξει ύστερα από ενημέρωση του server αγνοούνται. Ο έλεγχος της δεύτερης περίπτωσης πραγματοποιείται με τον έλεγχο το latest_update της βάσης και του αρχείου. Αν το latest_update της βάσης είναι ίσο με -1, σημαίνει ότι πραγματοποιούνται αλλαγές την συγκεκριμένη στιγμή, ή είναι ίσα μεταξύ τους τότε το αγνοεί. Διαφορετικά δημιουργεί το μήνυμα της ενημέρωσης και το προωθεί στην ουρά ενημερώσεων.

- on_moved:

  Σε αυτή την περίπτωση αγνοούνται όσες ειδοποιήσεις αφορούν γονέα και προσωρινά αρχεία. Δηλαδή αν αλλάξει το όνομα ενός φακέλου θα δεχθεί ειδοποίηση και για τα αντικείμενα που περιέχει οι οποίες θα αγνοηθούν. Επίσης θα αγνοηθούν και όσες προέρχονται από ενημέρωση του server ελέγχοντας αν υπάρχει καταχώρηση στην βάση αντίστοιχη με το παλιό όνομα ή αν το latest_udpate του είναι ίσο με -1, αν δηλαδή πραγματοποιείται ακριβώς εκείνη την

στιγμή η διαδικασία μετονομασίας του αντικειμένου. Διαφορετικά δημιουργεί την ενημέρωση και την προσθέτει στην ουρά ενημερώσεων.

- on_deleted:

  Τέλος και σε αυτήν την περίπτωση αγνοεί τα προσωρινά αρχεία και όσα έχουν διαγραφεί. Δημιουργει μήνυμα ενημέρωσης της διαγραφής και την προσθέτει στην ουρά ενημερώσεων.

3.3 Μηνύματα που ανταλλάσσονται

3.3.1 Μορφή μηνυμάτων

Στο project εμφανίζονται τριών ειδών μηνύματα:

1. **Μηνύματα ενημέρωσης:** είναι μηνύματα που στέλνει ο client στον server όταν πραγματοποιηθεί μια αλλαγή στον φάκελο συγχρονισμού όπως για παράδειγμα μια δημιουργία αρχείου ή μια διαγραφή φακέλου. Αυτά στέλνονται σε μορφή json και περιέχουν τις απαραίτητες πληροφορίες για την ενημέρωση του server.
2. **Μηνύματα αναφοράς:** είναι μηνύματα που στέλνει ο server που δέχεται ένα μήνυμα ενημέρωσης για να αναφέρει πως εξελίχθηκε η εκτέλεση του. Αυτά τα μηνύματα έχουν μορφή κειμένου και είναι σύντομα και περιεκτικά. Για παράδειγμα όταν στέλνεται ένα μήνυμα ενημέρωσης και η εκτέλεσή του ολοκληρωθεί με επιτυχία ο παραλήπτης στέλνει το μήνυμα “completed” για να ενημερώσει τον αποστολέα πως όλα εξελίχθηκαν καλά.
3. **Μηνύματα αποστολής δεδομένων:** είναι μηνύματα μέσω των οποίων στέλνονται τα bytes των αρχείων και δεν ξεπερνούν τα 1024 bytes. Η μεταφορά ενός αρχείο από τον client στον server και αντίστροφα γίνεται σύμφωνα με την παρακάτω διαδικασία. Ο αποστολέας στέλνει το αρχείο σε πακέτα των 1024 bytes χρησιμοποιώντας τον υποδοχέα που στέλνει και τις ενημερώσεις. Ο παραλήπτης για κάθε πακέτο που λαμβάνει γράφει τα bytes του πακέτου στο τέλος του αρχείου και στέλνει μήνυμα επιβεβαίωσης. Η διαδικασία επαναλαμβάνεται μέχρι ο αποστολέας να στείλει ολόκληρο το αρχείο. Στο μήνυμα ενημέρωσης

εμπεριέχεται και το μέγεθος του αρχείου, οπότε ο παραλήπτης χρησιμοποιώντας έναν μετρητή, ο οποίος μετράει τα bytes που έχει δεχθεί, αντιλαμβάνεται με μία σύγκριση των δύο μεταβλητών πότε η διαδικασία αποστολής έχει ολοκληρωθεί.

Τέλος όλα τα μηνύματα εκτός από τα bytes των αρχείων στέλνονται με κωδικοποίηση “utf-8”.

3.3.2 Περιεχόμενο

Τα μηνύματα ενημερώσεις χωρίζονται σε πέντε βασικές κατηγορίες όπου η παράμετρος “action” λαμβάνει τιμές όπως αριθμούνται παρακάτω και συμβολίζουν την αντίστοιχη ενέργεια.

1. Μήνυμα Δημιουργίας.

   Δημιουργία φακέλου:

   ```
   {"action": 1, "path": db_path, "type": 1, "latest_update": objects_update}
   ```

   Δημιουργίας αρχείου:

   ```
   {"action": 1, "path": db_path, "type": 0, "latest_update": objects_update,
   "size": objects_size}
   ```

2. Μήνυμα Αλλαγής.

   ```
   {"action": 2, "path": db_path,"latest_update": object_update,
   "old_latest_update": old_latest_update, "size": objects_size}
   ```

3. Μήνυμα Μετονομίας.

   ```
   {"action": 3, "old_path": db_old_path, "new_path": db_new_path}
   ```

4. Μήνυμα Διαγραφής

   ```
   {"action": 4, "path": db_path, "latest_update": latest_update}
   ```

Επίσης υπάρχουν και τα μηνύματα ειδικού σκοπού με κωδικό 5 και 6 τα οποία θα παρουσιάσουμε αργότερα.

3.4 Ταυτοποίηση χρήστη και δημιουργία λογαριασμού

Η ταυτοποίηση του χρήστη κατά την είσοδό του πραγματοποιείται μέσω των credentials του, δηλαδή με το username και το password του τα οποία αποθηκεύονται στην οντότητα Users της βάσης δεδομένων του server. Το username αποτελεί κλειδί οπότε είναι μοναδικό για κάθε χρήστη. Επίσης το password πριν σταλεί από τον client στον server λαμβάνει περαιτέρω κωδικοποίηση μέσω της μεθόδου κατακερματισμού MD5 η οποία είναι διαθέσιμη στην βιβλιοθήκη hashlib της python [13]. Με αυτό τον τρόπο ο server λαμβάνει και αποθηκεύει στην βάση δεδομένων μονάχα τον κωδικοποιημένο κωδικό πράγμα που διασφαλίζει την ιδιωτικότητα. Ο χρήστης κατά την δημιουργία λογαριασμού δίνει και το email του, το οποίο χρησιμοποιείται στην περίπτωση που χρειαστεί να ανακτήσει τον κωδικό του. Τέλος κατά την επιτυχή σύνδεση του χρήστη του επιστρέφεται ένας κωδικός ο οποίος χρησιμοποιείται σαν κλειδί για την συνδεσή του στα ports υποδοχής και αποστολής ενημερώσεων προκειμένου να πιστοποιηθεί ότι αυτός που προσπαθεί να συνδεθεί στα παραπάνω ports είναι ένας συνδεδεμένος χρήστης.

3.5 Περιπτώσεις σύγκρουσης αρχείων και διαχείρισή τους

Με την έννοια σύγκρουση αρχείων εννοούμε την περίπτωση όπου ένα αρχείο αλλάζει τοπικά σε μια συσκευή αλλά η αμέσως προηγούμενη καταγεγραμμένη κατάσταση του διαφέρει από αυτή του αρχείου στον server. Καταγεγραμμένη κατάσταση ενός αρχείου στον client θεωρείται η κατάσταση που βρίσκονταν στην τελευταία αλλαγή του που στάλθηκε με επιτυχία στον server. Όταν εντοπίζεται σύγκρουση αρχείων δημιουργείται στον client ένα νέο αρχείο τύπου conf, το οποίο είναι αντίγραφο του αρχείου του client και ονομάζεται “όνομα_αρχείου.conf” και σαν πρωτότυπο κρατάμε αυτό του server. Με αυτό τον τρόπο το watchdog εντοπίζει το νέο αρχείο και στέλνει μήνυμα δημιουργίας στον server.

3.6 Ενημέρωση κατά την είσοδο συσκευής

Μετά την επιτυχής σύνδεση μιας συσκευής ξεκινά η διαδικασία της ενημέρωσης τόσο του client όσο και του server. Η ενημέρωση πραγματοποιείται με μία συγκεκριμένη δομή η οποία μειώνει τα περιττά μηνύματα και τους ελέγχους που ακολουθούν. Η δομή αυτή αποτελείται από τα παρακάτω τμήματα τα οποία εκτελούνται με την σειρά που παρουσιάζονται.

1. Ο server στέλνει όσα renamed έχουν πραγματοποιηθεί από την τελευταία ενημέρωση της συσκευής και αργότερα.
2. Ο client στέλνει όλα τα updates και τα creates που έχουν πραγματοποιηθεί όσο ήταν offline.
3. Ο server στέλνει updates και deletes που έχουν πραγματοποιηθεί από την τελευταία ενημέρωση της συσκευής και μετά.
4. Ο client στέλνει όσα deletes πραγματοποιήθηκαν όσο ήταν offline.

Η ενημέρωση κατά αυτόν τον τρόπο επιτυγχάνεται με την χρήση σηματοφόρων τόσο στον client όσο και στον server. Συγκεκριμένα ο client πριν ξεκινήσει να στέλνει οτιδήποτε περιμένει ο server να του στείλει μήνυμα ενημέρωσης τύπου 5, (action:5), το οποίο σημαίνει ότι έχουν σταλεί όλα τα renames. Το νήμα αποστολής περιμένει να “κλειδώσει” τον αντιστοιχο σηματοφόρο τον οποίο “ξεκλειδώνει” το νήμα υποδοχής ενημερώσεων μόλις δεχθεί το κατάλληλο μήνυμα. Στην συνέχεια όσο ο client ξεκινά να στέλνει τα updates και τα creates ο server περιμένει και αυτός με την σειρά του να στείλει ο client μήνυμα ενημέρωσης τύπου 5, που σημαίνει ότι ο client ολοκλήρωσε την αποστολή των updates και creates. Κατά τον ίδιο τρόπο όπως στην περίπτωση του client ο server μόλις δεχθεί το μήνυμα “ξεκλειδώνει” τον αντίστοιχο σηματοφόρο και ξεκινά την ενημέρωση του client. Τέλος ο client περιμένει να λάβει μήνυμα ενημέρωσης τύπου 6 προκειμένου να στείλει τα deletes του όσο ήταν offline και στην συνέχεια οτιδήποτε άλλο προκύψει όσο είναι online. Μόλις ολοκληρώσει την ενημέρωση ο server στέλνει τα μήνυμα ενημέρωσης τύπου 6. Κατά την διάρκεια της ενημέρωσης τόσο η ουρά αποστολής του client όσο και η ουρά αποστολής του server προς την συγκεκριμένη συσκευή δέχονται νεότερες

ενημερώσεις. Επομένως όσα γίνονται στο διάστημα ενημέρωσης καταγράφονται και στέλνονται μόλις αυτή ολοκληρωθεί.

