Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Προηγμένα συστήματα πληροφορικής – Ανάπτυξη λογισμικού και τεχνητής νοημοσύνης»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Εξατομικευμένη διαμοίραση αρχείων με ασφάλεια** **Securely sharing personalized files**
Ονοματεπώνυμο Φοιτητή	**Γεώργιος Αναγνωστάκης**
Πατρώνυμο	**Ηρακλής**
Αριθμός Μητρώου	**ΜΠΣΠ 19003**
Επιβλέπων	**Ευάγγελος Σακκόπουλος, Επίκουρος καθηγητής**

Ημερομηνία παράδοσης: Μάρτιος 2021

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Μαρία Βίρβου Καθηγήτρια	Ευθύμιος Αλέπης Αναπληρωτής καθηγητής	Ευάγγελος Σακκόπουλος Επίκουρος καθηγητής

Αφιερωμένο στους γονείς μου…

Ευχαριστίες

Θα επιθυμούσα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή της διπλωματικής μου εργασίας Κ. Ευάγγελο Σακκόπουλο κυρίως για την εμπιστοσύνη που μου έδειξε στη ανάθεση της διπλωματικής αυτής εργασίας, καθώς και για την συνεχή και άμεση υποστήριξη του σε ότι χρειάστηκε. Θα ήθελα επίσης να ευχαριστήσω την οικογένεια μου για την ηθική τους συμπαράσταση και την υποστήριξη τους καθ' όλη τη διάρκεια της σταδιοδρομίας μου.

Εξατομικευμένη διαμοίραση αρχείων με ασφάλεια

Περίληψη

Η παρούσα μεταπτυχιακή διατριβή θα ασχοληθεί με την εξατομικευμένη διαμοίραση αρχείων με ασφάλεια. Αρχικά παρουσιάζονται βασικές έννοιες γύρω από την ασφάλεια και την κρυπτογράφηση, προκειμένου να διευκολύνουμε τον αναγνώστη στην κατανόηση της εφαρμογής και των λειτουργιών της. Επιπροσθέτως, αναλύεται ο τρόπος παραγωγής των απαραίτητων κλειδιών και πιστοποιητικών τόσο για την Αρχή Πιστοποίησης όσο και για το χρήστη.

Η κατανόηση της καθοριστικότητας της ασφάλειας δεδομένων στην εποχή μας και το πως μπορεί να εξασφαλιστεί αποτελεί τον κυριότερο σκοπό της εφαρμογής. Οι απαραίτητες λειτουργίες για τον χρήστη καταγράφονται και ενσωματώνονται με σκοπό την ομαλή λειτουργία της εφαρμογής και την ύπαρξη ορθής ροής. Ο χρήστης δύναται να εγγραφεί στο σύστημα μας, να δημιουργήσει ψηφιακή υπογραφή και να υπογράψει δικά του αρχεία τα οποία φιλοξενούνται στην εφαρμογή μας. Στη συνέχεια καθίσταται εφικτή η αποστολή τους μέσω ηλεκτρονικού ταχυδρομείου.

Σκόπιμα επιλέχθηκε η υλοποίηση της εφαρμογής σε τοπικό δίκτυο ώστε να αποφευχθούν επιπλέον προβλήματα ασφάλειας καθώς και να μειωθεί η ανάγκη δημιουργίας και συντήρησης ενός διαδικτυακού ιστοτόπου από τον διαχειριστή της εφαρμογής.

Abstract

This dissertation will deal with secure personalized file sharing. Basic concepts around security and encryption are introduced first, in order to facilitate the reader in understanding the application and its functions. In addition, the way of producing the necessary keys and certificates for both the Certification Authority and the user is analyzed.

Understanding the determinants of data security in our time and how it can be ensured is the main purpose of the application. The necessary functions for the user are recorded and integrated in order for the smooth operation of the application and the existence of a proper flow. The user can register in our system, create a digital signature and sign his own files which are hosted in our application. It then becomes possible to send them via email. analyzed.

The implementation of the application on a local network was deliberately chosen in order to avoid additional security problems as well as to reduce the need for the creation and maintenance of a website by the application administrator.

Περιεχόμενα

1. Εισαγωγή

Στο ηλεκτρονικό εμπόριο αλλά και γενικότερα στις ηλεκτρονικές επικοινωνίες ένα από τα σημαντικότερα ζητήματα είναι αυτό της ασφάλειας συναλλαγών. Σε περίπτωση που η επικοινωνία δύο μερών γίνεται μέσω κλειστού δικτύου τότε το πρόβλημα της ασφάλειας δεδομένων είναι σαφώς μειωμένο καθώς ο φορέας που λειτουργεί και ελέγχει το δίκτυο μπορεί να εντοπίσει εύκολα παρεμβολές ή υποκλοπές στις επικοινωνίες του δικτύου.

Στην περίπτωση όμως που η επικοινωνία δύο μερών συμβαίνει μέσω ενός ανοικτού, δημόσιου δικτύου, τότε κανείς δεν είναι ικανός να ανιχνεύσει αν το δίκτυο παρακολουθείται όπως και κανείς δεν μπορεί να εγγυηθεί πως το μήνυμα θα φθάσει ακέραιο στον προορισμό του. Μερικά από τα μεγαλύτερα προβλήματα στην ηλεκτρονική επικοινωνία δύο μερών μέσω ενός ανοικτού δημόσιου δικτύου εντοπίζονται παρακάτω:

- Η επικοινωνία να παρακολουθείται από τρίτους
- Πλαστοπροσωπία με τη χρήση πλαστής ηλεκτρονικής διεύθυνσης
- Το περιεχόμενο των μηνυμάτων να έχει αλλοιωθεί σκόπιμα από τρίτο πρόσωπο
- Να μην είναι δυνατόν να εξακριβωθεί η ταυτότητα του αποστολέα

Είναι γεγονός ότι η ηλεκτρονική επικοινωνία αποτελεί ένα αχανές κομμάτι της σύγχρονης τεχνολογίας η οποία μπορεί να εξελίσσεται συνεχώς αλλά τα προβλήματα που δημιουργεί είναι δισεπίλυτα. Στην πραγματικότητα η επικοινωνία αυτή είναι απρόσωπη και αυτοί που επικοινωνούν είναι οι ηλεκτρονικοί υπολογιστές. Είναι ανέφικτο να γνωρίζουμε ποιος βρίσκεται στην άλλη πλευρά, αν αυτός που ισχυρίζεται ότι είναι ισχύει και αν οι στόχοι του είναι κακόβουλοι.

Μια λύση σε όλα αυτά τα προβλήματα έρχεται να δώσει η ψηφιακή υπογραφή. Ως ψηφιακή υπογραφή ορίζεται η ηλεκτρονική υπογραφή η οποία παράγεται αποκλειστικά από συγκεκριμένες διαδικασίες και η οποία είναι σε θέση να εξασφαλίσει την αυθεντικότητα του υπογράφοντος καθώς και την ακεραιότητα του απορρήτου του μηνύματος. Αλλιώς, η ψηφιακή υπογραφή μας παρέχει την εγγύηση της αυθεντικότητας του αποστολέα και της μη αλλοίωσης του περιεχομένου των μηνυμάτων που διακινούνται ηλεκτρονικά.

1.1. Σκοπός

Σκοπός της παρούσας διπλωματικής εργασίας είναι η δημιουργία ενός διαδικτυακού πληροφοριακού συστήματος εξατομικευμένης διαμοίρασης αρχείων με ασφάλεια. Ο χρήστης θα έχει τη δυνατότητα, μέσα από ένα φιλικό περιβάλλον, να δημιουργήσει την προσωπική του ψηφιακή υπογραφή και να υπογράψει κάποιο προσωπικό του αρχείο. Στη

συνέχεια θα έχει την επιλογή να το αποστείλει μέσω mail και να παρακολουθεί την πρόοδο του, δηλαδή αν ανοίχτηκε και πότε από τον παραλήπτη.

Όλα τα στάδια, από την είσοδο του στο σύστημα, την δημιουργία πιστοποιητικού έως και την αποστολή του υπογεγραμμένου αρχείου περιλαμβάνουν επίπεδα ασφαλείας και ταυτοποίησης του χρήστη.

2. Ανασκόπηση πεδίου

Προκειμένου ο αναγνώστης να κατανοήσει το περιεχόμενο της διπλωματικής εργασίας κρίνεται απαραίτητη η γνώση των παρακάτω εννοιών και διαδικασιών.

2.1. Κρυπτογραφία

Ως κρυπτογραφία ορίζουμε γενικότερα την ανταλλαγή μηνυμάτων μεταξύ δύο μερών με τέτοιο τρόπο ώστε η κατανόηση του περιεχομένου των μηνυμάτων να είναι δυνατή μόνο από τον αποστολέα και τον παραλήπτη

2.2. Ψηφιακή υπογραφή

Η ψηφιακή υπογραφή είναι ένα μαθηματικό σύστημα που χρησιμοποιείται για την απόδειξη της γνησιότητας ενός ψηφιακού μηνύματος ή εγγράφου. Η έγκυρη ψηφιακή υπογραφή επιβεβαιώνει στον παραλήπτη του μηνύματος ότι το μήνυμα ανήκει στον αποστολέα που το υπέγραψε και δεν έχει υποστεί καμία σκόπιμη ή άσκοπη επεξεργασία κατά την μεταφορά του. Οι ψηφιακές υπογραφές χρησιμοποιούν έναν συνδυασμό κρυπτογραφικής συνάρτησης κατακερματισμού, ένα hash σε συνδυασμό με ασυμμετρική κρυπτογραφία

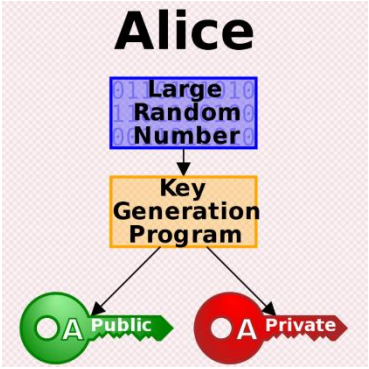
2.3. OpenSSL

Το OpenSSl είναι μια βιβλιοθήκη ανοικτού λογισμικού γραμμένη σε C που χρησιμοποιείται για την παραγωγή ιδιωτικών κλειδιών, αιτημάτων ψηφιακών πιστοποιητικών και υπογραφή πιστοποιητικών. Περιέχει υλοποιήσει των SSL και TLS πρωτοκόλλων και υποστηρίζει ένα πλήθος αλγορίθμων μεταξύ αυτών και τους Ciphers, MD5, SHA-1, SHA-2, RSA και άλλους.

2.4. Private/Public key

Ως private/public key (δημόσιο/ιδιωτικό κλειδί) ορίζουμε εναν αριθμό αρκετών bit που χρησιμοποιείται ως είσοδος στην κρυπτογράφηση.

Η δημιουργία τους γίνεται από ειδικές συναρτήσεις οι οποίες δέχονται σαν είσοδο έναν μεγάλο τυχαίο αριθμό και στην έξοδο παράγουν το ζεύγος κλειδιών. Είναι προφανές πως όσο πιο τυχαίος ο αριθμός τόσο πιο ασφαλή είναι τα κλειδιά που παράγονται.



2.5. Αρχή πιστοποίησης (Certificate authority - CA)

Η αρχή έκδοσης πιστοποιητικών (CA), επίσης μερικές φορές αναφέρεται ώς αρχή πιστοποίησης είναι μία εταιρεία ή οργανισμός που ενεργεί για την επικύρωση των ταυτοτήτων οντοτήτων (όπως ιστότοποι, διευθύνσεις ηλεκτρονικού ταχυδρομείου, εταιρείες ή μεμονωμένα άτομα) και τη σύνδεση τους με κρυπτογραφικά κλειδιά μέσω έκδοσης ηλεκτρονικών εγγράφων γνωστών ως ψηφιακών πιστοποιητικών. Ένα ψηφιακό πιστοποιητικό παρέχει:

- Πιστοποίηση, χρησιμεύοντας ως διαπιστευτήριο για την επικύρωση της ταυτότητας της οντότητας στην οποία εκδίδεται
- Κρυπτογράφηση, για ασφαλή επικοινωνία μέσω ανασφαλών δικτύων όπως το Διαδίκτυο
- Ακεραιότητα εγγράφων υπογραφεί με το πιστοποιητικό ώστε να μη μπορούν να τροποποιηθούν από τρίτο μέρος κατά τη μεταφορά

2.6. Πιστοποιητικό

Το ψηφιακό πιστοποιητικό είναι ένα ηλεκτρονικό έγγραφο που χρησιμοποιείται για την αναγνώριση μίας οντότητας και την ανάκτηση του δημοσίου κλειδιού της

2.7. X.509

Το X.509 είναι διεθνές πρότυπο που καθορίζει τον τρόπο λειτουργίας των Υποδομών Δημοσίου κλειδιού. Το πρότυπο προδιαγράφει τις μορφές διάθεσης της σχετικής πληροφορίας (κλειδιά, πιστοποιητικά, λίστες ανάκλησης) και τους αλγορίθμους επαλήθευσης του κύρους ενός πιστοποιητικού.

2.8. Αρχείο .csr

Ένα CSR αρχείο είναι ένα κωδικοποιημένο αρχείο κειμένου που περιλαμβάνει το δημόσιο κλειδί και άλλες πληροφορίες που θα περιληφθούν στο πιστοποιητικό (πχ όνομα, τομέας, οργανισμός, διεύθυνση email κτλ.) που θα «υπογράψει» με ενα ιδιωτικό κλειδί η αρχή πιστοποίησης και θα δημιουργήσει το πιστοποιητικό.