3.7 Κρυπτογράφηση μηνυμάτων που ανταλλάσσονται

Προκειμένου να διασφαλιστεί η ιδιωτικότητα των δεδομένων τα οποία στέλνουν και δέχονται οι χρήστες χρειάζεται τα μηνύματα που ανταλλάσουν να προστατεύονται από τρίτους. Η κρυπτογράφηση είναι ένας από τους πιο συνηθισμένους τρόπους προστασίας ευαίσθητων δεδομένων Το πρότυπο AES (Advanced Encryption Standard) χρησιμοποιεί κρυπτογράφηση συμμετρικού κλειδιού, η οποία περιλαμβάνει τη χρήση ενός μυστικού κλειδιού για την κρυπτογράφηση και την αποκρυπτογράφηση πληροφοριών [13]. Επίσης σε συνδυασμό με τη συνάρτηση κατακερματισμού SHA-256 (Secure Hash Algorithm) δημιουργούν ένα αρκετά ασφαλές δίκτυο επικοινωνίας. Ο εξυπηρετητής δημιουργεί ένα μοναδικό κλειδί για κάθε χρήστη που πραγματοποίησε επιτυχημένη είσοδο και του το στέλνει σαν απάντηση στο αίτημα σύνδεσης. Ο χρήστης χρησιμοποιεί αυτό το κλειδί για δύο σκοπούς, πρώτον σαν κλειδί επιβεβαίωσης όταν συνδέεται στα ports αποστολής και αποδοχής ενημερώσεων, προκειμένου ο εξυπηρετητης να διασταυρώσει ότι αυτός που συνδέεται σε αυτά τα ports είναι ένας συνδεδεμένος χρήστης και δεύτερον σαν κλειδί κωδικοποίησης. Στην δεύτερη περίπτωση δεν θα χρησιμοποιήσει αυτό καθ' αυτό το κλειδί, αλλά θα χρησιμοποιήσει αυτό που προκύπτει από τον κατακερματισμό του. Επομένως ο χρήστης λαμβάνει το κλειδί του το συμπεριλαμβάνει στα μηνύματα ταυτοποίησης, το χρησιμοποιεί σαν κλειδί κρυπτογράφηση και στέλνει κρυπτογραφημένα τα μηνύματα ταυτοποίησης στα port υποδοχής και αποστολής ενημερώσεων. Όταν, τέλος, η ταυτοποίηση πραγματοποιηθεί με επιτυχία όλα τα μηνύματα μεταξύ εξυπηρετητή και πελάτη στέλνονται χρησιμοποιώντας αυτή την κρυπτογράφηση. Ο εξυπηρετητής απο την πλευρά του κρατάει για κάθε συνδυασμό username, MAC address, με τον οποίο έχει πραγματοποιηθεί επιτυχής σύνδεση, το κλειδί που έχει στείλει και όταν δεχθεί μήνυμα ταυτοποίησης στα ports υποδοχής και αποστολής ελέγχει αν ταιριάζει το κλειδί με κάποιο από αυτά. Αν ναι στέλνει μήνυμα επιβεβαίωσης και ξεκινούν οι διαδικασίες αποστολής και παραλαβής ενημερώσεων.

3.8 Ενέργειες συντήρησης και αποκατάστασης

Υπάρχουν περιπτώσεις που για διάφορους λόγους μια συσκευή μπορεί να αποσυνδεθεί κατά την διάρκεια αποστολής ή υποδοχής μιας ενημέρωσης. Αυτό μπορεί να προκαλέσει πρόβλημα τόσο στα αρχεία του server όσο και στου client. Ο client αφού συνδεθεί και επιβεβαιωθούν τα διαπιστευτήρια του διαγράφει όσα αντικείμενα ήταν υπό επεξεργασία ύστερα από ενημέρωση που έλαβε από τον server όταν διακόπηκε η σύνδεση, δηλαδη σαρώνει την βάση δεδομένων του και όσα αντικείμενα έχουν latest_update ίσο με -1 τα διαγράφει, διότι τα αρχεία αυτά θα αποσταλούν ξανά. Ο server προκειμένου να προστατεύσει τα αρχεία του από κάποια αποτυχημένη ενημέρωση, αποθηκεύει πρώτα τα αρχεία σε ένα προσωρινό φάκελο και με το που ολοκληρωθεί η μεταφορά των bytes αντικαθιστά το αρχείο στο φάκελο του χρήστη με το προσωρινό αρχείο. Αν για κάποιο λόγο διακοπεί η σύνδεση με τον client κατά το “ανέβασμα” ενός αρχείου, τότε παραλείπει την ενημέρωση και διατηρεί στο φάκελο του χρήστη το αμέσως προηγούμενο αρχείο. Επίσης μετά την αποσύνδεση μια συσκευής του χρήστη ο server ελέγχει αν τα αντικείμενα της βάσης τα οποία έχουν υποστεί διαγραφή η μετονομασία έχουν ενημερωθεί σε όλες του τις συσκευές και σε περίπτωση που αυτό ισχύει διαγράφει τις αντίστοιχες καταχωρήσεις.

3.9 Σενάριο λειτουργίας συστήματος

Προκειμένου να κατανοηθεί καλύτερα η επικοινωνία μεταξύ server και client θα αναπτύξουμε το παρακάτω σενάριο. Τα μηνύματα που ανταλλάσσονται εμφανίζονται στην μορφή που έχουν πριν την κρυπτογράφηση. Έστω ότι συνδέεται ένας χρήστης από μια συσκευή από την οποία έχει ξανασυνδεθεί στο παρελθόν.

Αρχικά θα συνδεθεί στο port 8001 και θα στείλει τα διαπιστευτήρια του ώστε να κάνει login.

```
client,8001->{"action": "1", "username": "novas",
"password":"e10adc3949ba59abbe56e057f20f883e",
"mac": "581653591521O4"}
```

```
server:8001->"RL0C0VX3TO1JZ3Y5"
```

Παρατηρούμε ότι ο κωδικός στέλνεται κωδικοποιημένος και ότι η τιμή του action είναι l (login). Τα διαπιστευτήρια ταιριάζουν οπότε ο server στέλνει το κλειδί.

Στην συνέχεια ξεκινάει η ενημέρωση του client και του server. Ο client συνδέεται στο port 8002 στο οποίο θα λαμβάνει ενημερώσεις και στο port 8003 στο οποίο θα στέλνει ενημερώσεις. Επίσης στέλνει και στις δύο συνδέσεις το username του και το mac address της συσκευής και το κλειδί ταυτοποίησης.

```
client:8002->{"users_pass":"RL0C0VX3TO1JZ3Y5",
"username": "novas", "mac": "58165359152104"}


client:8003->{"users_pass":"RL0C0VX3TO1JZ3Y5",
"username": "novas", "mac": "58165359152104"}

server:8003->"completed"
```

Στην περίπτωση του port 8002 δεν χρειάζεται να λάβει επιβεβαίωση καθώς αρκεί απλά να περιμένει ενημερώσεις εκεί ενώ στο port 8003 πρέπει να λάβει επιβεβαίωση ότι ο server είναι έτοιμος προκειμένου να αρχίσει να στέλνει ενημερώσεις. Η συγκεκριμένη σειρά αποστολής από πλευράς client δεν είναι προκαθορισμένη, αλλά είναι η επικρατέστερη, διότι πραγματοποιείται από διαφορετικά νήματα (παραλαβής και αποστολής ενημερώσεων) από τα οποία το πρώτο (παραλαβής) προηγείται χρονικά του δεύτερου (αποστολής). Σε περίπτωση που δεν ταυτοποιηθεί ο χρήστης στο port 8003

επιστρέφεται μήνυμα με περιεχόμενο “not access” και διακόπτεται η σύνδεση ενώ στο port 8002 διακόπτεται απευθείας από τον εξυπηρετητή. Όπως έχουμε δει προηγουμένως η ενημέρωση πραγματοποιείται σε τέσσερις φάσεις. Αρχικά ο server στέλνει τις μετονομασίες που πραγματοποιήθηκαν όσο ο client ήταν offline. Στο συγκεκριμένο παράδειγμα δεν πραγματοποιήθηκε κάποια μετονομασία οπότε στέλνει απευθείας μήνυμα ότι η φάση αυτή ολοκληρώθηκε.

```
server:8002->{"action":5}

client:8002->"completed"
```

Στην δεύτερη φάση ο client στέλνει όλα τα updates και τα creates που πραγματοποιήθηκαν όσο ήταν offline.

```
client:8003->{"action":1,
"path":"novas/documents/test.py","type":0,
"latest_update": 1615995275.0, "size": 361}
server:8003->"send bytes"
client:8003->b"nvlfnvsl..."
server:8003->"ok"
server:8003->"completed"
client:8003->{"action": 5}
server:8003->"completed"
```

Όπως εύκολα μπορούμε να διακρίνουμε είχε δημιουργηθεί ένα αρχείο test.py στον φάκελο documents, ο client στέλνει το μήνυμα ενημέρωσης ο server ζητά τα bytes που το αποτελούν και τότε ο client ξεκινά την αποστολή του αρχείου. Στέλνει τα bytes από την θέση μηδέν του αρχείου και μετά. Το αρχείο είναι μικρότερο από 1024 bytes οπότε στέλνεται με ένα μήνυμα δεδομένων, δέχεται επιβεβαίωση και ενημερώνει ότι ολοκληρώθηκε το δεύτερο στάδιο. Αξίζει να σημειωθεί ότι όλα τα μηνύματα του παραπάνω παραδείγματος στέλνονται στον ίδιο socket επικοινωνίας, δεν χρησιμοποιείται κάποια πρόσθετο socket για την μεταφορά του αρχείου. Στην τρίτη φάση της ενημέρωσης ο server στέλνει creates, updates και deletes.

```
server:8002->{"action":1,"path":"novas/ticket.pdf",
"type": 0,"latest_update": 1620129803.7560327,
"size": 263162}
client:8002->"send bytes"
server:8002->b"nvlfncsfvsfvsvsvdsv..."
client:8002->"ok"
server:8002->b"nvlfnvsvdevavdvssv..."
client:8002->"ok"
```

```
server:8002->b"nvlfdevavdvssv..."
client:8002->"ok"
client:8002->"completed"
server:8002->{"action": 6 }
client:8002->"completed"
```

Στο παράδειγμα φαίνεται ότι μία άλλη συσκευή είχε δημιουργήσει το αρχείο με όνομα ticket.pdf στον βασικό φάκελο ενώ δεν πραγματοποιήθηκε κάποια διαγραφή. Η διαδικασία αποστολής του αρχείου δεν εμφανίζεται ολόκληρη για λόγους συντόμευσης. Τέλος στην τέταρτη φάση ο client στέλνει τα deletes.

```
client:8003->{"action":4,
"path":"novas/documents/XeroxScan.PDF",
"latest_update": 1620134178.6272762}
server:8003->"completed"
```

Σε αυτή την φάση δεν χρειάζεται να στείλει ο client μήνυμα ότι τελείωσε με την διαδικασία. Αν κάποια άλλη συσκευή πραγματοποιήσει μια αλλαγή σε αρχείο το οποίο έχει διαγραφεί στην συγκεκριμένη συσκευή τότε όποια ενημέρωση και αν προηγηθεί (διαγραφή ή αλλαγή) θα επικρατήσει η αλλαγή. Αφού έχουν ολοκληρωθεί τουλάχιστον τα πρώτα τρία στάδιο της ενημέρωσης ο server στέλνει αλλαγές που πραγματοποιήθηκαν από άλλες συσκευές είτε κατά την διάρκεια της ενημέρωσης είτε σε πραγματικό χρόνο.

3.10 Γραφικό περιβάλλον χρήστη

Για την δημιουργία του GUI (Graphical user interface) χρησιμοποιήθηκε το πιο δημοφιλές εργαλείο για δημιουργία γραφικού περιβάλλοντος που προσφέρει η python το tkinter. Με το tkinter μπορούν να δημιουργηθεί εύκολα και γρήγορα ένα φιλικό προς τον χρήστη γραφικό περιβάλλον. Ακόμη το tkinter έχει το πλεονέκτημα ότι υποστηρίζεται από τις περισσότερες πλατφόρμες Unix όπως επίσης και απο τα Windows [16].

3.11 Δημιουργία εκτελέσιμου αρχείου και αρχείου setup

Για την δημιουργία του εκτελέσιμου αρχείου αξιοποιήθηκε ακόμα ένα εργαλείο γραμμένο σε python το pyinstaller. Το pyinstaller παίρνει σαν είσοδο ένα αρχείο python και όλες τις εξαρτήσεις του και δημιουργεί ένα εκτελέσιμο αρχείο. Ο χρήστης πλέον χωρίς να χρειάζεται τον αρχικό κώδικα μπορεί να χρησιμοποιήσει την εφαρμογή απλά εκτελώντας το εκτελέσιμο αρχείο. Στην συνέχεια προκειμένου να δημιουργήσουμε ένα

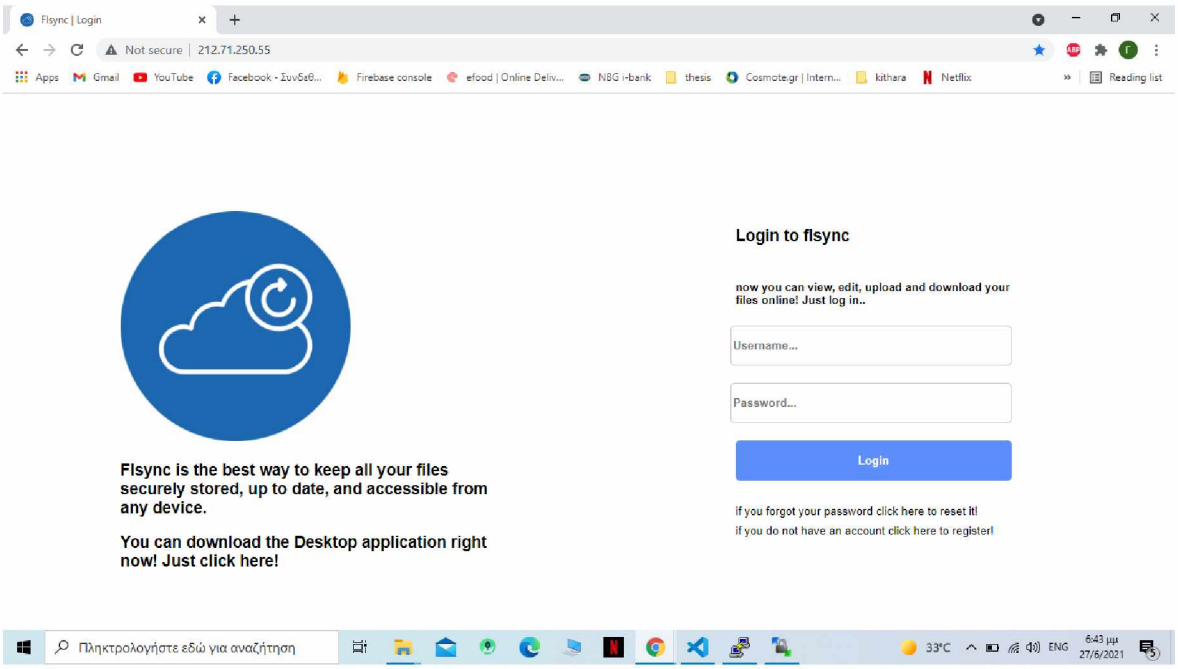
αρχείο setup χρησιμοποιήσαμε το πρόγραμμα NSIS (Nullsoft Scriptable Install System) το οποία λαμβάνει σαν είσοδο τα αρχεία που χρειάζεται να περιλαμβάνονται στον φάκελο της εφαρμογής και δημιουργεί ένα setup αρχείο. Συγκεκριμένα χρησιμοποιήθηκε η υπηρεσία zip2exe, η οποία λαμβάνει σαν είσοδο ένα .zip αρχείο [23].

ΚΕΦΆΛΑΙΟ 4

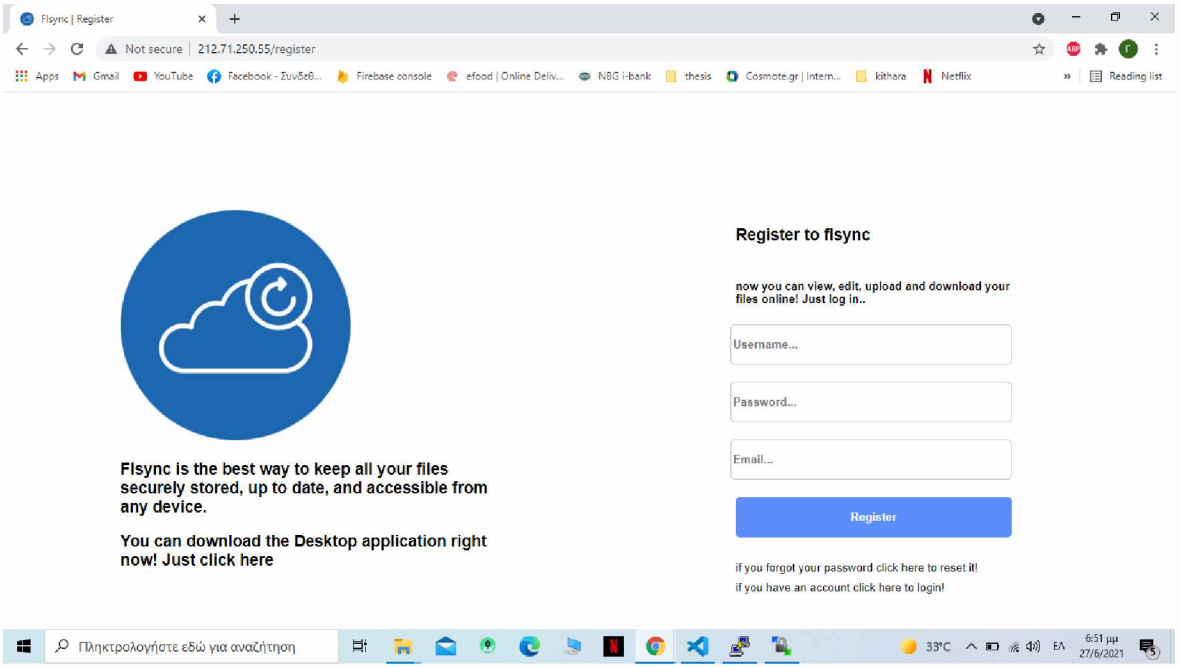
ΛΕΙΤΟΥΡΓΙΚΗ ΠΕΡΙΓΡΑΦΗ ΤΟΥ WEB-INTERFACE

4.1 Είσοδος με credentials (login και register)