2.9. OTP (One Time Password)

OTP ή αλλιώς one-time PIN είναι ένας κωδικός ο οποίος ισχύει για περιορισμένη χρονική διάρκεια και για μία μόνο περίοδο σύνδεσης ή συναλλαγής σε σύστημα υπολογιστή ή άλλης ψηφιακής συσκευής.

Οι κωδικοί αυτοί αποφεύγουν ορισμένα μειονεκτήματα που σχετίζονται με τον παραδοσιακό έλεγχο ταυτότητας βάσει κωδικού πρόσβασης. Οι αλγόριθμοι δημιουργίας τέτοιων κωδικών χρησιμοποιούν ψευδοτυχαιότητα ή τυχαιότητα καθιστώντας δύσκολη την πρόβλεψη τους από άλλους καθώς και συναρτήσεις κρυπτογραφικού κατακερματισμού οι οποίες κάνουν δύσκολο σε «τρίτους» να ανακαλύψουν τα δεδομένα που χρησιμοποιήθηκαν για κατακερματισμό. Με αυτόν τον τρόπο αποφεύγεται η πρόβλεψη μελλοντικών κωδικών παρατηρώντας τα προηγούμενα.

Για την ενημέρωση του χρήστη με αυτόν τον κωδικό χρησιμοποιούμε διάφορες μεθόδους. Ορισμένα συστήματα χρησιμοποιούν διάφορες ηλεκτρονικές συσκευές τις οποίες μεταφέρει μαζί του ο χρήστης και δημιουργούν τον κωδικό ανά συγκεκριμένο χρονικό διάστημα ενώ άλλες χρησιμοποιούν λογισμικό που τρέχει στο κινητό τηλέφωνο του χρήστη. Ιδιαίτερα σε συναλλαγές τραπεζών χρησιμοποιείται SMS προκειμένου ο χρήστης να λάβει τον κωδικό του.

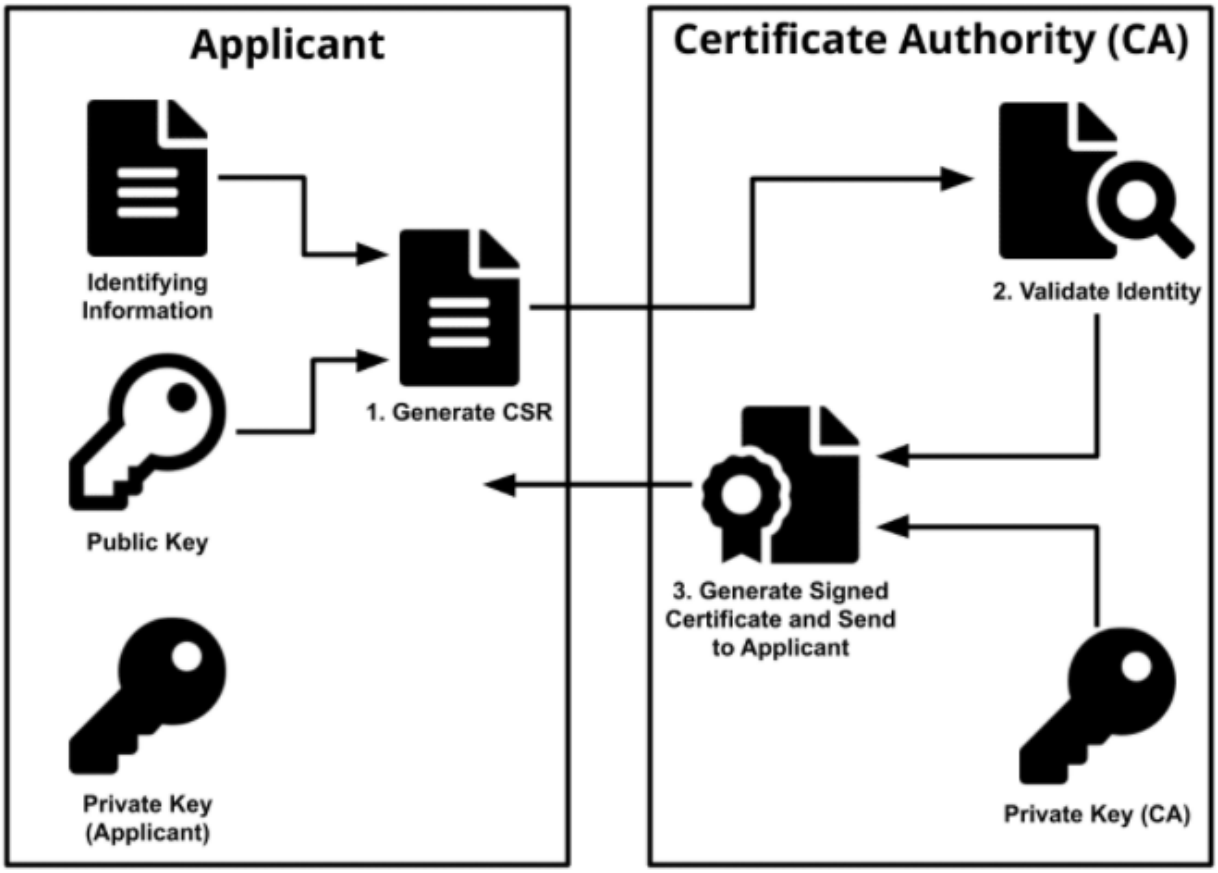
Αξίζει να σημειώσουμε πως η χρήση τους γίνεται ως ενισχυτής των παραδοσιακών κωδικών πρόσβασης προσφέροντας ένα επιπλέον επίπεδο ασφάλειας.

2.10. Διαδικασία υπογραφής πιστοποιητικού

Η διαδικασία της λήψης ενός πιστοποιητικού απαιτεί κάποιες ενέργειες τόσο του αιτών όσο και της Αρχής Πιστοποίησης.

Ο αιτών αρχικά θα δημιουργήσει με κάποια συνάρτηση ένα ζεύγος κλειδιών το οποίο περιλαμβάνει ένα δημόσιο και ένα ιδιωτικό κλειδί. Επίσης απαραίτητη είναι από μέρους του και η δημιουργία ενός αιτήματος υπογραφής πιστοποιητικού ή αλλιώς CSR το οποίο περιλαμβάνει και το δημόσιο κλειδί.

Μετά τη δημιουργία του csr ο αιτών το αποστέλλει στην αρχή πιστοποίησης. Με τη σειρά της είναι υποχρεωμένη να επαληθεύσει ότι οι πληροφορίες που περιέχει το csr είναι σωστές και να υπογράψει ψηφιακά το πιστοποιητικό με ένα δικό της ιδιωτικό κλειδί.



Εικόνα 1 Διαδικασία υπογραφής πιστοποιητικού

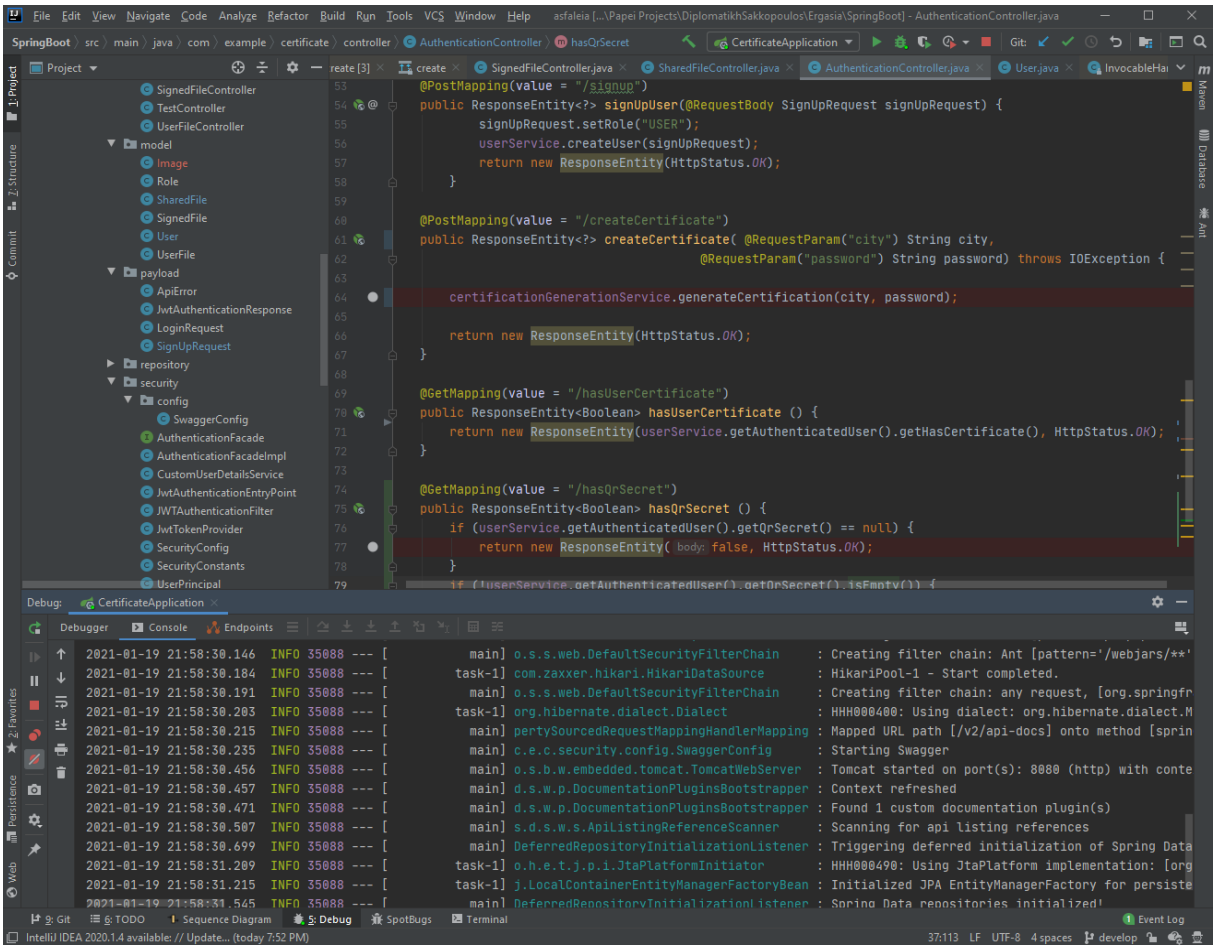
Όταν το υπογεγραμμένο πιστοποιητικό παρουσιαστεί σε τρίτο μέρος (όπως σε έναν ιστότοπο που ανήκει στον κάτοχο του πιστοποιητικού, σε ένα έγγραφο pdf κτλ) μπορεί να επιβεβαιωθεί η ψηφιακή υπογραφή της Αρχής Πιστοποίησης και ότι οι πληροφορίες δεν έχουν αλλάξει από τότε που υπογράφηκε.

3. Τεχνικές, πρότυπα και εργαλεία ανάπτυξης

Για την ολοκλήρωση της εφαρμογής χρησιμοποιήθηκαν πολλές τεχνολογίες, βιβλιοθήκες και εργαλεία ανάπτυξης. Σε αυτό το κεφάλαιο θα μελετήσουμε τα βασικά κομμάτια της εφαρμογής και θα περιγράψουμε τη χρήση τους.

3.1. IDE

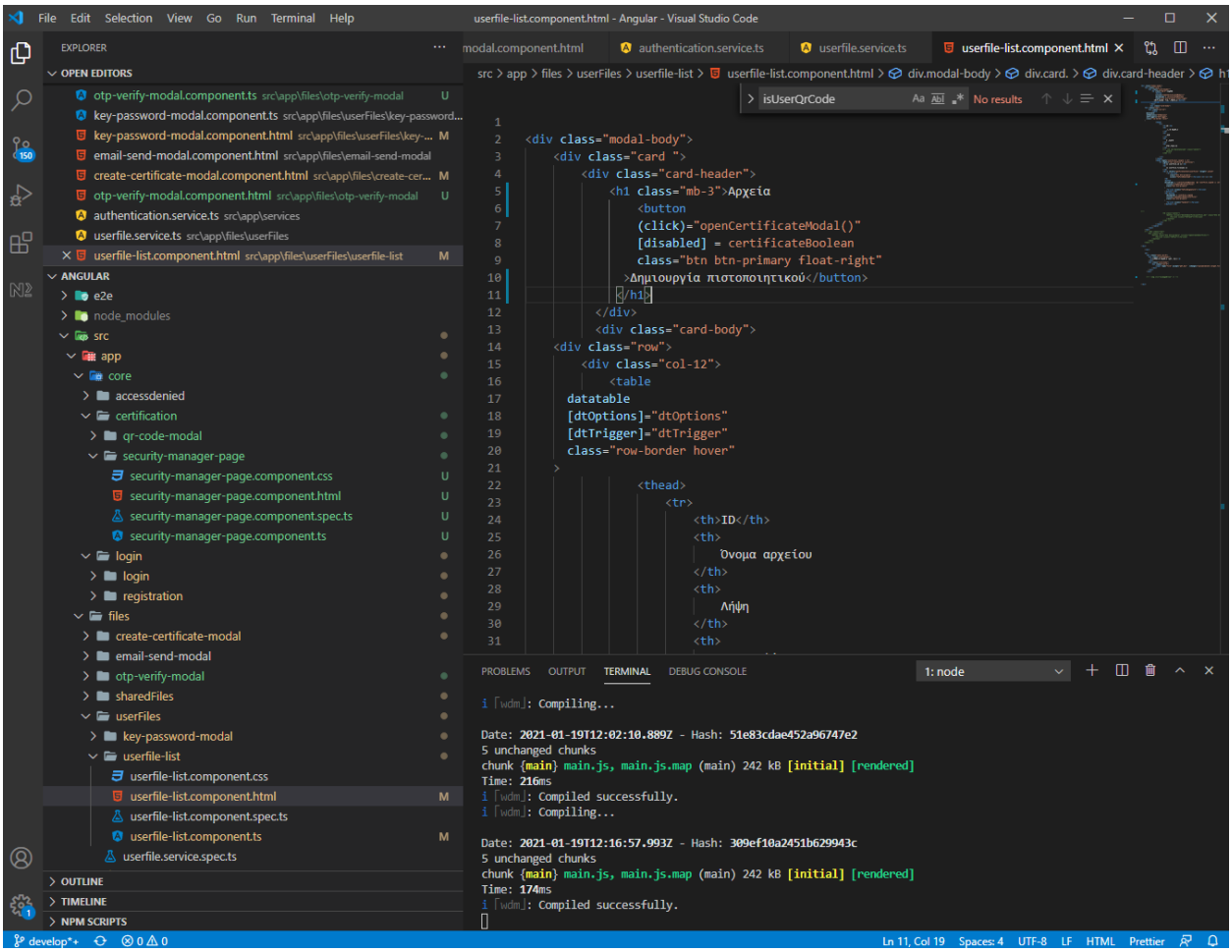
Ξεκινώντας για το πίσω μέρος της εφαρμογής (backend) χρησιμοποιήθηκε το IntelliJ IDEA. Αποτελεί ένα ολοκληρωμένο περιβάλλον ανάπτυξης γραμμένο σε JAVA για την ανάπτυξη λογισμικού υπολογιστή. Αναπτύχθηκε από την JetBrains το 2001 και υποστηρίζει μια μεγάλη γκάμα γλωσσών προγραμματισμών καθώς και framework.



Εικόνα 2 IntelliJ IDE

Για την υλοποίηση του περιβάλλοντος επικοινωνίας του χρήστη με το backend κομμάτι της εφαρμογής μας χρησιμοποιήθηκε το Microsoft Visual Studio, ένα εργαλείο δημιουργημένο από τη Microsoft και συμβατό με όλα τα λειτουργικά. Διατίθενται αρκετά plugins με σκοπό την

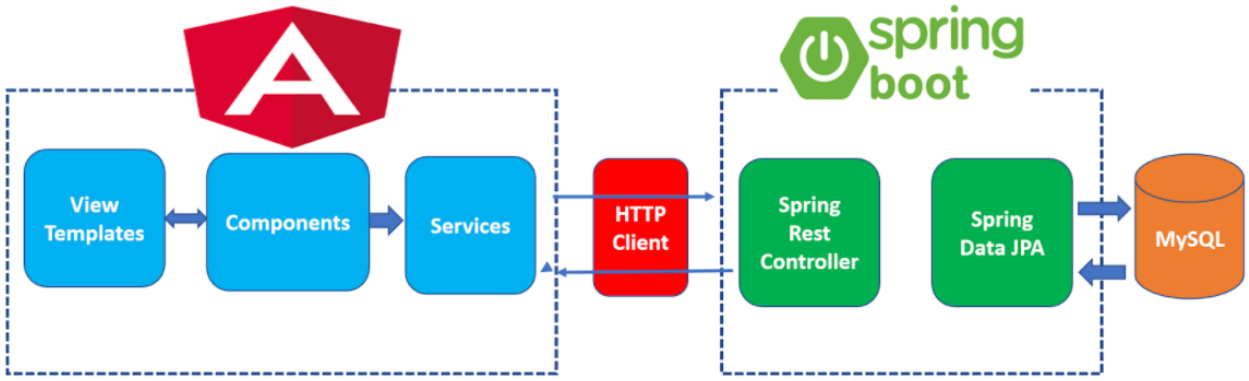
υποστήριξη διάφορων γλωσσών προγραμματισμών, βελτίωση της ποιότητας του κώδικα μας αλλά και την εμφάνιση του.



Εικόνα 3 Visual Studio Code IDE

3.2. Τεχνολογίες, γλώσσες προγραμματισμού και βιβλιοθήκες

Η παρούσα εφαρμογή απαρτίζεται από τα τρία υποσυστήματα, μια εφαρμογή υλοποιημένη χρησιμοποιώντας τις τεχνολογίες Angular (frontend), Spring boot (backend), Spring security (backend) και MySQL (database). Η αρχιτεκτονική της εφαρμογής απεικονίζεται στο ακόλουθο σχήμα.



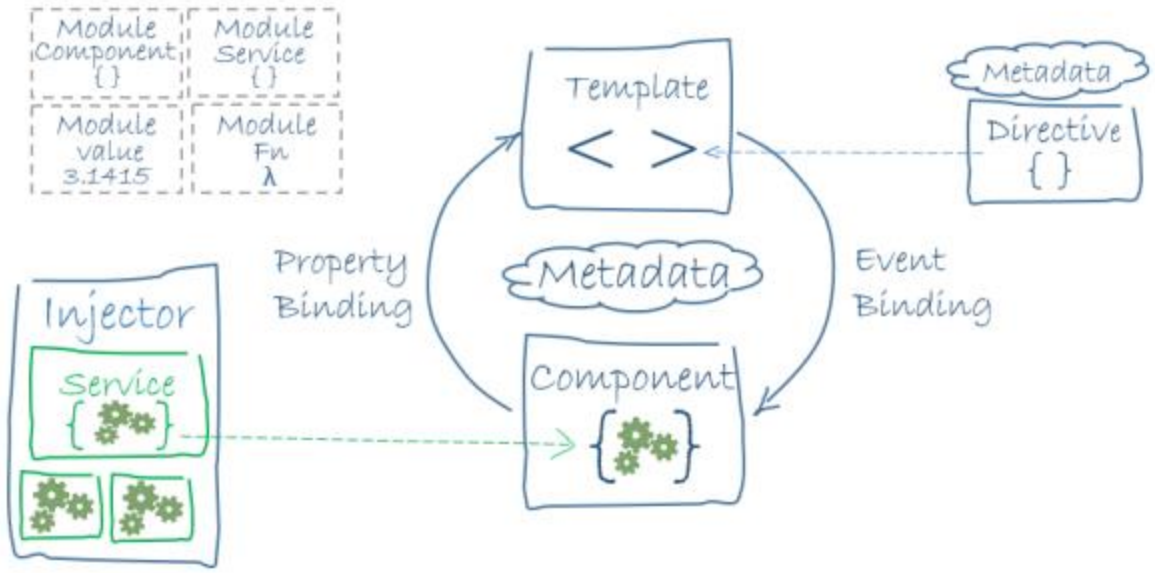
Εικόνα 4 Αρχιτεκτονική συστήματος

3.1. Frontend εφαρμογή

Αποτελεί το κομμάτι της εφαρμογής το οποίο είναι άμεσα προσβάσιμο από τους χρήστες της και τους επιτρέπει να χρησιμοποιήσουν τα εργαλεία και τις υπηρεσίες που αυτή προσφέρει. Αποτελείται από διάφορα κομμάτια τα οποία θα αναλυθούν παρακάτω.

3.4.1. Angular

Η Angular είναι ένα από τα πιο δημοφιλή JavaScript Framework για τη δημιουργία διαδραστικών client side εφαρμογών που τρέχουν στον browser. Η έκδοση που χρησιμοποιήθηκε είναι η έκδοση 6 και δεν έχει καμία σχέση με την AngularJS που οι περισσότεροι γνωρίζουν. Είναι ένα framework υλοποιημένο από τη Google, γραμμένο πλήρως από την αρχή με αρκετές βελτιώσεις και είναι διαθέσιμο από το Σεπτέμβριο του 2016. Οι δυνατότητες του περιλαμβάνουν τη δημιουργία σύγχρονων και γρήγορων Application ή ιστοσελίδων. Μία σύντομη περιγραφή της αρχιτεκτονικής της φαίνεται στην παρακάτω εικόνα.



Εικόνα 5 Αρχιτεκτονική Angular

3.4.2. HTML

Η HTML είναι το ακρωνύμιο των λέξεων HyperText Markup Language (γλώσσα μορφοποίηση υπερκειμένου) και είναι η βασική γλώσσα δόμηση σελίδων του World Wide Web (ή απλά ιστού: Web). Είναι μία γλώσσα προγραμματισμού. Χρησιμοποιείται για να σημαίνει ένα τμήμα κειμένου και να το κάνει να εμφανίζεται καλύτερα. Επιτρέπει την ενσωμάτωση ήχου και εικόνων στις Web σελίδες. Αρχικά είχε κατασκευασθεί με σκοπό μόνο την μορφοποίηση κειμένου, αλλά μεγάλωσε και ενσωμάτωσε σχεδιαστικές τεχνικές κ.α.

Η γλώσσα χρησιμοποιεί ένα αριθμό από tags για την μορφοποίηση κειμένου, για την δημιουργία συνδέσμων (links) μετάβασης ανάμεσα των σελίδα, για την εισαγωγή εικόνων, ήχου κ.α. Όταν ένας Web Browser ανοίγει ένα αρχείο HTML τα στοιχεία (tags) μεταφράζονται σε κατάλληλα χαρακτηριστικά με αποτελέσματα στην εμφάνιση και στην λειτουργικότητα της συγκεκριμένης σελίδας.

3.4.3. CSS

Τι είναι CSS; Η ονομασία CSS βγαίνει από τις λέξεις Cascading Style Sheets δηλαδή, διαδοχικά φύλλα στυλ. Πρόκειται για μια τεχνολογία ορισμού προβολής δεδομένων σε μια ιστοσελίδα η οποία έρχεται και δένει μαζί με αυτή της HTML η οποία με την σειρά της καθορίζει την δομή μιας σελίδας.

Τα CSS καθορίζουν τον τρόπο μορφοποίησης με τον οποίο θα προβάλλεται το περιεχόμενο μιας html σελίδας όπως είναι τα χρώματα και η θέση των στοιχείων της σελίδας. Ένα μεγάλο πλεονέκτημα των CSS είναι ότι μπορούν να αποθηκευτούν σε εξωτερικό αρχείο με κατάληξη .css και να χρησιμοποιηθούν από κοινού σε περισσότερες από μια .html σελίδες, εξοικονομώντας με τον τρόπο αυτό χρόνο και κόπο υλοποίησης. Αυτό πρακτικά σημαίνει πως μια αλλαγή σε αυτό το αρχείο θα ενημερώσει όλες τις html σελίδες με τις οποίες συνδέεται.

Επίσης αξίζει να σημειωθεί πως μέσω της CSS όταν χρησιμοποιούμε εξωτερικό αρχείο ο browser το κατεβάζει μόνο την πρώτη φορά και το αποθηκεύει στην cache, κάνοντας έτσι τις σελίδες μας γρηγορότερες.

3.2. Backend εφαρμογή

Με τον όρο backend χαρακτηρίζεται το σύστημα το οποίο χειρίζεται ένας διαχειριστής και βρίσκεται στο πίσω μέρος μιας εφαρμογής που χρησιμοποιείται από κάποιον χρήστη. Μέσω αυτού του συστήματος μπορεί ο χρήστης του να αλλάζει τις ρυθμίσεις και τις διαθέσιμες επιλογές που του προσφέρονται. Το backend μας αποτελείται ουσιαστικά από το Spring Boot framework στο οποίο συμπεριλάβαμε το Spring Security, Spring data JPA και αρκετά ακόμα τα οποία θα επεξηγηθούν στη συνέχεια.