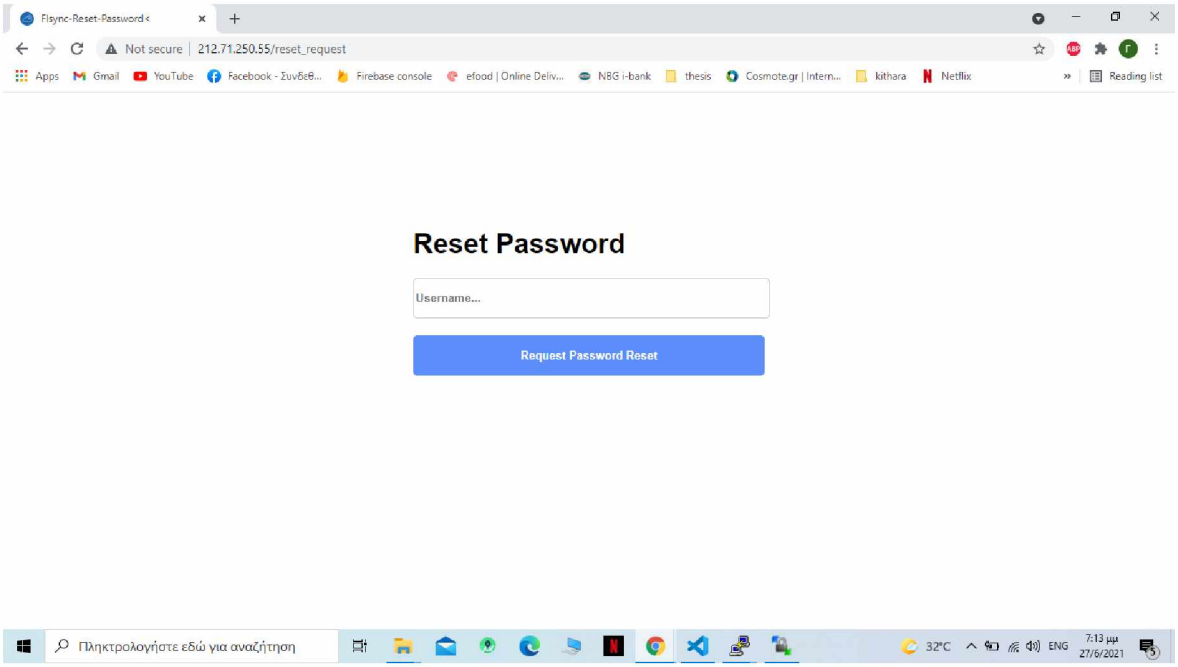
Ο χρήστης καλείται να συνδεθεί στην ιστοσελίδα της εφαρμογής μέσω των διαπιστευτηρίων του, δηλαδή με το username και το password του. Επίσης σε περίπτωση που δεν διαθέτει λογαριασμό έχει την δυνατότητα δημιουργίας νέου. Εφόσον δεν κάνει αποσύνδεση, τις επόμενες φορές που θα συνδεθεί από τον ίδιο browser δεν θα χρειαστεί να δώσει ξανά τα credentials του αλλά θα παραμείνει συνδεδεμένος. Τέλος έχει την δυνατότητα ανάκτησης του κωδικού πρόσβασης μέσω email. Δίνοντας δηλαδή το username του θα του αποσταλεί ένα email το οποίο θα περιέχει ένα link για να ενημερώσει το password του.



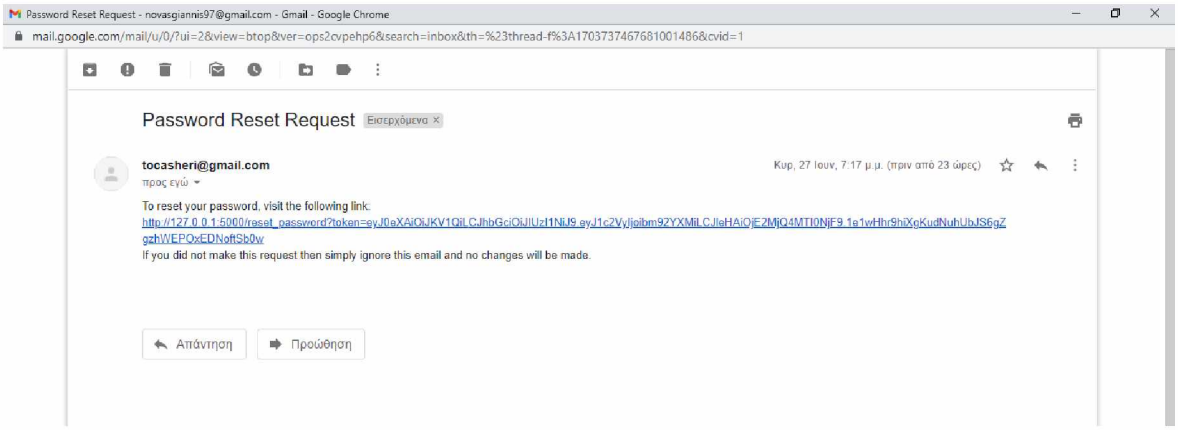
Εικόνα 4.1: Σελίδα Download Desktop app-Login



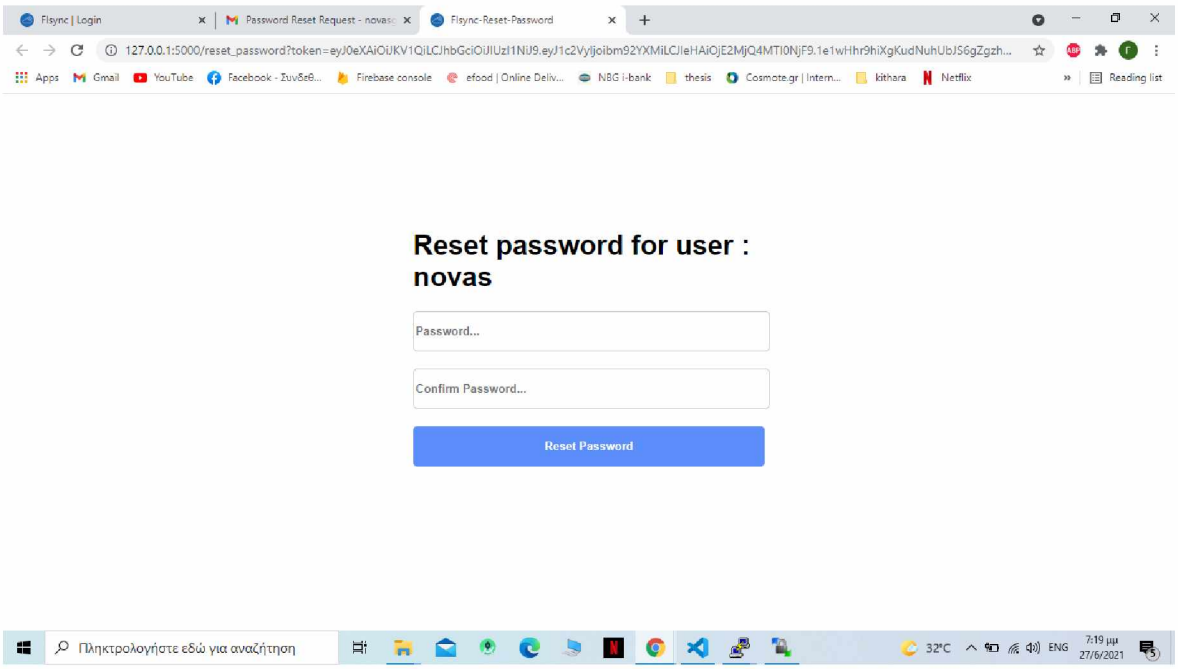
Εικόνα 4.2: Σελίδα Download Desktop app-Register



Εικόνα 4.3: Reset password request



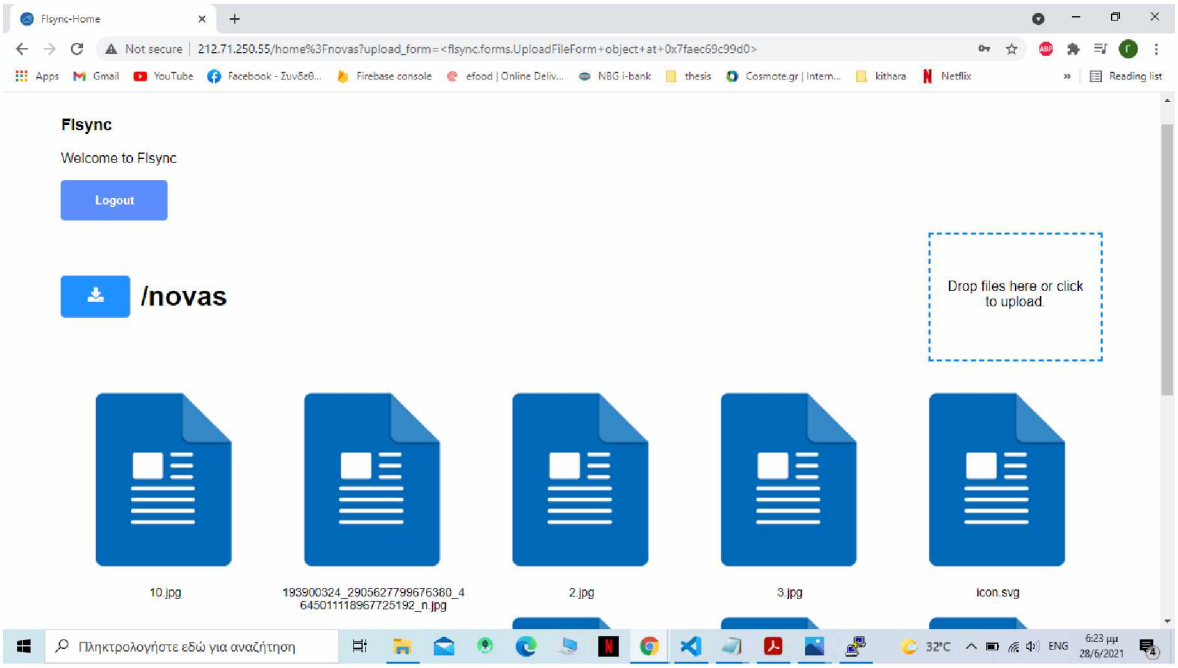
Εικόνα 4.4: Email με σύνδεσμο αλλαγής password



Εικόνα 4.5: Σελίδα αλλαγής password

4.2 Δυνατότητες του web-interface

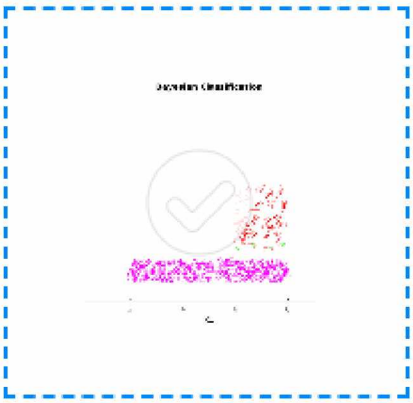
Αφού ο χρήστης συνδεθεί με επιτυχία μπορεί να περιηγηθεί στον φάκελο συγχρονισμού και στους υποφακέλους του. Οι φάκελοι και τα αρχεία εμφανίζονται με το αντίστοιχο εικονίδιο. Επίσης έχει την δυνατότητα να ανεβάσει, να κατεβάσει και να ενημερώσει αρχεία και φακέλους όπως θα δούμε αναλυτικότερα.