3.2.1. Spring Boot

Το Spring Boot είναι ένα framework βασισμένο στη Java, το οποίο χρησιμοποιείται για τη δημιουργία services. Είναι μια επέκταση του Spring Framework η οποία εξαλείφει τις επαναλαμβανόμενες και πολύπλοκες ρυθμίσεις οι οποίες απαιτούνται για την δημιουργία μιας Spring εφαρμογής. Αποτελεί ουσιαστικά ένα αρκετά ταχύτερο σύστημα ανάπτυξης λογισμικού, αξιόπιστο και συναγωνιστικό με πολλά άλλα frameworks. Κάποια από τα χαρακτηριστικά του και τα οφέλη του σημειώνονται παρακάτω

- Dependencies, μέσω των οποίων απλοποιείται η ανάπτυξη και παραμετροποίηση της εφαρμογής
- Ενσωματωμένος server (tomcat) για την αποφυγή της ξεχωριστής εγκατάστασης του
- Δεν απαιτείται παραμετροποίηση XML αρχείων
- Παροχή μετρήσεων ελέγχου κατάστασης και εξωτερικής παραμετροποίησης της εφαρμογής

3.2.2. Java

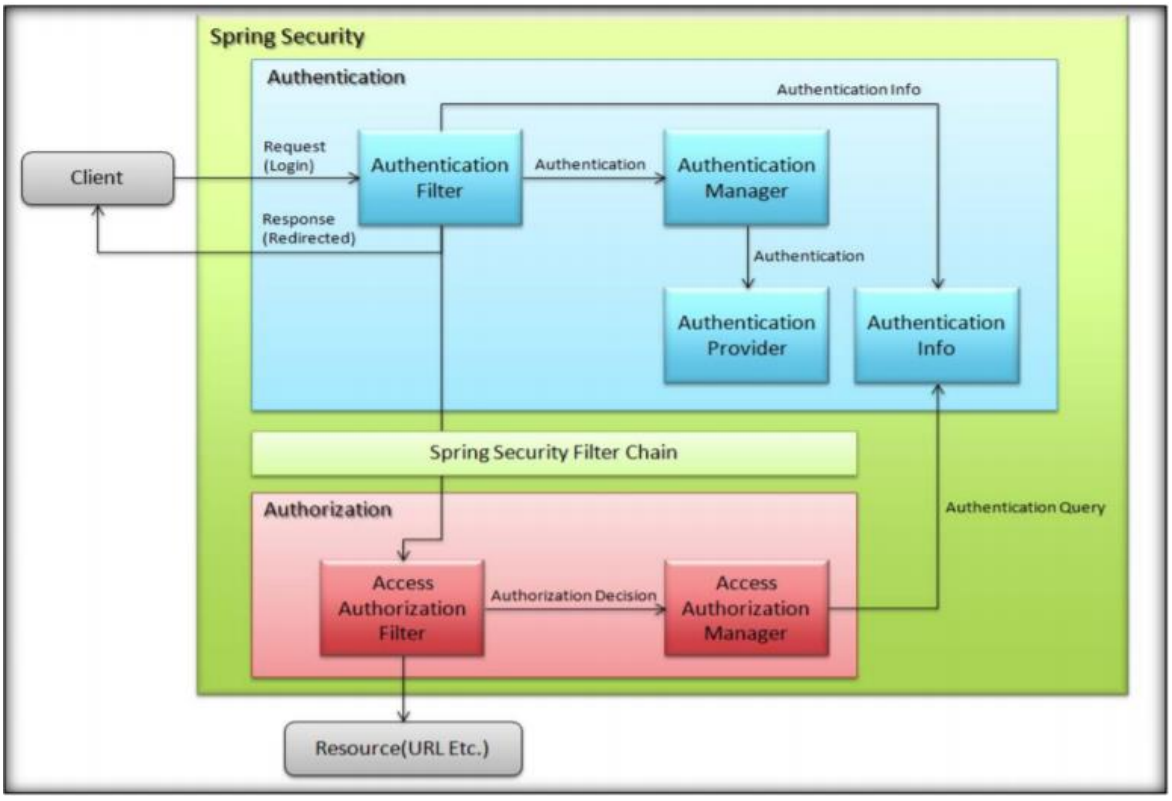
Η java είναι μια σύγχρονη και ασφαλής αντικειμενοστραφής γλώσσα προγραμματισμού. Ένα από τα βασικά πλεονεκτήματα της καθώς και ο λόγος για τον οποίο την επιλέξαμε είναι ότι έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε Java τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh (σύντομα θα τρέχουν και σε Playstation καθώς και σε άλλες κονσόλες παιχνιδιών) χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα.

3.2.2. Spring Security

Το Spring Security είναι ένα framework που επιδιώκει να εξασφαλίσει την ασφάλεια των διαδικτυακών εφαρμογών βασισμένων στο Spring Framework. Το Spring Security παρέχει υπηρεσίες ελέγχου ταυτότητας και πρόσβασης, ενώ παράλληλα δίνει τη δυνατότητα στον προγραμματιστή να επιλέξει τη μέθοδο κρυπτογράφησης των κωδικών χωρίς να γνωρίζει το σύνολο των ρυθμίσεων που απαιτούνται στο παρασκήνιο για την ορθή λειτουργία του framework. Επίσης, παρέχει προστασία από ορισμένους τύπους επιθέσεων όπως Enumeration attacks, SQL Injection, Brute Force Authentication, Session Hijacking και CSRF επιθέσεις.

Η αρχιτεκτονική του Spring Security είναι σχεδιασμένη έτσι ώστε να διαχωρίζει τις υπηρεσίες ελέγχου ταυτότητας (Authentication) από τις υπηρεσίες ελέγχου πρόσβασης (Authorization). Η βασική στρατηγική που χρησιμοποιείται για τον έλεγχο ταυτότητας είναι ο AuthenticationManager όπου χαρακτηρίζει ένα αίτημα ταυτοποίησης ως έγκυρο, άκυρο ή αβέβαιο εφόσον δε μπορεί να προκύψει ασφαλές συμπέρασμα ότι η προσπάθεια ταυτοποίησης γίνεται από τον πραγματικό χρήστη. Ο έλεγχος ταυτότητας αφορά το Login.

Ο έλεγχος πρόσβασης γίνεται μέσω του μηχανισμού AccessDecisionManager που σε κάθε προσπάθεια του χρήστη να αποκτήσει πρόσβαση σε μια συγκεκριμένη πηγή ή μέθοδο, επιστρέφει με βάση τις προδιαγραφές που έχουν δοθεί ένα αλφαριθμητικό, που αφορά τους κανόνες πρόσβασης στην πηγή. Για να εξασφαλιστεί ο έλεγχος πρόσβασης, αποδίδεται σε κάθε κατηγορία χρηστών ένας ρόλος, ο οποίος παρέχει συγκεκριμένα δικαιώματα πρόσβασης.



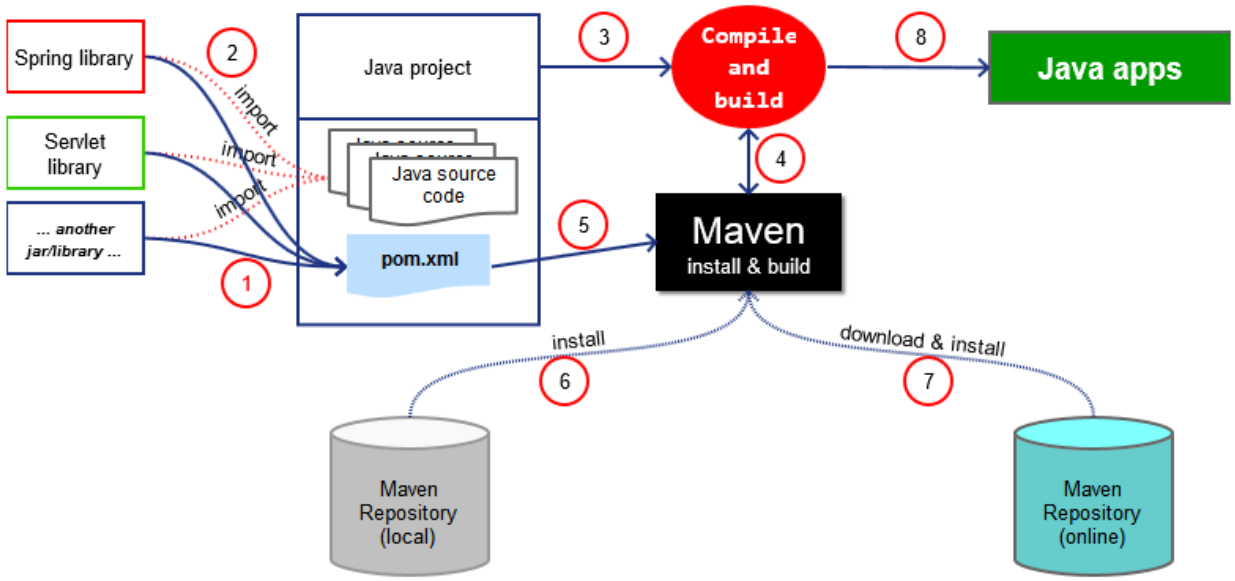
Εικόνα 6 Αρχιτεκτονική Spring Security

3.2.3. Spring Data JPA

Το Spring Data JPA δημιουργήθηκε προκειμένου να δώσει λύση στην διατήρηση των δεδομένων και για να λειτουργήσει ως δίαυλος επικοινωνίας μεταξύ του spring boot και της βάσης δεδομένων MySQL.

3.2.4 Maven

Το maven είναι ένα εργαλείο αυτοματισμού κατασκευής που χρησιμοποιείται κυρίως σε έργα Java. Διαχειρίζεται αυτόματα τα dependencies, βιβλιοθήκες, την μεταγλώττιση καθώς και πολλές άλλες λειτουργίες ενός project. Μέσω αυτής της αυτοματοποίησης ελαχιστοποιείται η πιθανότητα λάθους του προγραμματιστή και εξοικονομείται σημαντικός χρόνος.



Εικόνα 7 Λειτουργία maven

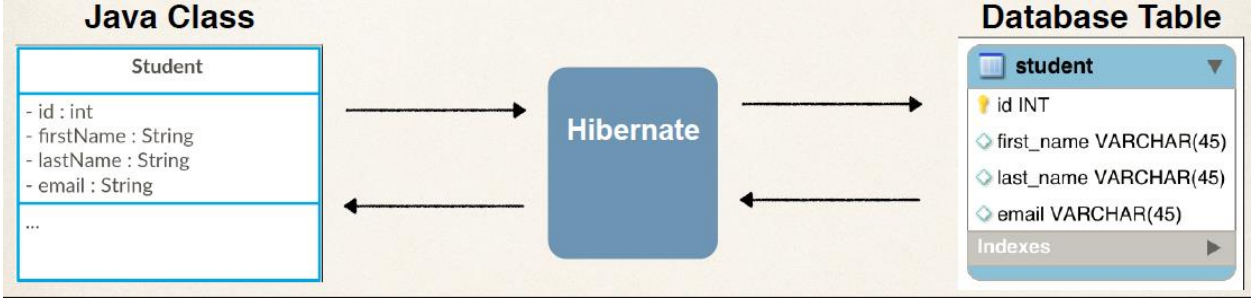
3.2.5. Hibernate

Το hibernate συγκαταλέγεται στην κατηγορία λογισμικού ORM(Object Relational Mapping). Το λογισμικό ORM στοχεύει στη δημιουργία διεπαφής μεταξύ των σχεσιακών βάσεων δεδομένων και του αντικειμενοστραφούς προγραμματισμού. Με απλά λόγια, προσφέρει τη χρησιμοποίηση μιας σχεσιακής βάσης δεδομένων σαν να ήταν αντικειμενοστραφής.

Για να επιτευχθεί αυτό δημιουργεί αντιστοιχίες μεταξύ των εννοιών του αντικειμενοστραφούς προγραμματισμού (συσχετίσεις, κληρονομικότητα, πολυμορφισμός) - που δεν υπάρχουν σε μια σχεσιακή βάση δεδομένων - και των πινάκων και σχέσεων μεταξύ των πινάκων μιας σχεσιακής βάσης. Με αυτό τον τρόπο ο προγραμματιστής *βλέπει* τελικά μια αντικειμενοστραφή βάση δεδομένων, παρόλο που στην ουσία χρησιμοποιεί μια σχεσιακή.

Επομένως, το hibernate, γνωρίζοντας την αντιστοιχία μεταξύ λογικής της εφαρμογής και βάσης, κατασκευάζει τις κατάλληλες SQL εντολές οι οποίες και στέλνονται στη βάση δεδομένων. Στη συνέχεια τα αποτελέσματα της βάσης επιστρέφονται ως αντικείμενα της εφαρμογής.

Το hibernate αποτελεί ένα εργαλείο τόσο ευελιξίας όσο και εξοικονόμησης χρόνου. Αυτοματοποιώντας τις βασικές λειτουργίες CRUD (create/read/update/delete) ο προγραμματιστής καταβάλει σημαντικά μικρότερη προσπάθεια και χρόνο για την υλοποίηση της επικοινωνίας της βάσης με το πρόγραμμα. Παράλληλα η συμβατότητα του με διαφορετικές βάσεις δεδομένων με την απαίτηση ελάχιστων παραμετροποιήσεων κάνει το πρόγραμμα μας ευέλικτο.



Εικόνα 8 Επικοινωνία εφαρμογής - βάσης μέσω hibernate

3.3. Βάση δεδομένων (MySQL)

Για την αποθήκευση όλων των απαραίτητων δεδομένων χρησιμοποιούμε μία βάση δεδομένων MySQL (8.0.1) η οποία βρίσκεται στον ίδιο διακομιστή και την επεξεργαζόμαστε μέσω του MySQL Workbench. Η βάση δεδομένων μας περιέχει όλα τα δεδομένα που κρίνονται απαραίτητα για την ορθή λειτουργία της εφαρμογής μας. Να σημειωθεί πως κάποια πεδία που κρίνεται απαραίτητη η ασφάλεια τους αποθηκεύονται κρυπτογραφημένα και διαχειρίζονται από το Spring Security.