Εικόνα 4.6: Κεντρική σελίδα του web-interface

4.2.1 "Ανέβασμα" αρχείου και φακέλου

Στο πάνω δεξιά μέρος της οθόνης σχηματίζεται με διακεκομμένες μπλε γραμμές ένα ορθογώνιο το οποίο εξυπηρετεί το ανέβασμα αρχείων και φακέλων. Ο χρήστης έχει την δυνατότητα να ανεβάσει το αντικείμενο που επιθυμεί μέσω drag and drop, σέρνοντας δηλαδή αντικείμενα μέσα στον νοητό κουτί, ή κάνοντας απλώς click μέσα στο κουτί και επιλέγοντας από το filesystem του. Σε περίπτωση επιτυχίας εμφανίζεται η αντίστοιχη ένδειξη και σε λίγα δευτερόλεπτα αδειάζει πάλι το κουτί. Σε αντίθετη περίπτωση εμφανίζεται ένδειξη αποτυχίας με το κατάλληλο μήνυμα. Επίσης υπάρχει η δυνατότητα πολλαπλής επιλογής αρχείων για ανέβασμα, είτε επιλέγοντας κάθε αρχείο ξεχωριστά κατά την διαδικασία επιλογής από το filesystem, είτε ανεβάζοντας ένα συμπιεσμένο αρχείο τύπου .zip, όπου στην συγκεκριμένη περίπτωση δημιουργούνται και οι φάκελοι και οι υποφάκελοι όπως ακριβώς υπάρχουν μέσα στο συμπιεσμένο αρχείο. Οι τύποι αρχείων που υποστηρίζονται είναι κάθε τύπος αρχείου κειμένου, εικόνας ή βίντεο ενώ για συμπιεσμένα αρχεία υποστηρίζεται μόνο η μορφή .zip.



Εικόνα 4.7: Ανέβασμα αρχείου

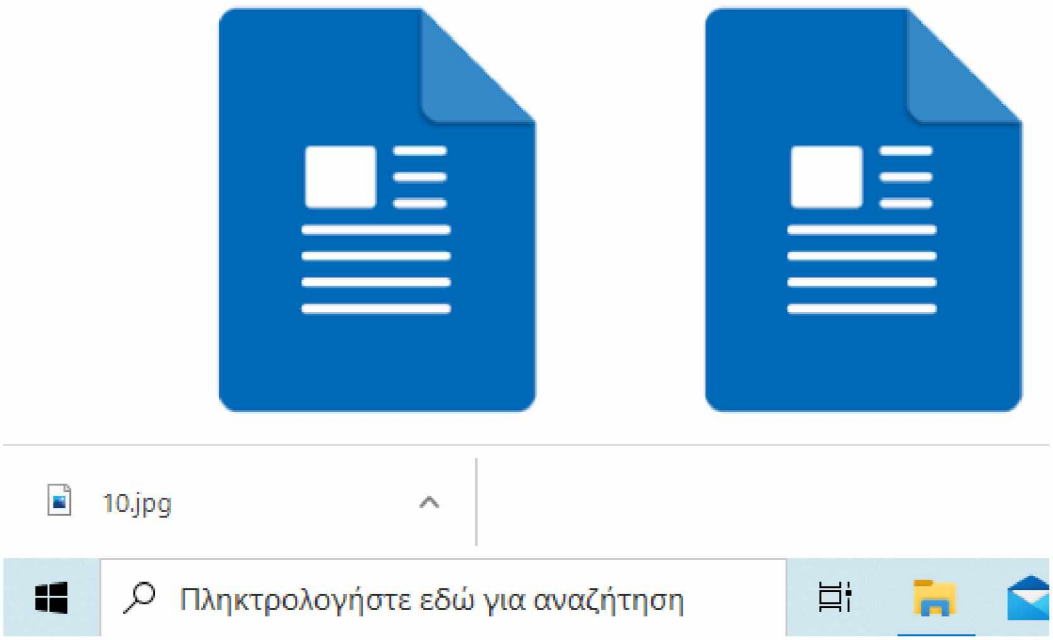
4.2.2 Ενημέρωση υπάρχοντος αρχείου και φακέλου

Ο χρήστης ανεβάζοντας ένα αρχείο με πλήρες όνομα αρχείου ίδιο με κάποιου προϋπάρχων αντικαθιστά το παλιό αρχείο με αυτό που ανεβάζει. Επίσης αν συμβεί το ίδιο με κάποιο φάκελο τότε θα προστεθούν στον φάκελο τα καινούρια αρχεία και φάκελοι και θα συμβεί αντικατάσταση όσων υπήρχαν ήδη. Δηλαδή για παράδειγμα αν ο χρήστης θέλει να ενημερώσει ένα project αρκεί να ανεβάσει ένα συμπιεσμένο αρχείο που περιλαμβάνει το project με αποτέλεσμα όσα αρχεία έχουν τροποποιηθεί να ενημερωθούν και όσα δεν υπήρχαν να δημιουργηθούν.

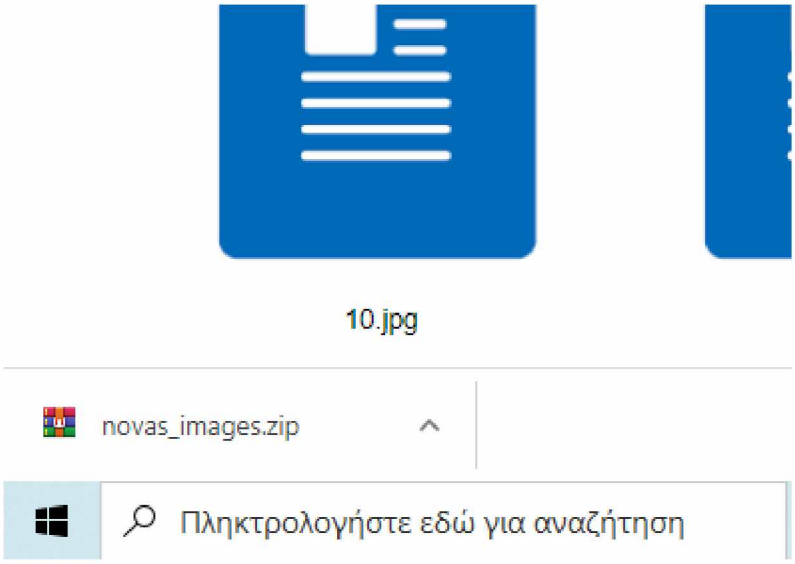
4.2.3 "Κατέβασμα" αρχείου και φακέλου

Ακόμη ο χρήστης έχει την δυνατότητα να κατεβάσει αρχεία και φακέλους μέσω του web-interface. Συγκεκριμένα μπορεί να κατεβάσει ένα αρχείο απλά επιλέγοντας το με ένα click στο εικονίδιο του αρχείου. Επίσης όπως φαίνεται στα στιγμιότυπα που ακολουθούν το όνομα του φακέλου στον οποίο βρίσκεται ο χρήστης εμφανίζεται πάνω αριστερά όπως

επίσης αριστερά του ονόματος βρίσκεται ένα κουμπί τύπου download. Κάνοντας click σε αυτό το κουμπί πραγματοποιείται download ολόκληρου του φακέλου σε μορφή συμπιεσμένου αρχείου τύπου .zip.



Εικόνα 4.8: Κατέβασμα αρχείου



Εικόνα 4.9: Κατέβασμα φακέλου με την μορφή συμπιεσμένου αρχείου

ΚΕΦΆΛΑΙΟ 5

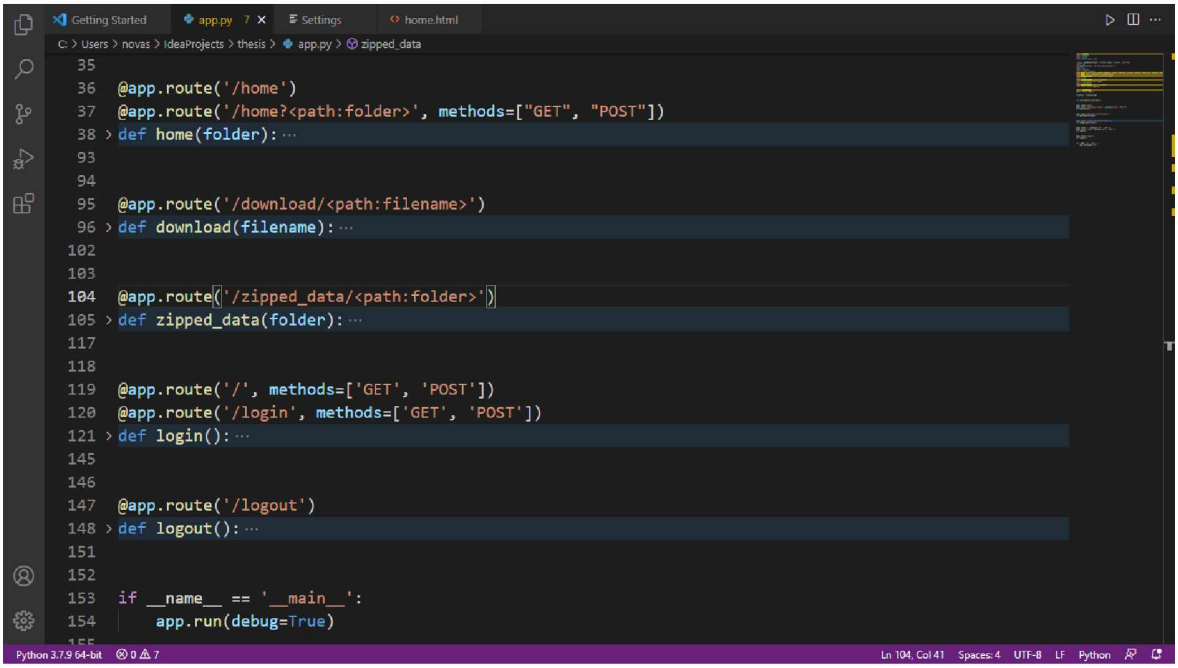
ΤΕΧΝΙΚΗ ΠΕΡΙΓΡΑΦΗ ΚΑΙ SETUP ΤΟΥ WEB-INTERFACE

5.1 Flask

Το Flask [1] είναι ένα micro web framework WSGI γραμμένο σε python. Κατηγοριοποιείται ως microframework επειδή δε δεσμεύεται από συγκεκριμένα εργαλεία και βιβλιοθήκες. Αυτό δίνει την δυνατότητα στον προγραμματιστή να δημιουργήσει απλές ιστοσελίδες, ενώ δεν περιορίζει τη δυνατότητα επέκτασής τους αξιοποιώντας ανεξάρτητες βιβλιοθήκες και εργαλεία της python. Δεν διαθέτει κάποια προεπιλεγμένη βάση δεδομένων, πράγμα που εξυπηρετεί ιδιαίτερα το project της συγκεκριμένης εργασίας καθώς κάνει την πρόσβαση στην βάση αρκετά ευέλικτη.

Πρωτοεμφανίστηκε προκειμένου να ενώσει τη μηχανή προτύπου jinja και την εργαλειοθήκη Werkzeug WSGI αλλά τελικά εξελίχθηκε σε ένα από τα δημοφιλέστερα framework της python ενώ σύμφωνα με το Github βρίσκεται δεύτερο στις προτιμήσεις μετά το Django και ψηφίστηκε σαν το πιο δημοφιλές web framework στο Python Developers Survey του 2018.

Το Flask επιλέχθηκε για την υλοποίηση του web-interface του project καθώς οι δυνατότητες του σε συνδυασμό με την απλότητα του εξυπηρετούν ακριβώς τις απαιτήσεις της εφαρμογής. Στο αρχείο routes.py ορίζονται οι συναρτήσεις οι οποίες είναι υπεύθυνες για την απόκριση του συστήματος μας ανάλογα με το URL που ορίζεται στο root( ).



Εικόνα 5.1: Μέρος του αρχείου route.py

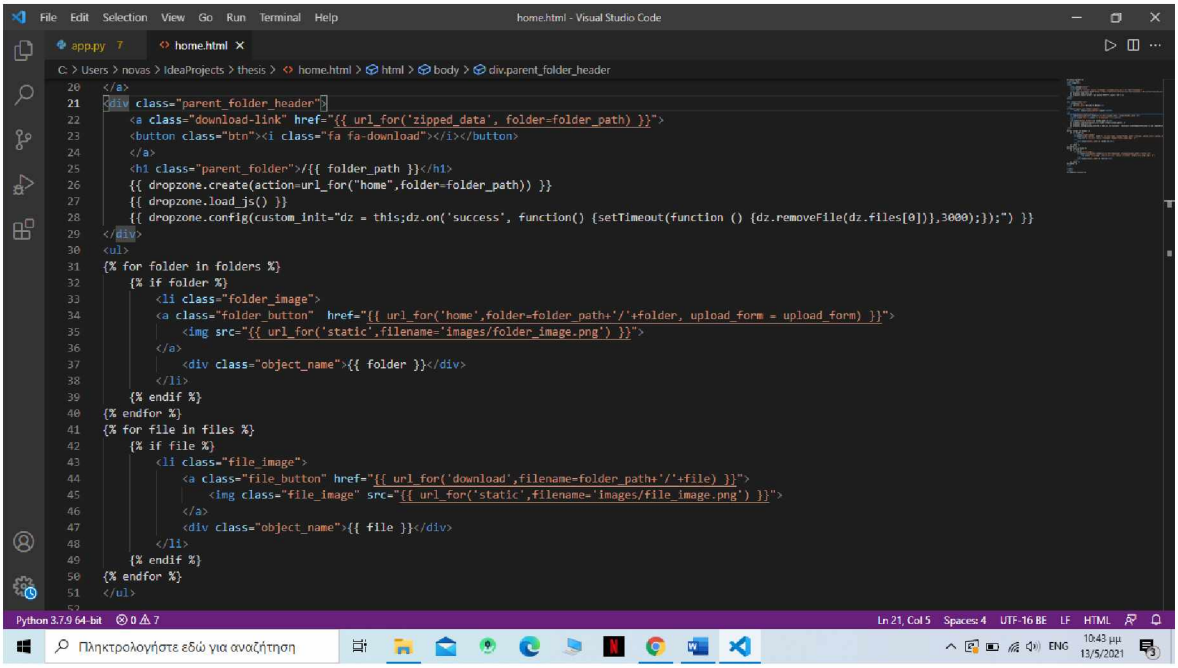
5.2 HTML5, CSS3, jinja2

Η HTML (Hyper Text Markup Language) είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες, και τα στοιχεία της είναι τα βασικά δομικά στοιχεία των ιστοσελίδων. Η πιο πρόσφατη έκδοσή της είναι η HTML5 η οποία χρησιμοποιείται στο web-interface του project. Τα πλεονεκτήματα της νεότερης έκδοσης είναι το πλήθος νέων στοιχείων που περιέχει, τα οποία δίνουν την δυνατότητα στον προγραμματιστή να περιγράψει με μεγαλύτερη ακρίβεια το περιεχόμενο των αντικειμένων που θέλει να απεικονίσει, καθώς και η συμβατότητα της με όλα τα προγράμματα περιήγησης ιστού ανεξαρτήτως συσκευής. Ο σκοπός ενός web browser είναι να διαβάζει τα έγγραφα HTML και να τα συνθέσει σε σελίδες που μπορεί κανείς να διαβάσει ή να ακούσει [2]. Οι Web browsers μπορούν επίσης να αναφέρονται σε στυλ μορφοποίησης CSS για να ορίζουν την εμφάνιση και τη διάταξη του κειμένου και του υπόλοιπου υλικού.

Η τεχνολογία πίσω από τη μορφοποίηση κάθε ιστοσελίδας ονομάζεται CSS (Cascading Style Sheet). Πρόκειται για μια γλώσσα, η οποία ορίζει δομές μορφοποίησης όπως γραμματοσειρές, χρώματα και θέσεις. Οι μορφοποιήσεις CSS μπορούν να αποθηκευτούν απευθείας σε μια ιστοσελίδα HTML ή σε ένα ξεχωριστό αρχείο. Σε κάθε περίπτωση οι εντολές CSS περιέχουν κανόνες, οι οποίοι εφαρμόζονται στα στοιχεία ενός συγκεκριμένου

τύπου. Όταν χρησιμοποιούνται εξωτερικά, οι κανόνες μορφοποίησης τοποθετούνται σε ένα εξωτερικό αρχείο με την κατάλληξη ονόματος αρχείου .css. Ένας κανόνας είναι μια οδηγία μορφοποίησης, που μπορεί να εφαρμοστεί σε ένα στοιχείο ιστοσελίδας, όπως σε μια παράγραφο κειμένου ή σε μια σύνδεση. Οι κανόνες αποτελούνται από μία ή περισσότερες ιδιότητες στυλ και τις σχετικές τιμές τους. Για μια όμορφη και καλοσχεδιασμένη ιστοσελίδα η χρήση της CSS κρίνεται απαραίτητη. Η CSS3 είναι η νεότερη έκδοση της CSS. Το CSS βοηθά τους προγραμματιστές ιστού να δημιουργήσουν μια ομοιόμορφη εμφάνιση σε πολλές σελίδες μιας τοποθεσίας Web. Αντί να ορίζει το στυλ κάθε πίνακα και κάθε μπλοκ κειμένου μέσα σε HTML μιας σελίδας, τα στυλ που χρησιμοποιούνται συνήθως πρέπει να προσδιορίζονται μόνο μία φορά σε ένα έγγραφο CSS. Επιπλέον, το CSS διευκολύνει την αλλαγή στυλ σε πολλές σελίδες ταυτόχρονα [3].

Ο τελικός παράγοντας που ολοκληρώνει το front-end της ιστοσελίδας της εφαρμογής είναι το jinja2, μια σύγχρονη γλώσσα προτύπων για προγραμματιστές της Python. Χρησιμοποιείται για τη δημιουργία HTML, XML ή άλλων μορφών σήμανσης που επιστρέφονται στο χρήστη μέσω αιτήματος HTTP. Το jinja2 ικανοποιεί την ανάγκη των προγραμματιστών να δημιουργούν δυναμικές ιστοσελίδες με τη χρήση της python. Μέσω του jinja2 επεκτείνουμε το πρότυπο αρχείο HTML και αξιοποιούμε τις παραμέτρους που δεχόμαστε σαν είσοδο για την εμφάνιση δυναμικών δεδομένων στον χρήστη όπως λίστες πίνακες και μεταβλητές που προσδιορίζονται ανάλογα με τις απαιτήσεις του χρήστη. Στο interface της εργασίας χρησιμοποιείται το jinja2 για να εμφανιστούν το όνομα του τρέχοντος φακέλου, οι φάκελοι και τα αρχεία που περιλαμβάνει καθώς επίσης και η αρχικοποίηση του dropzone.



Εικόνα 5.2: Μέρος του αρχείου home.html

5.3 Αναλυτική τεχνική περιγραφή back-end

5.3.1 Login, Logout

Η διαδικασία του login πραγματοποιείται με την χρήση φόρμας. Όταν ο χρήστης έχει συμπληρώσει το username και το password του, δηλαδή όταν η φόρμα, έχει συμπληρωθεί γίνεται ο έλεγχος των στοιχείων μέσω της μεθόδου post. Επιλέχθηκε η μέθοδος post αντί της get καθώς η πρώτη είναι καταλληλότερη για μεταφορά ευαίσθητων δεδομένων όπως κωδικοί και επίσης τα δεδομένα που αποστέλλονται δεν αποθηκεύονται στο ιστορικό και ούτε εμφανίζονται στην διεύθυνση URL, πράγμα που κάνει την χρήση της ακόμα ασφαλέστερη. Η ταυτοποίηση πραγματοποιείται με τον ίδιο τρόπο που πραγματοποιείται στην εφαρμογή του client. Δημιουργείται σύνδεση μέσω socket με τον server στο port 8001 και ανάλογα με την απάντηση του κρίνεται η επιτυχία εισαγωγής. Σε περίπτωση που ο χρήστης συνδεθεί με επιτυχία δημιουργείται ένα session με κλειδί το username και τιμή το όνομα χρήστη. Το session αποθηκεύει πληροφορία τόσο στην πλευρά του client όσο και στην πλευρά του server. Επίσης η πληροφορία και στις δύο πλευρές αποθηκεύεται κωδικοποιημένη. Αυτό δίνει την δυνατότητα στον προγραμματιστή να μην ζητάει από τον χρήστη, κάθε φορά που ο δεύτερος κλείνει τον browser του ή το φύλλο περιήγησής του,

τα διαπιστευτήρια του αλλά αντίθετα να τον κρατά συνδεδεμένο μέχρι να αποφασίσει να κάνει αποσύνδεση. Οπότε ελέγχοντας αν υπάρχει το session με κλειδί username δίνει απευθείας πρόσβαση στο home της εφαρμογής. Όταν ο χρήστης κάνει logout το session με κλειδί username διαγράφεται.