3.4. Εργαλεία και βιβλιοθήκες

3.4.1. GitHub

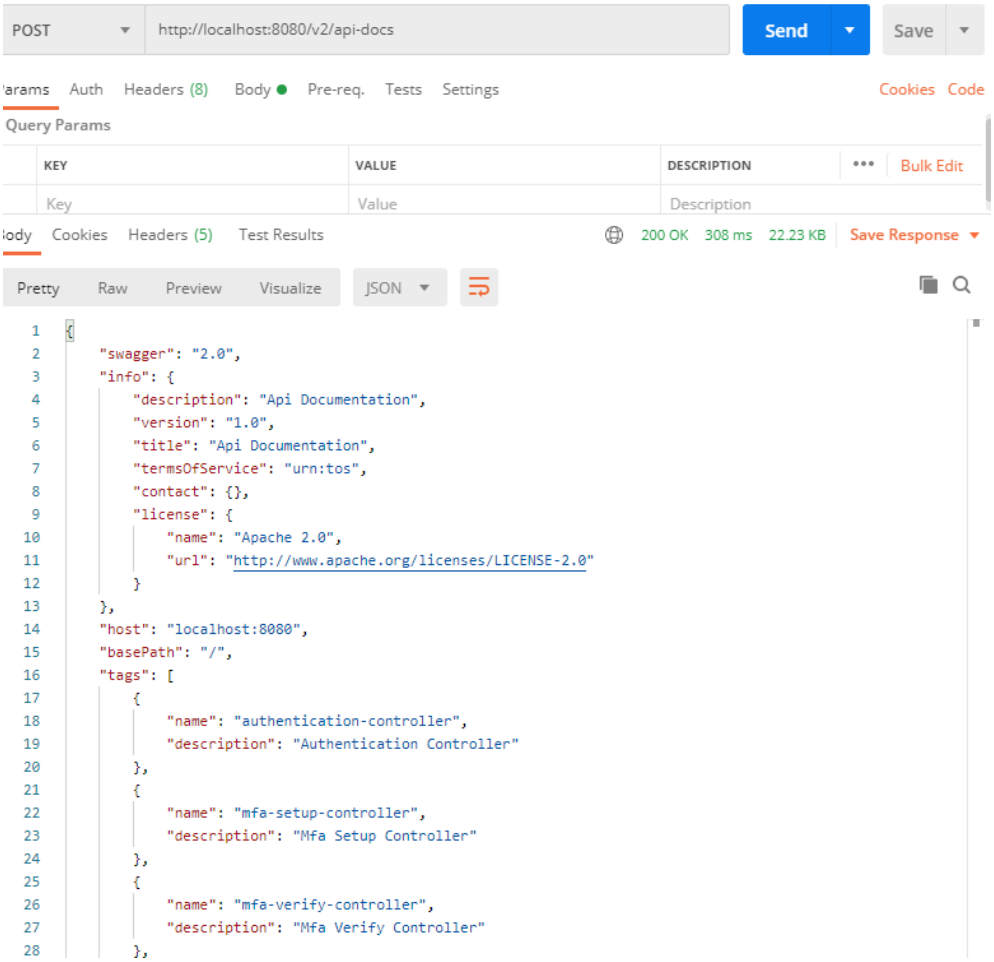
Το GitHub όπως σε κάθε πρότζεκτ έπαιξε καθοριστικό ρόλο στην ασφαλή αποθήκευση του κώδικα μας καθώς και στον εντοπισμό λάθος αλλαγών που έχουν γίνει. Είναι ένα σύστημα ελέγχου εκδόσεων (ή αλλιώς σύστημα ελέγχου πηγαίου κώδικα) με έμφαση στην ταχύτητα, στην ακεραιότητα των δεδομένων και στην υποστήριξη για κατανεμημένες μη γραμμικές ροές εργασίας. Κάθε κατάλογος εργασίας Git είναι ένα ολοκληρωμένο αποθετήριο λογισμικού με πλήρες ιστορικό και δυνατότητες πλήρους παρακολούθησης της έκδοσης ανεξάρτητα από την πρόσβαση δικτύου ή ενός κεντρικού διακομιστή.

3.4.2. Swagger

Το swagger είναι ένα εργαλείο διεπαφής το οποίο χρησιμοποιείται για την εξαγωγή RESTful API ως JSON αρχείο. Σε χρήση με άλλα open-source λογισμικά καταφέραμε να δημιουργήσουμε τα κατάλληλα REST web services στο Frontend μας.

```
<!-- Swagger 2 -->
<!-- https://mvnrepository.com/artifact/io.springfox/springfox-swagger2 -->
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger2</artifactId>
    <version>2.9.2</version>
</dependency>


<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger-ui</artifactId>
    <version>2.9.2</version>
</dependency>
```

Εικόνα 9 Βιβλιοθήκες swagger



Εικόνα 10 Παραγωγή json αρχείου με τα Rest services

4. Καταγραφή λειτουργικών και μη λειτουργικών απαιτήσεων

Στην υλοποίηση ενός έργου, καθοριστική κρίνεται η απόφαση των λειτουργικών και μη απαιτήσεων του συστήματος. Σε αυτό το κεφάλαιο θα αναλυθούν και τα δύο αυτά μέρη και θα προσπαθήσουμε να γίνουν πιο κατανοητά στον αναγνώστη μέσα από uml διαγράμματα.

4.1. Ανάλυση απαιτήσεων συστήματος

Για την ανάλυση απαιτήσεων του πληροφοριακού μας συστήματος μελετήσαμε αντίστοιχα συστήματα τα οποία παρέχουν τις λειτουργίες τους εδώ και αρκετά χρόνια καλύπτοντας τις ανάγκες των πελατών τους. Παράλληλα απαραίτητο κρίθηκε να ληφθεί υπόψιν οι νέες ανάγκες των χρηστών ηλεκτρονικών μέσων οι οποίες προήλθαν από την ραγδαία εξέλιξη της τεχνολογίας.

4.1.1. Λειτουργικές απαιτήσεις

Κύριος γνώμονας για την ανάπτυξη της εφαρμογής μας ήταν η ορθή λειτουργία καθώς και το φιλικό αλλά και εύκολα διαχειρίσιμο από το χρήστη UI. Η εφαρμογή μας καλύπτει όλες τις παρακάτω λειτουργίες και ικανότητες:

Εγγραφή χρήστη

Ο χρήστης θα πρέπει να μπορεί να κάνει εγγραφή στο σύστημα μας εισάγοντας όνομα χρήστη, κωδικό και κάποιες άλλες βασικές πληροφορίες.

QR code

Ο χρήστης πρέπει σκανάρει ένα τυχαία δημιουργημένο qrcode με μια Authenticator εφαρμογή προκειμένου να του παρέχεται ένα επιπλέον επίπεδο ασφάλειας

Δημιουργία πιστοποιητικού

Ο χρήστης πρέπει να μπορεί να δημιουργήσει ένα προσωπικό πιστοποιητικό το οποίο θα μπορούμε να το υπογράψουμε εμείς ως CA (Certificate Authority)

Ανέβασμα αρχείων

Ο χρήστης πρέπει να μπορεί να ανεβάσει αρχεία της μορφής doc και pdf.

Ανέβασμα αρχείων

Ο χρήστης πρέπει να μπορεί να υπογράψει τα αρχεία τα αρχεία που έχουν ανέβει.

Αποστολή αρχείων

Ο χρήστης πρέπει να έχει τη δυνατότητα αποστολής των υπογεγραμμένων αρχείων του με email σε χρήστες ή όχι της εφαρμογής.

Κατέβασμα αρχείων

Ο παραλήπτης θα πρέπει να μπορεί να κατεβάσει το αρχείο μόνο μία φορά ή για περιορισμένο χρονικό διάστημα χωρίς να είναι απαραίτητη η εγγραφή στο σύστημα.

Ακύρωση αποστολής αρχείου

Ο χρήστης πρέπει να έχει τη δυνατότητα να ακυρώσει την αποστολή του σε περίπτωση που το αρχείο δεν έχει ανοιχτεί από τον παραλήπτη.

Προβολή κοινοποιημένων αρχείων

Πρέπει να παρέχεται σελίδα όπου ο χρήστης θα μπορεί να δει τα κοινοποιημένα αρχεία, αν έχουν ανοιχθεί καθώς και ώρα και ημερομηνία προβολής.

Προβολή εισερχόμενων αρχείων

Πρέπει να παρέχεται σελίδα όπου εγγεγραμμένος χρήστης στο σύστημα θα μπορεί να δει όλα τα διαμοιρασμένα προς αυτόν αρχεία.

4.1.2. Μη λειτουργικές απαιτήσεις

Χρηστικότητα

Το πληροφοριακό σύστημα θα προσφέρει γραφικό περιβάλλον επικοινωνίας με το χρήστη. Η χρήση από υπολογιστή θα γίνεται κανονικά ενώ από ταμπλετ με την οθόνη αφής.

Ασφάλεια

Το πληροφοριακό σύστημα θα μπορεί να χρησιμοποιηθεί από άτομα των οποίων οι λογαριασμοί έχουν δημιουργηθεί. Τα επίπεδα χρηστών που θα υπάρχουν είναι τα εξής:

1. Ο διαχειριστής συστήματος
2. Απλός χρήστης

Κάθε επίπεδο χρηστών θα έχει τα αντίστοιχα δικαιώματα που απαιτούνται για την ομαλή εργασία του.

Αξιοπιστία

Το πληροφοριακό σύστημα θα πρέπει να λειτουργεί ομαλά και να μη διακινδυνεύει απώλεια προσωπικών δεδομένων των χρηστών ή προσωπικά τους αρχεία.

Ορθότητα

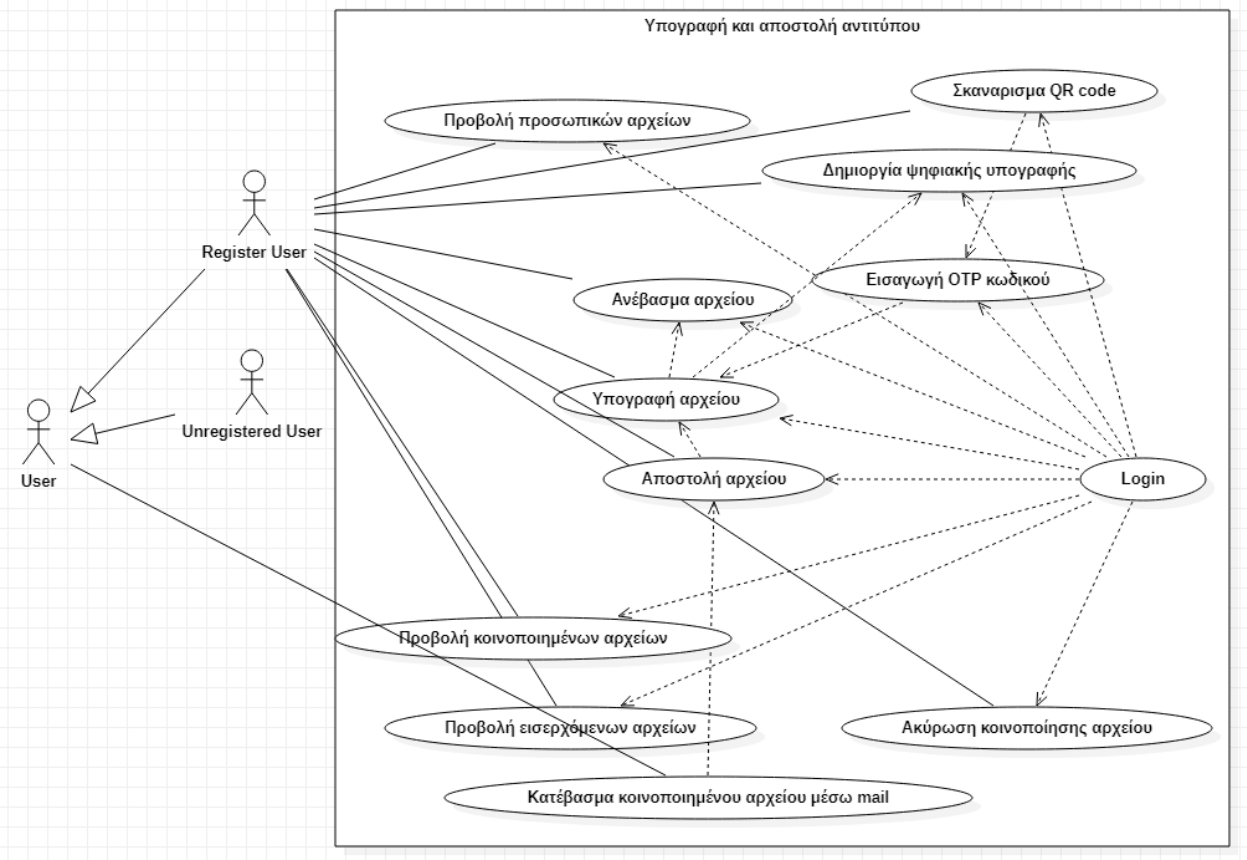
Το πληροφοριακό σύστημα θα πρέπει να συμπεριφέρεται σύμφωνα με τις απαιτήσεις που έχουν αναφερθεί και να παράγει ορθά αποτελέσματα.

Επεκτασιμότητα

Το πληροφοριακό σύστημα θα πρέπει να είναι ικανό να αναπτυχθεί και να επεκταθεί. Οι τεχνολογίες που θα επιλεχθούν είναι απαραίτητο να προσφέρουν τόσο την σωστή λειτουργία σήμερα όσο και να έχει τη δυνατότητα να υποστηρίξει μελλοντικές ανάγκες όπως:

1. Αύξηση των χρηστών που εξυπηρετεί
2. Υπογραφή νέων τύπων αρχείων
3. Δημιουργία νέων ρόλων όπου θα ανατεθούν διαφορετικοί συνδυασμοί δικαιωμάτων

4.2. Διάγραμμα περιπτώσεων χρήσης



Εικόνα 11 Διάγραμμα περιπτώσεων χρήσης

Υπογραφή και αποστολή αντιτύπου

Προϋποθέσεις:

- Ο χρήστης έχει κάνει επιτυχή είσοδο στο σύστημα.
- Ο χρήστης έχει σκανάρει επιτυχώς τον QR code με μία Authenticator εφαρμογή στην πρώτη επιτυχή είσοδο στο σύστημα
- Ο χρήστης έχει δημιουργήσει επιτυχώς ψηφιακή υπογραφή.

Βασική Ροή.