5.3.2 Home, upload, download

Στην κεντρική σελίδα του web-interface εμφανίζονται, όπως έχουμε δει, το όνομα του φακέλου που βρισκόμαστε, οι φάκελοι και τα αρχεία που τον απαρτίζουν και το πλαίσιο του upload. Είναι πρακτικά η σελίδα στην οποία λαμβάνουν χώρα όλες οι δραστηριότητες του χρήστη. Η εμφάνιση των στοιχείων της σελίδας πραγματοποιείται δυναμικά. Ο web-server συνδέεται με την βάση δεδομένων και ανάλογα με το όνομα του φακέλου λαμβάνει τις αντίστοιχες πληροφορίες, δηλαδή τα ονόματα των φακέλων και των αρχείων που βρίσκονται στο φάκελο. Η εμφάνιση τους στο αρχείο html πραγματοποιείται με την χρήση jinja2.

Η λειτουργία upload πραγματοποιείται με τη χρήση φόρμας και της μεθόδου post. Όταν ο χρήστης σύρει ή επιλέξει ένα αρχείο στο πλαίσιο του upload στέλνει ένα αίτημα Http μέσω POST .Ο web-server ελέγχει αν ο τύπος του αρχείου είναι αποδεκτός, αποθηκεύει το αρχείο ή τα αρχεία, αν πρόκειται για αρχείο τύπου .zip, με ασφαλές όνομα στο φάκελο συγχρονισμού του χρήστη στον server, συνδέεται μέσω socket με τον server στο port 8004 και του στέλνει μήνυμα ενημέρωσης το οποίο περιλαμβάνει το όνομα του χρήστη, το ασφαλές όνομα και το πραγματικό όνομα. Στην συνέχεια περιμένει μήνυμα επιβεβαίωσης από τον server.

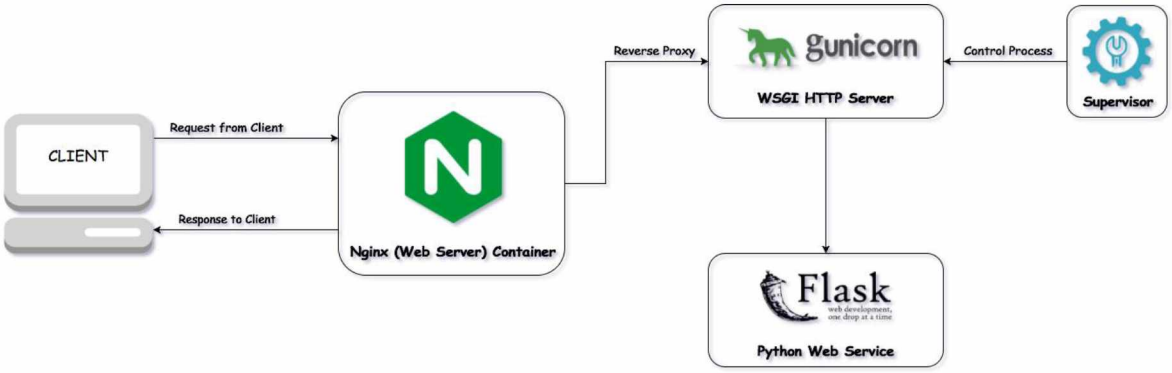
Η διαδικασία υποδοχής ενημερώσεων στον server που προέρχονται από web-clients είναι σχεδόν ίδια με αυτή της υποδοχής ενημερώσεων από εφαρμογή client με την μόνη διαφορά ότι σαν mac address χρησιμοποιεί το μηδέν και τα στοιχεία νεότερης αλλαγής τα παίρνει απευθείας από το αρχείο που αποθηκεύεται στον server. Επίσης το port 8004 χρησιμοποιείται μονάχα για υποδοχή ενημερώσεων που προέρχονται από τον ιστό. Σε περίπτωση που έχουμε συμπιεσμένο αρχείο τύπου .zip πραγματοποιεί αποσυμπίεση του αρχείου και πραγματοποιεί την παραπάνω διαδικασία για κάθε αρχείο και φάκελο του συμπιεσμένου αρχείου.

Το download πραγματοποιείται απευθείας από τον web-server. Αν πρόκειται για αρχείο αποθηκεύεται απευθείας σαν συνημμένο, ενώ αν πρόκειται για φάκελο προηγείται αυτού η δημιουργία σε προσωρινή μνήμη ενός συμπιεσμένο αρχείου το οποίο λαμβάνει τελικά ο χρήστης.

Τέλος η ανάκτηση κωδικού πραγματοποιείται με την χρήση του πρωτόκολλου smtp (Simple Mail Transfer Protocol) το οποίο έχει καθιερωθεί ως την πρώτη επιλογή για μετάδοση μηνυμάτων ηλεκτρονικού ταχυδρομείο στο Διαδίκτυο [24] και με την αξιοποίηση του JWT (JSON Wen Token). Το JWT είναι προτεινόμενο πρότυπο διαδικτύου το οποίο παρέχει web tokens σε μορφή json. Τα δεδομένα κρυπτογραφούνται με βάση ένα κλειδί της επιλογής του προγραμματιστή και συνήθως έχουν συγκεκριμένη διάρκεια ζωής. Όταν ένα web token έχει λήξει, ο χρήστης χάνει την πρόσβαση που μπορεί να είχε σε κάποια υπηρεσία. Στην περίπτωση μας όταν ο χρήστης ζητά να ανακτήσει τον κωδικό του καλείται να εισάγει το username του. Τότε δέχεται ένα μήνυμα στο email που είχε καταχωρήσει όταν έκανε την εγγραφή το οποίο περιέχει έναν σύνδεσμο που θα τον οδηγήσει σε μία φόρμα δημιουργίας νέου κωδικού. Σε αυτόν τον σύνδεσμο περιλαμβάνεται ένα ενεργό jwt το οποίο περιέχει το όνομα του χρήστη που ζητά να επαναφέρει τον κωδικό του. Ο χρήστης πρέπει να μπει και να ολοκληρώσει την διαδικασία εντός μισής ώρας αφού κάνει το αίτημα καθώς τότε λήγει το jwt.

5.4 Deploy and Setup

Το web-interface, καθώς επίσης και η εφαρμογή του server, φιλοξενούνται σε έναν virtual private server (VPS) ο οποίος παρέχεται από την ιδιωτική εταιρία Linode. Το λειτουργικό σύστημα που διαθέτει είναι το Ubuntu 20.10. Το deploy του flask application πραγματοποιήθηκε με βάση το παρακάτω μοντέλο, χρησιμοποιώντας τις τεχνολογίες nginx, gunicorn και supervisor [14].



Εικόνα 5.3: Σχηματικό διάγραμμα του Deploy

Όπως φαίνεται και στο σχήμα το Nginx [11] έχει το ρόλο του Web Server, το gunicorn του WSGI HTTP Server, το Flask του Web Service και το Supervisor του controller. Πριν ξεκινήσουμε την διαδικασία του setup θα χρειαστεί να εγκαταστήσουμε ένα firewall, δηλαδή ένα τείχος προστασίας, προκειμένου να θωρακίσουμε τον server μας. Οπότε προχωράμε στην εγκατάσταση του ufw (Uncomplicated Firewall) το οποίο αποτελεί μια απλή λύση για το πρόβλημα μας. Η εγκατάσταση πραγματοποιείται με την παρακάτω εντολή και τα δικαιώματα παραχωρούνται με τις εντολές που ακολουθούν. Αν δεν έχουμε συνδεθεί στον server μας σαν root τότε θα χρειαστεί στην αρχή των εντολών να συμπληρώνουμε με την εντολή sudo και να εισάγουμε τον κωδικό του root προκειμένου να εκτελούνται οι εντολές με δικαιώματα διαχειριστή.

```
# apt install ufw

# ufw default allow outcoming

# ufw default deny incoming

# ufw allow ssh

# ufw enable
```

Στην συνέχεια προχωράμε στην εγκατάσταση της νεότερης έκδοσης της python, του python-pip και του python-virtualenv. Το pip είναι ένα σύστημα διαχείρισης πακέτων σε python και χρησιμοποιείται για την εγκατάσταση και την διαχείριση πακέτων λογισμικού. Το virtualenv είναι εργαλείο το οποίο παράγει εικονικό περιβάλλον python και σε συνδυασμό με το pip θα δημιουργήσει το περιβάλλον στο οποίο θα τρέχει το Flask application μας. Συγκεκριμένα

```
# apt install python3

# apt install python3-pip

# apt install python3-venv
```

Δημιουργούμε και τρέχουμε το venv.

```
#python3 -m venv flask-server/venv

#source flask_server/bin/activate
```

Πλέον ότι εγκαθιστούμε μέσω του pip αποθηκεύεται στο venv. Οπότε μπορούμε να εγκαταστήσουμε τις αναγκαίες βιβλιοθήκες. Επίσης το vevn είναι ένα απαραίτητο εργαλείο για το deploy του flask application καθώς μέσω αυτού λειτουργεί και το gunicorn. Εγκαθιστούμε την Flask και τις υπόλοιπες βιβλιοθήκες

```
(venv) # pip install Flask

(venv) # pip install -r requirements.txt
```

Στην συνέχεια προκειμένου να προστατεύσουμε τα ευαίσθητα περιεχόμενα του κωδικά μας δημιουργούμε ένα αρχείο τύπου config.json μέσα στον φάκελο /etc ο οποίος είναι προσβάσιμος μόνο μέσω του διαχειριστή και ανακτάμε πλέον κάθε ευαίσθητη πληροφορία μέσα από αυτό το αρχείο.

Για παράδειγμα ένα αρχείο config.json μπορεί να περιέχει το SECRET_KEY και τα στοιχεία ενός email

```
{
    "SECRET_KEY":"my secret key",
    "MAIL_USERNAME":"usename@gmail.com",
    "MAIL_PASSWORD":"mypassword"
}
```

Τέλος για να είναι λειτουργικό το Flask application θα πρέπει να εγκαταστήσουμε την βάση δεδομένων ρησιμοποιούμε Mysql [8], οπότε θα πρέπει να την εγκαταστήσουμε και

καλό θα ήταν να δημιουργήσουμε ένα χρήστη ο οποίος θα συνδέεται στην βάση και θα πραγματοποιεί αλλαγές. Αυτά υλοποιούνται με τις παρακάτω εντολές.

Εγκατάσταση

apt-get install mysql-server

Δημιουργία χρήστη και παραχώρηση δικαιωμάτων

mysql -u root -p

mysql> CREATE USER 'username'@'localhost' IDENTIFIED BY 'password';

mysql> GRANT ALL PRIVILEGES ON *.* TO 'username'@'locahost';

mysql> quit

Αφού έχουμε ολοκληρώσει και το παραπάνω βήμα μπορούμε να ελέγξουμε την λειτουργικότητα του Flask application τρέχοντας το μέσα από το venv.

(venv) # export FLASK_APP=run.py

(vevn) # flask run --host=0.0.0.0

Σημείωση: Στο αρχείο run.py θα πρέπει να αρχικοποιούμε την μεταβλητή app και επίσης θα πρέπει να επιτρέψετε την πρόσβαση στο port 5000 καθώς το Flask χρησιμοποιεί αυτό σαν προεπιλογή. Επομένως

ufw allow 5000

ufw enable

Εφόσον έχουμε κάνει τις δοκιμές μας και έχουμε διορθώσει τυχόν προβλήματα είμαστε έτοιμοι να προχωρήσουμε. Σειρά έχει η εγκατάσταση και το setup του nginx.

apt install nginx

Διαγράφουμε το default αρχείο και δημιουργούμε το αρχείο του Flask app

rm /etc/nginx/sites-enabled/default

nano /etc/nginx/sites-enabled/flask-project

```
server {
    listen 80;
    server_name your.host.public.ip;
    location /static {
            alias /home/user/flask-server/project-folder/static;
    }
    location / {
            proxy_pass http://localhost:8000;
            include /etc/nginx/proxy_params;
            proxy_redirect off;
    }
}
```

Σε αυτό το σημείο κάνοντας ένα restart το nginx και επιτρέποντας την πρόσβαση σε αιτήματα http/tcp θα πρέπει να μπορείτε να δείτε τα static αρχεία σας από οποιοδήποτε browser όμως όχι ακόμη το flask app. Επίσης πλέον δεν χρειαζόμαστε την πρόσβαση στο port 5000 καθώς το nginx έχουμε ορίσει σαν port υποδοχής το 80.

```
# ufw allow http/tcp
# ufw delete 5000
# ufw enable
# systemctl restart nginx
```

Για να δούμε το flask application online θα πρέπει να εγκαταστήσουμε το gunicorn στο venv και να ελέγξουμε πως όλα δουλεύουν όπως πρέπει.

(vevn) # pip install gunicron

(venv) # gunicorn -w 3 run:app

Εφόσον τα παραπάνω εξελίχθηκαν ομαλά, για να ολοκληρώσουμε την διαδικασία του setup θα πρέπει να εγκαταστήσουμε το supervisor και να δημιουργήσουμε το .conf το οποίο θα χρησιμοποιεί.

apt install supervisor

nano /etc/supervisor/conf.d/flask-project.conf

[program:flask-project]

directory=/home/novas/flask-server

command=/home/novas/flask-server/venv/bin/gunicorn -w 3 run:app

user=user

autostart=true

autorestart=true

stopasgroup=true

killasgroup=true

stderr_logfile=/var/log/flask-project/flask-project.err.log

stdout_logfile=/var/log/flask-project/flask-project.out.log

Επίσης δημιουργούμε τα αρχεία που θα καταγράφονται τα errors και τα output και κάνοντας ένα reload τον supervisor το Flask application είναι πλέον online.

touch /var/log/flsync/flsync.err.log

touch /var/log/flsync/flsync.out.log

supervisorctl reload

Αν χρειαστεί να κάνουμε κάποια αλλαγή αρκεί να κάνουμε restart το nginx και η αλλαγή μας θα είναι διαθέσιμη.

systemctl restart nginx

ΚΕΦΆΛΑΙΟ 6

ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

Η υπηρεσία που δημιουργήθηκε στο πλαίσιο της παρούσας διπλωματικής εργασίας έχει σαν στόχο να καλύψει κυρίως τις ανάγκες του συγχρονισμού αρχείων σε πραγματικό χρόνο. Αυτό την κατατάσει στην κατηγορία υπηρεσιών που αξιοποιούν το cloud κατά κύριο λόγο για συγχρονισμό και κατά δεύτερο λόγο για αποθήκευση αρχείων. Επομένως είναι λογικό ότι δόθηκε περισσότερη έμφαση στην υλοποίηση του συγχρονισμού με όσο το δυνατόν γρηγορότερα και ασφαλέστερα μέσα.

Το παραπάνω έχει ως αποτέλεσμα η υπηρεσία να παρέχει περιορισμένες δυνατότητες σχετικά με το ιστορικό των αποθηκευμένων αρχείων και φακέλων. Μια σημαντική προσθήκη επομένως θα ήταν η δυνατότητα ανάκτησης αρχείων που έχουν διαγραφεί ή ακόμη και η ανάκτηση παλαιότερων εκδόσεων του ίδιου αρχείου.

Για την ενίσχυση της ασφάλειας και της ιδιωτικότητας της αποθηκευμένης πληροφορίας είναι σημαντικό για την ασφάλεια των αρχείων η αποθήκευση τους στον εξυπηρετητή να πραγματοποιείται κρυπτογραφημένα με κλειδί που επιλέγει ο τελικός χρήστης και χωρίς η ίδια η υπηρεσία να έχει πρόσβαση στο μην κρυπτογραφημένο περιεχόμενο. Επιπλέον, η υποστήριξη ενός backup εξυπηρετητή ή άλλου μηχανισμού διπλότυπης καταγραφής θα διασφάλιζε ότι σε περίπτωση κατάρρευσης του κυρίως εξυπηρετητή θα διασωθούν τα δεδομένα των χρηστών.

Ως προς την εμπειρία του χρήστη η desktop εφαρμογή της υπηρεσίας θα μπορούσε να γίνει πολύ πιο εύχρηστη με την δημιουργία της λειτουργίας remember me για τα στοιχεία ταυτοποίησης. Ο χρήστης δεν θα χρειάζεται να καταχωρεί τα διαπιστευτήρια του κάθε φορά που χρησιμοποιεί την υπηρεσία και σε συνδυασμό με την δυνατότητα αυτόματης έναρξης θα γλιτώνει χρόνο και κόπο. Τέλος χρήσιμο θα ήταν ο χρήστης να έχει την δυνατότητα να βλέπει σε πραγματικό χρόνο ποιά αντικείμενα του είναι ενημερωμένα και ποια εκκρεμούν.

Τέλος, μια σημαντική προσθήκη θα ήταν εκτός από δημιουργία μεμονωμένων χρηστών να υπήρχε και η δυνατότητα δημιουργίας ομάδων χρηστών που εργάζονται πάνω σε κοινά διαμοιραζόμενους φακέλους (π.χ. project). Η δυνατότητα αυτή, σε συνδυασμό με την

δυνατότητα καταγραφής και ανάκτησης του ιστορικού των αλλαγών για κάθε αρχείο θα παρείχε επαυξημένες δυνατότητες συνεργατικού διαμοιρασμού αρχείων και φακέλων

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Miguel Grinberg, "Flask Web Development : Developing Web Applications with Python", O'Reilly Media,Inc, USA, 2018, ISBN : 978-1491991732.

[2] Jo Foster, "Learn HTML for Beginners: The Illustrated Guide to Coding" , Ellumeniten Press, 2019, ISBN : 978-1911174912.

[3] Jon Duckett, "HTML and CSS: Design and Build Websites", Wiley 2019, ISBN : 978-118008188.

[4] Alex Berson, "Client/Server architecture", McGraw-Hill 1996, ISBN: 978-0070056640.

[5] Leon Shklar, Rich Rosen, "Web Application Architecture : Principles, Protocols and Practices", John Wiley and Sons Inc, 2009, ISBN: 978-0470518601.

[6] Thomas Erl, Ricardo Puttini, Zaigham Mahmood, "Cloud Computing: Concepts, Technology, & Architecture (The Pearson Service Technology Series from Thomas Erl)" , Pearson Technology Group, 2013, ISBN: 978-0133387520.

[7] Sergey A. Babkin, "The Practice of Parallel Programming", CreateSpace Independent Publishing Platform, 2010, ISBN: 978-1451536614.

[8] Jesper Wisborg Krogh, "MySQL Connector/Python Revealed: SQL and NoSQL Data Storage Using MySQL for Python Programmers", Apress 2018, ISBN: 978-1484236932.

[9] Mark Lutz, "Learning Python", O'Reilly Media 2013, ISBN: 978-1449355739.

[10] Paul Sanderson, Dr. Richard Hipp, Brett Shavers, Heather Mahalik, Eric Zimmerman, "SQLite Forensics", Independently published 2018, ISBN:978-1980293071.

[11] Rahul Soni, "Nginx: From Beginner to Pro" , Apress 2016, ISBN: 978-14184216576.

[12] Jay LaCroix, "Mastering Ubuntu Server: Gain expertise in the art of deplaying, configuring, managing, and troubleshooting Ubuntu Server", Packt Publishing, 2020, ISBN:978-1800564640.

[13] Christof Paar, Jan Pelzl, Bart Preneel, "Understanding Cryptography: A Textbook for Students and Practitioners", Springer 2010, ISBN:978-3642041006.

[14] Miguel Grinberg, "The New And Improved Flask Mega-Tutorial" Independently published 2017, ISBN:978-1977051875.

[15] Lindsay Bassett, “Introduction to JavaScript Object Notation: A To-the-Point Guide to JSON”, O’Reilly Media, 2015, ISBN:978-1491929483.

[16] Mark Roseman, “Modern Tkinter for Busy Python Developers: Quickly learn to create great looking user interface for Windows, Mac and Linux using Python’s standard GUI toolkit”, Late Afternoon Press, 2020, ASIN: B08KWRQFW1.

[17] Dropbox: https://www.dropbox.com/?landing=dbv2

[18] MEGA: https://mega.io/

[19] Pithos: https://okeanos.grnet.gr/services/pithos/

[20] Google drive: https://www.google.com/intl/el_gr/drive/

[21] Box: https://www.box.com/home

[22] Backblaze: https://www.backblaze.com/

[23] NSIS : https://nsis.sourceforge.io/Main_Page

[24] JWT : https://jwt.io/