1. Ο χρήστης ανεβάζει ένα προσωπικό αρχείο
2. Ο χρήστης επιλέγει το αρχείο για υπογραφή
3. Το σύστημα ανακατευθύνει τον χρήστη σε παράθυρό όπου του ζητάει τον otp κωδικό του
4. Ο χρήστης βάζει τον otp κωδικό
5. Το σύστημα ελέχει τον otp κωδικό

6. Το σύστημα ανακατευθύνει τον χρήστη σε παράθυρο όπου ζητάει τον κωδικό της ψηφιακής υπογραφής του χρήστη
7. Ο χρήστης εισάγει τον κωδικό της ψηφιακής του υπογραφής
8. Το σύστημα ελέγχει τον κωδικό
9. Το αρχείο υπογράφεται επιτυχώς
10. Το σύστημα προβάλει τα αρχεία του χρήστη με ενεργοποιημένο το κουμπί αποστολή σε αυτά που είναι υπογεγραμμένα
11. Ο χρήστης πατάει το κουμπί αποστολή στο αρχείο που επιθυμεί
12. Αναδύεται κατάλληλα διαμορφωμένα παράθυρο για εισαγωγή email παραλήπτη
13. Ο χρήστης εισάγει το email που επιθυμεί να αποσταλεί το αρχείο
14. Το αρχείο αποστέλλεται επιτυχώς

Εναλλακτικές Ροές

* Σε οποιοδήποτε σημείο το λογισμικό καταρρέει

    1. Ο διαχειριστής εκκινεί το σύστημα
    2. Ο χρήστης οδηγείται στη σελίδα εισόδου

1α Ο χρήστης ανεβάζει αρχείο διαφορετικής μορφής αρχείο από τις αποδεκτές

1. Το αρχείο δεν γίνεται αποδεκτό από το σύστημα

2. Το σύστημα βγάζει μήνυμα λάθους στο χρήστη

3. Ο χρήστης επιστρέφει στο βήμα 1 της κανονικής ροής και μπορεί να ανεβάσει εκ νέου αρχείο

5α Ο χρήστης βάζει λάθος otp κωδικό

1. Το σύστημα δεν ταυτοποιεί επιτυχώς τον κωδικό

2. Το σύστημα προβάλει μήνυμα λάθους

3. Ο χρήστης οδηγείται ξανά στην εισαγωγή otp κωδικού

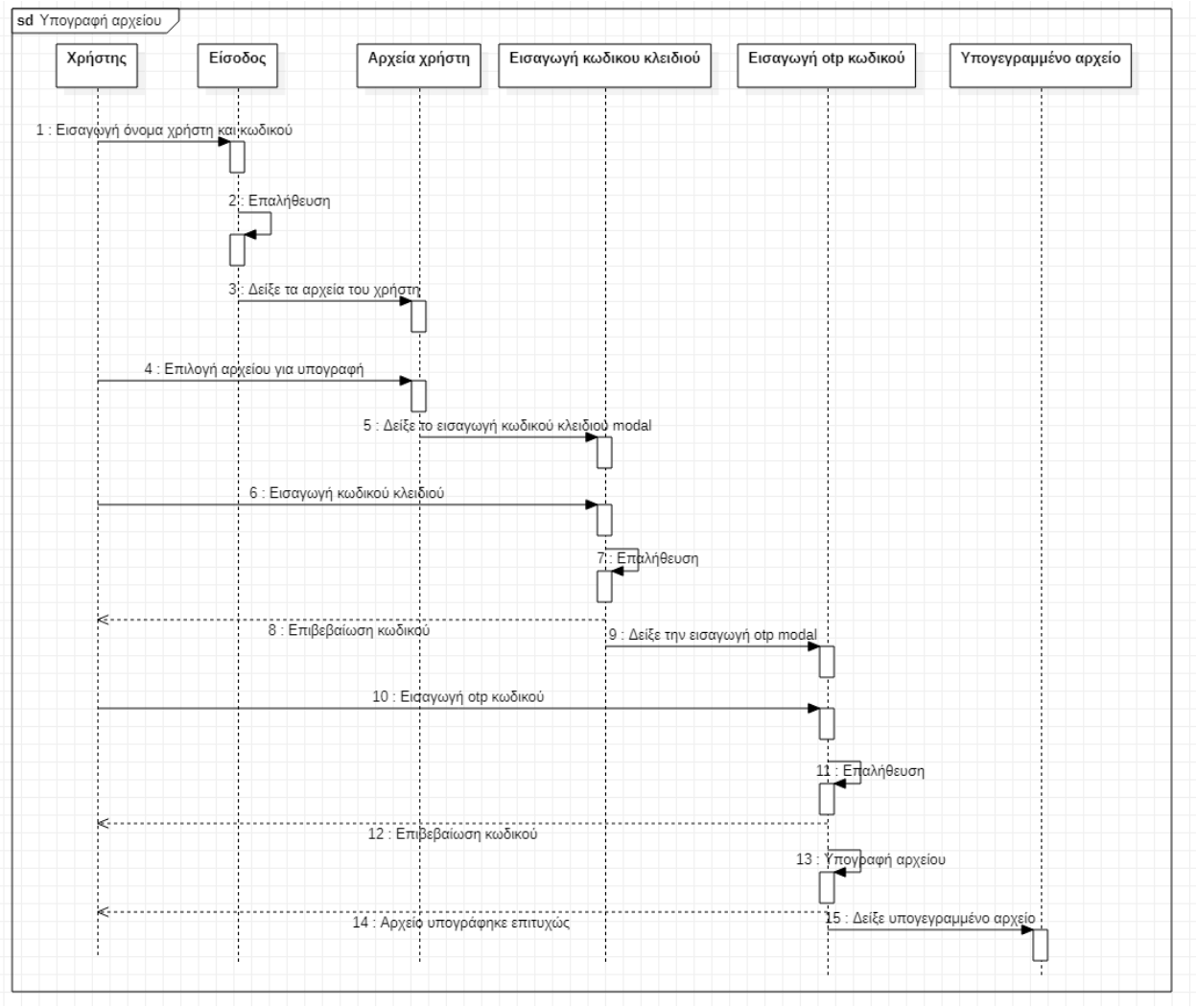
8α Ο χρήστης βάζει λάθος κωδικό ψηφιακής υπογραφής

1. Το σύστημα δεν ταυτοποιεί επιτυχώς τον κωδικό

2. Το σύστημα προβάλει μήνυμα λάθους

3. Ο χρήστης οδηγείται ξανά στην εισαγωγή κωδικού ψηφιακής υπογραφής

4.3. Διάγραμμα ακολουθίας



Εικόνα 12 Διάγραμμα ακολουθίας

Σε αυτό το διάγραμμα ακολουθίας περιλαμβάνεται όλη η πορεία για την υπογραφή ενός ανεβασμένου αρχείου. Αρχικά ο χρήστης κάνει είσοδο στο σύστημα και μεταφέρεται στη σελίδα με τα αρχεία του. Στη συνέχεια επιλέγει ένα αρχείο που δεν έχει υπογραφεί και του εμφανίζονται κατά σειρά οι σελίδες «Εισαγωγή κωδικός κλειδιού» και η «Εισαγωγή otp κωδικού». Σε περίπτωση σωστής καταχώρησης κωδικών ο χρήστης ενημερώνεται για την επιτυχή υπογραφή του αρχείου και οδηγείται στην αντίστοιχη σελίδα.

5. Η ανάπτυξη του συστήματος

Το κεφάλαιο αυτό ξεκινάει με την ανάλυση των απαιτήσεων του συστήματος μας και τι λειτουργίες και αυτοματισμούς περιμένουμε από αυτό. Στη συνέχεια για να γίνουν κατανοητές οι βασικές λειτουργίες θα δούμε κάποια διαγράμματα περιπτώσεων χρήσης και αντικειμένων. Τέλος θα δούμε πως τα κομμάτια που αναφέραμε «συνδέονται» μεταξύ τους καθώς και τις βασικές λειτουργίες και πως αυτές υλοποιούνται στον κώδικα μας.

5.1. Δημιουργία self-signed CA πιστοποιητικού

Χρησιμοποιώντας το openssl δημιουργούμε το Root Key. Με αυτό το κλειδί μπορούν να υπογράφονται πιστοποιητικά εκ μέρους μας.

```
openssl genrsa -des3 -out rootCA.key 4096
```

Στη συνέχεια χρησιμοποιώντας το κλειδί μας (rootCA.key) θα δημιουργήσουμε ένα δικό μας πιστοποιητικό. Το συγκεκριμένο πιστοποιητικό προσθέτοντας το είτε σε έναν περιηγητή είτε στο περιβάλλον των windows όλα τα πιστοποιητικά που θα υπογραφούν από εμάς θα αναγνωρίζονται ως ασφαλή. Με την εκτέλεση της εντολής

```
openssl req -x509 -new -nodes -key rootCA.key -sha256 -days 1024 -out rootCA.crt
```

έχει δημιουργηθεί το πιστοποιητικό μας rootCA.crt.

Εικόνα 13 Το κλειδί/πιστοποιητικό του CA

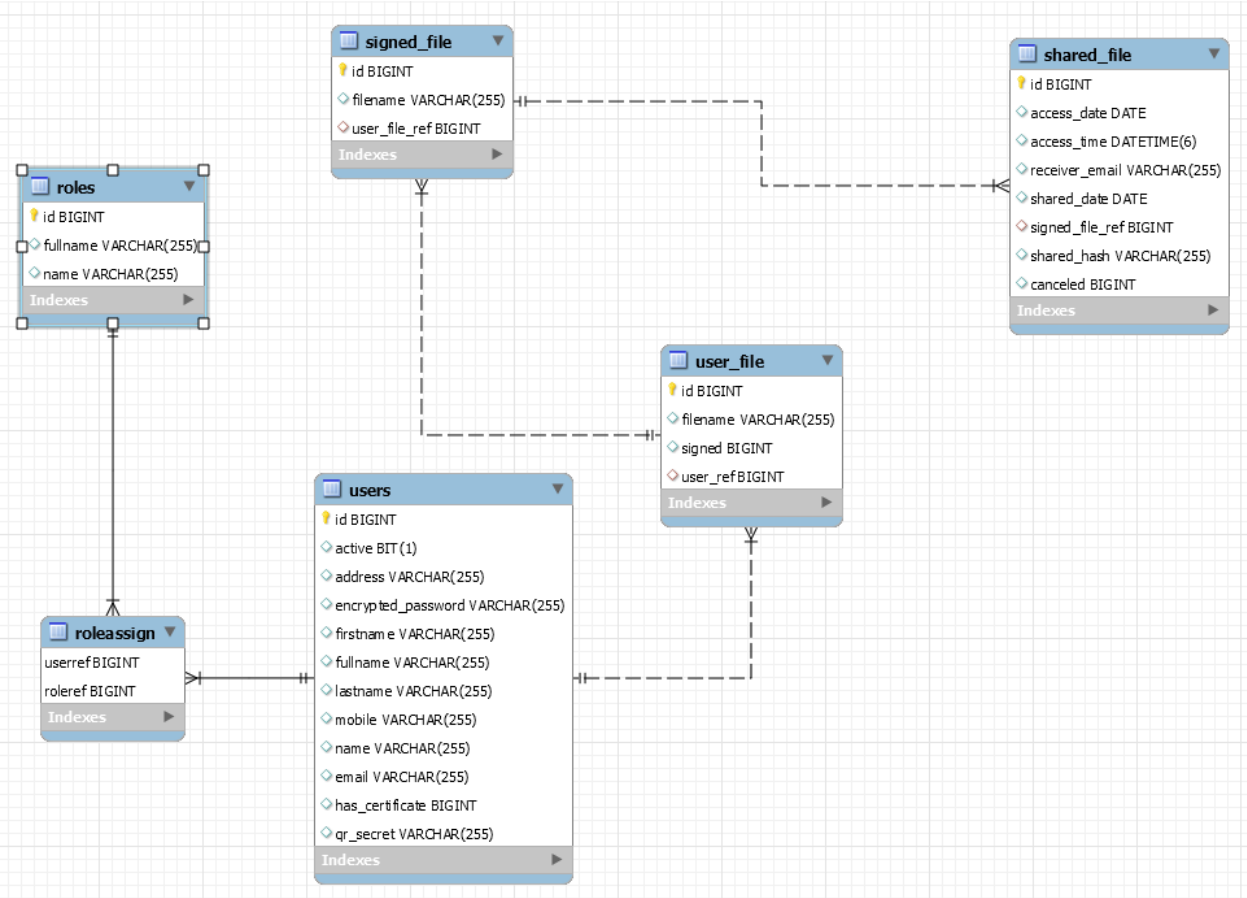
5.2. Το web API

Αρχικά θα αναλύσουμε το backend της εφαρμογής το οποίο αποτελεί και το service layer (ή αλλιώς web API) και είναι γραμμένο σε Spring Boot όπως ήδη αναφέραμε. Το κομμάτι αυτό της εφαρμογής δεν προσφέρει απλά τα service και την επικοινωνία με τη βάση αλλά περιέχει και το μεγαλύτερο μέρος τη λογικής που είναι απαραίτητη για την ορθή εκτέλεση της εφαρμογής μας. Θα δούμε συνοπτικά τους βασικότερους controllers (κλάσεις που διαχειρίζονται κλήσεις HTTP

σχετικά το σύστημα μας) και θα αναλύσουμε καθοριστικά για την εφαρμογή κομμάτια όπως η δημιουργία κλειδιών/πιστοποιητικών, η παραγωγή του jwt token και η υπογραφή αρχείων.

5.2.1. Models

Ξεκινώντας, προκειμένου να κατανοήσουμε τα βασικά μέρη της εφαρμογής μας θα δούμε ένα διάγραμμα (object diagram) το οποίο περιέχει τα models που χρειάστηκαν για την εφαρμογή καθώς και την μεταξύ τους σχέση.



Εικόνα 14 Διάγραμμα κλάσεων

Όπως φαίνεται και στο σχήμα το Web API μας αποτελείται από έξι βασικές κλάσεις. Οι ίδιοι πίνακες υπάρχουν και πίσω στη βάση μας.

Αρχικά υπάρχει ο χρήστης ο οποίος περιέχει κάποια βασικά στοιχεία όπως ονοματεπώνυμο, τηλέφωνο και email. Επίσης περιέχει τα properties has_certificate, qr_secret τα οποία μας

ενημερώνουν αν ο χρήστης έχει δημιουργήσει το πιστοποιητικό του και αν έχει σκανάρει το qr_code του αντίστοιχα.

Οι κλάσεις roleassign και roles μας επιτρέπουν να δώσουμε ρόλους στο χρήστη και να τον διαχειριστούμε σύμφωνα με αυτόν.

Στη συνέχεια παρατηρούμε πως ο users έχει oneToMany σχεση με το user_file. Αυτό σημαίνει ότι ένας χρήστης έχει μηδέν ή περισσότερα ανεβασμένα αρχεία.

Ο user_file «δένεται» με τον signed_file όπου ουσιαστικά δηλώνεται αν το αρχείο έχει υπογραφεί. Να σημειώσουμε πως κάθε αρχείο μπορεί να υπογραφεί μόνο μία φορά και γι αυτό ορίζουμε την μεταξύ τους σχέση ως OneToOne.

Αξίζει να παρατηρήσουμε τα πεδία της τελευταίας μας κλάσης, της shared_file. Μία shared_file εγγραφή δημιουργείται όταν ο χρήστης επιλέξει να αποστείλει ένα υπογεγραμμένο αρχείο. Περιέχει τα εξής πεδία:

- access_Date: Η ημέρα που προβλήθηκε το αρχείο από τον παραλήπτη
- access_Time: Η ώρα που προβλήθηκε το αρχείο από τον παραλήπτη
- access_Email: Το email του παραλήπτη
- shared_Date: Η ημερομηνία αποστολής του αρχείο
- shared_Hash: Μοναδικό hash το οποίο δημιουργείται για τον χαρακτηρισμό του Shared_File
- canceled: Μεταβλητή που μας δείχνει αν ακυρώθηκε η αποστολή του αρχείου

Τέλος παρατηρούμε ότι η signed_file κλάση δένεται με την κλάση shared_file ως oneToMany. Αυτό συμβαίνει διότι ένα υπογεγραμμένο αρχείο μπορεί να αποσταλεί σε περισσότερους από έναν παραλήπτες

5.2.2. Controllers

Σε ένα restful Service ορίζονται ως οι μέθοδοι μιας κλάσης που αναλαμβάνουν να χειριστούν κάθε request. Κάθε μέθοδος χαρακτηρίζεται με ένα annotation που ορίζει αν η μέθοδος στο http request θα αντιστοιχεί σε GET, POST, PUT ή DELETE. Παρακάτω βλέπουμε αναλυτικά τις βασικές μεθόδους που περιλαμβάνουν οι controllers μας:

Όνομα μεθόδου	Annotation	URI	Λειτουργία
signUpUser	@PostMapping	/signup	Η μέθοδος αυτή λαμβάνει ως παράμετρο ένα signUpRequest αντικείμενο το οποίο περιλαμβάνει το όνομα και τον κωδικό του χρήστη που εγγράφεται.
authenticateUser	@PostMapping	/login	Η μέθοδος αυτή λαμβάνει ως παράμετρο το όνομα χρήστη και τον κωδικό του χρήστη ώστε να τον ταυτοποιήσει.
createCertificate	@PostMapping	/createCertificate	Η μέθοδος καλείται όταν ο χρήστης επιλέξει να δημιουργήσει το πιστοποιητικό του. Δέχεται ως παραμέτρους τον κωδικό και βασικά στοιχεία που θα περιέχει το πιστοποιητικό.
hasUserCertificate	@GetMapping	/hasUserCertificate	Η μέθοδος αυτή επιστρέφει αν ο χρήστης έχει δημιουργήσει πιστοποιητικό.
hasQrSecret	@GetMapping	/hasQrSecret	Η μέθοδος αυτή επιστρέφει αν ο χρήστης έχει σκαναρει το qr

			code για otp authentication.
setupQrCode	@GetMapping	/mfa/setup	Η μέθοδος αυτή δημιουργεί και επιστρέφει ένα qrCode προκειμένου να το σκανάρει ο χρήστης με το κινητό του
saveFile	@ PostMapping	/userFiles	Η μέθοδος αυτή δέχεται ως παράμετρο ένα αρχείο από το χρήστη και το αποθηκεύει.
downloadFile	@GetMapping	/ downloadFile	Η μέθοδος αυτή επιστρέφει στο χρήστη το αρχείο που έχει επιλέξει.
signUserFile	@PostMapping	/userFiles/signFile/{password}	Η μέθοδος αυτή δέχεται ως παραμέτρους τον κωδικό του προσωπικού κλειδιού του και το αρχείο που θέλει να υπογράψει.
sharefile	@PostMapping	/sharefile	Η μέθοδος αυτή δέχεται ως παραμέτρους ένα υπογεγραμμένο αρχείο και το email του παραλήπτη.
cancelSharedfile	@PostMapping	/cancelSharedFileSent	Η μέθοδος αυτή δέχεται ως παράμετρο ένα αρχείο που είχε επιλέξει για

			διαμοιρασμό και ακυρώνει την αποστολή του.
userFiles	@GetMapping	/loggedInUserUserFiles	Η μέθοδος αυτή επιστρέφει μια λίστα με τα αρχεία που έχει ανεβάσει ο χρήστης.
sharedFiles	@GetMapping	/sharedFiles	Η μέθοδος αυτή επιστρέφει όλους τους διαμοιρασμούς που έχει κάνει ο χρήστης.
getAllSharedFilesToMeByEmail	@GetMapping	/sharedFilesByEmail	Η μέθοδος αυτή επιστρέφει όλα τα αρχεία που έχουν διαμοιραστεί στο χρήστη βάση του προσωπικού του email.

5.2.3 One Time Password (OTP)

Ο αλγόριθμος αυτός όπως ήδη εξηγήσαμε είναι υπεύθυνος για τη δημιουργία κωδικών με περιορισμένη χρονική διάρκεια. Τον χρησιμοποιήσαμε προκειμένου να παρέχουμε ένα επιπλέον επίπεδο ασφάλειας στην υπογραφή ενός αρχείου. Με αυτόν τον τρόπο ο χρήστης πρέπει να παρέχει σε πρώτη φάση τον κωδικό του προσωπικού του κλειδιού και στη συνέχεια τον otp κωδικό του.

Για την δημιουργία του χρησιμοποιήσαμε την παρακάτω βιβλιοθήκη:

```xml
<dependency>
    <groupId>dev.samstevens.totp</groupId>
    <artifactId>totp-spring-boot-starter</artifactId>
    <version>1.7.1</version>
</dependency>
```

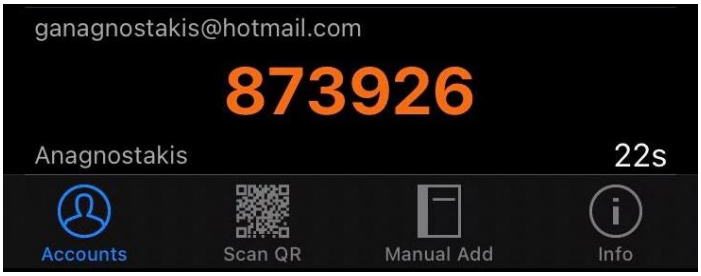
Στη συνέχεια παράγουμε την εικόνα (qr code) με τον παρακάτω κώδικα:

```
QrData data = qrDataFactory.newBuilder()
    .label(loggedInUser.getEmail())
    .secret(secret)
    .issuer(loggedInUser.getUsername())
    .build();

// Generate the QR code image data as a base64 string which
String qrCodeImage = getDataUriForImage(
    qrGenerator.generate(data),
    qrGenerator.getImageMimeType()
);

loggedInUser.setQrSecret(secret);
userService.updateUser(loggedInUser);

Image image = new Image();
image.setImageUri(qrCodeImage);
```

Στη συνέχεια ο χρήστης βλέπει το qr code και πρέπει να το σκανάρει με μία εφαρμογή 2-step authenticator όπως η Microsoft Authenticator.

Εικόνα 15 QR Code

Τέλος μέσω της εφαρμογής κάθε 30 δευτερόλεπτα έχει έναν νέο, τυχαίο αριθμό ο οποίος δεν έχει καμία σχέση με τον προηγούμενο.



Εικόνα 16 Otp κωδικός

5.2.4. Json Web Token (JWT)

Για την ασφάλεια του api μας απαραίτητη κρίνεται η χρήση του jwt προκειμένου να αυθεντικοποιηθεί ο χρήστης και να απαλλαγούμε από τη χρήση cookies ή άλλων session data. Το token αυτό δημιουργείται στο web api μας ύστερα από την επιτυχή αυθεντικοποίηση του χρήστη και επιστρέφεται σε αυτόν. Περιέχει βασικά δεδομένα του χρήστη, τον αλγόριθμο που δημιουργήθηκε και αρκετά άλλα. Παρακάτω φαίνεται όλη η δομή του:

Header	`{ "alg": "HS256", "typ": "JWT" }`	Identifies which algorithm is used to generate the signature `HS256` indicates that this token is signed using HMAC-SHA256. Typical cryptographic algorithms used are HMAC with SHA-256 (HS256) and RSA signature with SHA-256 (RS256). JWA (JSON Web Algorithms) RFC 7518 introduces many more for both authentication and encryption.[8]
Payload	`{ "loggedInAs": "admin", "iat": 1422779638 }`	Contains a set of claims. The JWT specification defines seven Registered Claim Names which are the standard fields commonly included in tokens.[1] Custom claims are usually also included, depending on the purpose of the token. This example has the standard Issued At Time claim (`iat`) and a custom claim (`loggedInAs`).
Signature	`HMAC-SHA256( secret, base64urlEncoding(header) + '.' + base64urlEncoding(payload) )`	Securely validates the token. The signature is calculated by encoding the header and payload using Base64url Encoding and concatenating the two together with a period separator. That string is then run through the cryptographic algorithm specified in the header, in this case HMAC-SHA256. The *Base64url Encoding* is similar to base64, but uses different non-alphanumeric characters and omits padding.

Εικόνα 17 Δομή JWT token

Τα τρία αυτά μέρη κωδικοποιούνται ξεχωριστά και δημιουργούν το JWT. Ένα παράδειγμα δικού μας token είναι το εξής:

```
eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiIxMCIsInJvbGVzIjpbIlJPTEVfVVNFUiJdLCJ1c2VybmFtZSI6InRlc3QxMDAiLCJ1c2VyIjp7ImlkIjoxMCwidXNlcm5hbWUiOiJ0ZXN0MTAwIiwiZmlyc3ROYW1lIjpudWxsLCJsYXN0TmFtZSI6bnVsbCwiZnVsbE5hbWUiOm51bGwsIm1vYmlsZSI6IjY5NTg1NDg1NCIsImFkZHJlc3MiOiJQbGFwb3V0YSAyNSIsImVtYWlsIjoidGVzdDEwMEB0ZXN0LmdyIiwiYWN0aXZlIjp0cnVlLCJoYXNDZXJ0aWZpY2F0ZSI6bnVsbCwicXJTZWNyZXQiOm51bGwsInJvbGVzIjpbeyJpZCI6MSwibmFtZSI6IlVTRVIiLCJkZXNjcmlwdGlvbiI6IlVTRVIifV19LCJpYXQiOjE2MTEwNTgxNDksImV4cCI6MTYxMjQ5ODE0OX0.YpyxU00fY12_yT0fRJIbpitzhHwW2PsQRalBS_8C0ijxn1rUAbd5SSPvFXPMjG89yAhG9bKOu5gd6Mg8WhYwoA
```

και περιέχει τα παρακάτω δεδομένα

HEADER: ALGORITHM & TOKEN TYPE


```
{
  "alg": "HS512"
}
```


PAYLOAD: DATA


```
{
  "sub": "10",
  "roles": [
    "ROLE_USER"
  ],
  "username": "test100",
  "user": {
    "id": 10,
    "username": "test100",
    "firstName": null,
    "lastName": null,
    "fullName": null,
    "mobile": "695854854",
    "address": "Plapouta 25",
    "email": "test100@test.gr",
    "active": true,
    "hasCertificate": null,
    "qrSecret": null,
    "roles": [
      {
        "id": 1,
        "name": "USER",
        "description": "USER"
      }
    ]
  },
  "iat": 1611058149,
  "exp": 1612498149
}
```


5.2.5. Δημιουργία κλειδιού – πιστοποιητικού χρήστη

Όπως αναφέραμε παραπάνω, δίνεται στο χρήστη η δυνατότητα να δημιουργήσει το δικό του πιστοποιητικό. Οι προϋποθέσεις για τη δημιουργία είναι ένας κωδικός και κάποια βασικά στοιχεία που θα περιέχει το πιστοποιητικό. Η διαδικασία ολοκληρώνεται σε δύο φάσεις.

Αρχικά δημιουργούμε τους κατάλληλους φακέλους που αντιστοιχούν σε αυτόν τον χρήστη και θα αποθηκευτούν τα κλειδιά – πιστοποιητικά με τις παρακάτω εντολές.

```
File directory = new File("fileStorage");
File userDirectory = new File("fileStorage"+ File.separator + user.getId() + File.separator + "certificate");

if (!userDirectory.exists()) {
   userDirectory.mkdirs();
}
```

Στη συνέχεια χρησιμοποιώντας την γραμμή εντολών μέσα από το spring boot πρόγραμμα μας κάνουμε τις παρακάτω ενέργειες:

1. Δημιουργούμε ένα 4096 bit RSA key (ιδιωτικό κλειδί)
2. Δημιουργούμε ένα αίτημα υπογραφής πιστοποιητικού όπου εισάγουμε τις βασικές πληροφορίες του χρήστη και το κλειδί
3. Ως CA υπογράφουμε το αίτημα με αποτέλεσμα τη δημιουργία του πιστοποιητικού.
4. Τέλος δημιουργούμε ένα .pfx αρχείο με το οποίο υπογράφουμε τα έγγραφα (pdf, doc) και αποθηκεύουμε όλα τα κλειδιά στους κατάλληλους φακέλους.

Οι γραμμές κώδικα που χρησιμοποιήθηκαν για την δημιουργία όλων των προαναφερθέντων είναι οι παρακάτω:

```
ProcessBuilder builder = new ProcessBuilder(
            "cmd.exe", "/C", "openssl genrsa -out " + keyName + " 4096 &&" +
            "openssl req -new -sha256 -key " + keyName + " -subj
\"/emailAddress="+user.getEmail()+"/C=US/ST=CA/O=MyOrg, Inc./CN=mydomain.com\" -out " + csrName +
" && " +
            "openssl x509 -req -in " + csrName + " -CA ca.crt -CAkey ca.key -CAcreateserial -out " + crtName + " -
days 500 -sha256 && " +
            "openssl pkcs12 -export -in ca.crt -out " + pfxName + " -inkey " + keyName + " -in " + crtName + " -
password pass:"+ password
            );
builder.directory(new File(String.valueOf(directory)));
builder.redirectErrorStream(true);
Process p = builder.start();
BufferedReader r = new BufferedReader(new InputStreamReader(p.getInputStream()));
String line;
while (true) {
   line = r.readLine();
   if (line == null) { break; }
}
```

5.2.6. Υπογραφή αρχείου

Για την διαδικασία της υπογραφής ενός αρχείου χρησιμοποιήσαμε τη βιβλιοθήκη groupDocs-signature η οποία μας απλοποιεί κάποια βήματα με απλές εντολές.

```
<!-- https://mvnrepository.com/artifact/com.groupdocs/groupdocs-conversion -->
<dependency>
    <groupId>com.groupdocs</groupId>
    <artifactId>groupdocs-signature</artifactId>
    <version>20.9</version>
</dependency>
```

Εικόνα 18 Βιβλιοθήκη groupdocs-signature

Με τη χρήση της, η όλη ροή για την υπογραφή του αρχείου γίνεται με τα παρακάτω βήματα:

1. Φορτώνουμε την υπογραφή (το pfx αρχείο)
2. Δημιουργούμε το «παραθυράκι» όπου θα εμφανίζεται η υπογραφή, γράφοντας μέσα βασικές πληροφορίες, εισάγοντας φωτογραφία ως φόντο και επιλέγοντας τη θέση του στο αρχείο μας
3. Υπογράφουμε το αρχείο
4. Αποθηκεύουμε το αρχείο στον κατάλληλο φάκελο
5. Ενημερώνουμε τη βάση ότι το αρχείο έχει υπογραφεί επιτυχώς.

Ο παρακάτω κώδικας περιλαμβάνει όλη τη διαδικασία που αναφέραμε:

```
String outputFilePath = new File("sign" + userFile.getFilename()).getPath();
try {
   Signature signature = new Signature(fileName);

   DigitalSignOptions options = new DigitalSignOptions(certificateFile.toString());

   // optional: setup image file path
   options.setImageFilePath("C:\\Users\\Anagnostakis\\Desktop\\Papei
Projects\\Diplomatikh\\SpringBoot\\unipi.jpg");
   options.setReason("");
   options.setContact(user.getMobile());
   options.setLeft(50);
   options.setTop(50);
   options.setWidth(300);
   options.setHeight(120);
   options.setPageNumber(1);
   options.setPassword(password);
   options.getExtensions().add(new SpreadsheetPosition(10,10));
   // sign document to file
   signature.sign(outputFilePath, options);
   System.out.print("\nSource document signed successfully.\nFile saved at " + outputFilePath);
```

```
  Files.move
        (Paths.get(outputFilePath),
              Paths.get(signedFileDirectory+ File.separator + outputFilePath));


  // mark file as Signed and update
  userFile.setSigned(1L );
  UserFile newUserFile = userFileService.updateUserFile(userFile);

  SignedFile signedFile = new SignedFile();
  signedFile.setFilename(outputFilePath);
  signedFile.setUserFile(newUserFile);
  signedFileService.createSignedFile(signedFile);
}
```

5.2.7 Αποστολή email και διαμοιρασμός αρχείου

Για την αποστολή email χρησιμοποιήσαμε τη βιβλιοθήκη “spring-boot starter-mail”

```
<!-- send email -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-mail</artifactId>
</dependency>
```

Εικόνα 19 Βιβλιοθήκη spring boot starter mail

Στη συνέχεια στο application.properties αρχείο μας θέσαμε τις απαραίτητες ρυθμίσεις αφού πρώτα σετάραμε το email μας.

```
spring:

  mail:
    host: smtp.gmail.com
    port: 587
    username: geor.anagnostakis@gmail.com
    password: qzso[redacted]ez
    properties:
      mail:
        smtp:
          starttls:
            enable: true
```

Εικόνα 20 Ρυθμίσεις για αποστολή email

Για τον διαμοιρασμό του αρχείου δημιουργούμε μία εγγραφή στον πίνακα Shared_Files. Δημιουργούμε ένα τυχαίο αλφαριθμητικό 25 χαρακτήρων το οποίο χαρακτηρίζει την συγκεκριμένη εγγραφή και είναι εφικτή η ταυτοποίηση του αρχείου που έχει διαμοιραστεί και η ενημέρωση της εγγραφής. Παρακάτω θα δούμε τον κώδικα δημιουργίας του hash καθώς και δημιουργίας/αποστολής του “shared_file”.

```
static String getAlphaNumericString(int n)
{
   // chose a Character random from this String
   String AlphaNumericString = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
         + "0123456789"
         + "abcdefghijklmnopqrstuvxyz";

   // create StringBuffer size of AlphaNumericString
   StringBuilder sb = new StringBuilder(n);

   for (int i = 0; i < n; i++) {
      // generate a random number between
      // 0 to AlphaNumericString variable length
      int index
            = (int)(AlphaNumericString.length()
            * Math.random());

      // add Character one by one in end of sb
      sb.append(AlphaNumericString
            .charAt(index));
   }
   return sb.toString();
}
```

Δημιουργία hash 25 αλφαριθμητικών χαρακτήρων

```
public void sendEmailWithFile(UserFile userFile, String email) {

   String hash = getAlphaNumericString(25);
   String url="http://localhost:8080/downloadsharedfile/" + hash;
   SharedFile sharedFile = new SharedFile();
   sharedFile.setSharedDate(new Date());
   sharedFile.setReceiverEmail(email);
   sharedFile.setSignedFile(userFile.getSignedFile());
   sharedFile.setSharedHash(hash);

   sharedFileService.createSharedFile(sharedFile);

   sendEmailService.sendEmailForSharedFile(email, url);
}
```

Δημιουργία και αποστολή shared_file

5.3. Angular – Frontend

Αυτό το κομμάτι της εφαρμογής περιλαμβάνει όλο το interface με το οποίο ο χρήστης αλληλοεπιδρά με την εφαρμογή μας. Η angular εκτός των σελίδων που προβάλλονται στο χρήστη μας δίνει τη δυνατότητα να συμπεριλάβουμε και εδώ κάποια κομμάτια λογικής καθώς και να διαχειριστούμε τα request με ασφάλεια. Στη συνέχεια θα μελετήσουμε κάποια κομμάτια που αποτελούν καθοριστικό ρόλο στην σωστή και ασφαλή εκτέλεση της εφαρμογής.

5.3.1. Jwt token

Η όλη δημιουργία του jwt token γίνεται στο API πίσω και επιστρέφεται στην angular εφαρμογή μας. Στη συνέχεια διαβάζονται τα στοιχεία του και αποθηκεύονται μαζί με το token στο localstorage. Έτσι ξεκινάει το session του χρήστη .

```
setSession(accessToken: any): JwtPayload {
  const decodedToken = this.getDecodedAccessToken(accessToken);
  const jwtPayload: JwtPayload = {
    roles: decodedToken.roles,
    username: decodedToken.username,
    user: decodedToken.user,
    iat: decodedToken.iat,
    exp: decodedToken.exp,
    accessToken
  };
  localStorage.setItem('currentJwtPayload', JSON.stringify(jwtPayload));
  this.currentJwtPayLoadSubject.next(jwtPayload);
  return jwtPayload;
}
```

Εικόνα 21 Η αποθήκευση token και των πληροφοριών του χρήστη στο localstorage

5.3.2. Interceptor και http requests

Η ανάγκη για ασφάλεια των request που γίνονται και ταυτοποίηση του χρήστη απαιτεί την αποστολή του jwt token σε κάθε κλήση του web api μας. Αυτό γίνεται με τη χρήση του HttpInterceptor, ένα interface το οποίο είναι διαθέσιμο από την Angular 4.3. Η λειτουργία του είναι, με λίγες γραμμές κώδικα, να περιλαμβάνει σε κάθε http request το jwt token οδηγώντας σε μία εύκολη ταυτοποίηση (Authorization) του χρήστη.

Η παραπάνω καθοριστική λειτουργία για την εφαρμογή μας γίνεται με τις παρακάτω γραμμές κώδικα.

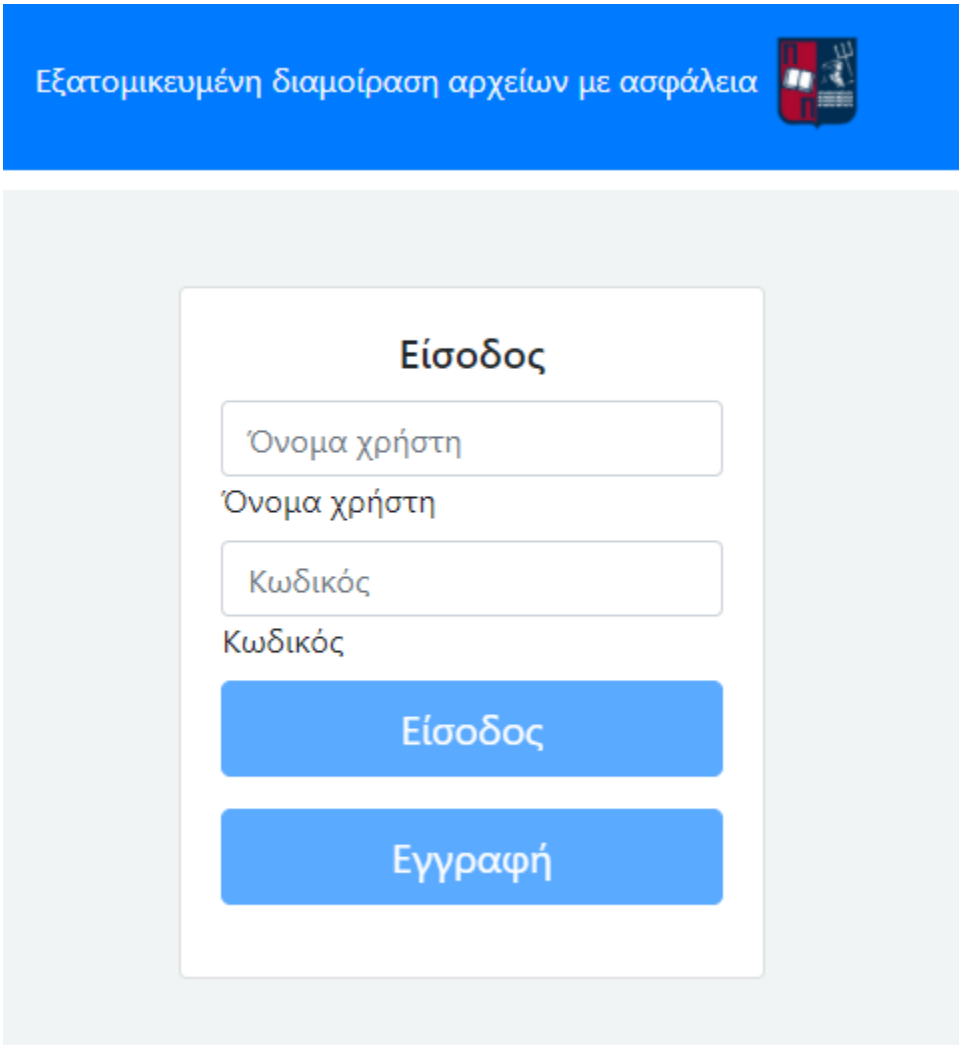
```
intercept(request: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<an
{
 const currentJwtPayload = JSON.parse(localStorage.getItem('currentJwtPayload'));
    if (currentJwtPayload) {
      request = request.clone({
        setHeaders: {
          Authorization: `Bearer ${currentJwtPayload.accessToken}`
        }
      });
    } return next.handle(request);
  }
```

Με αυτό τον τρόπο σε κάθε κλήση στους controllers μας περιλαμβάνεται στο Authorization header το jwt token μας.

6. Η παρουσίαση της εφαρμογής

6.1. Αρχική οθόνη

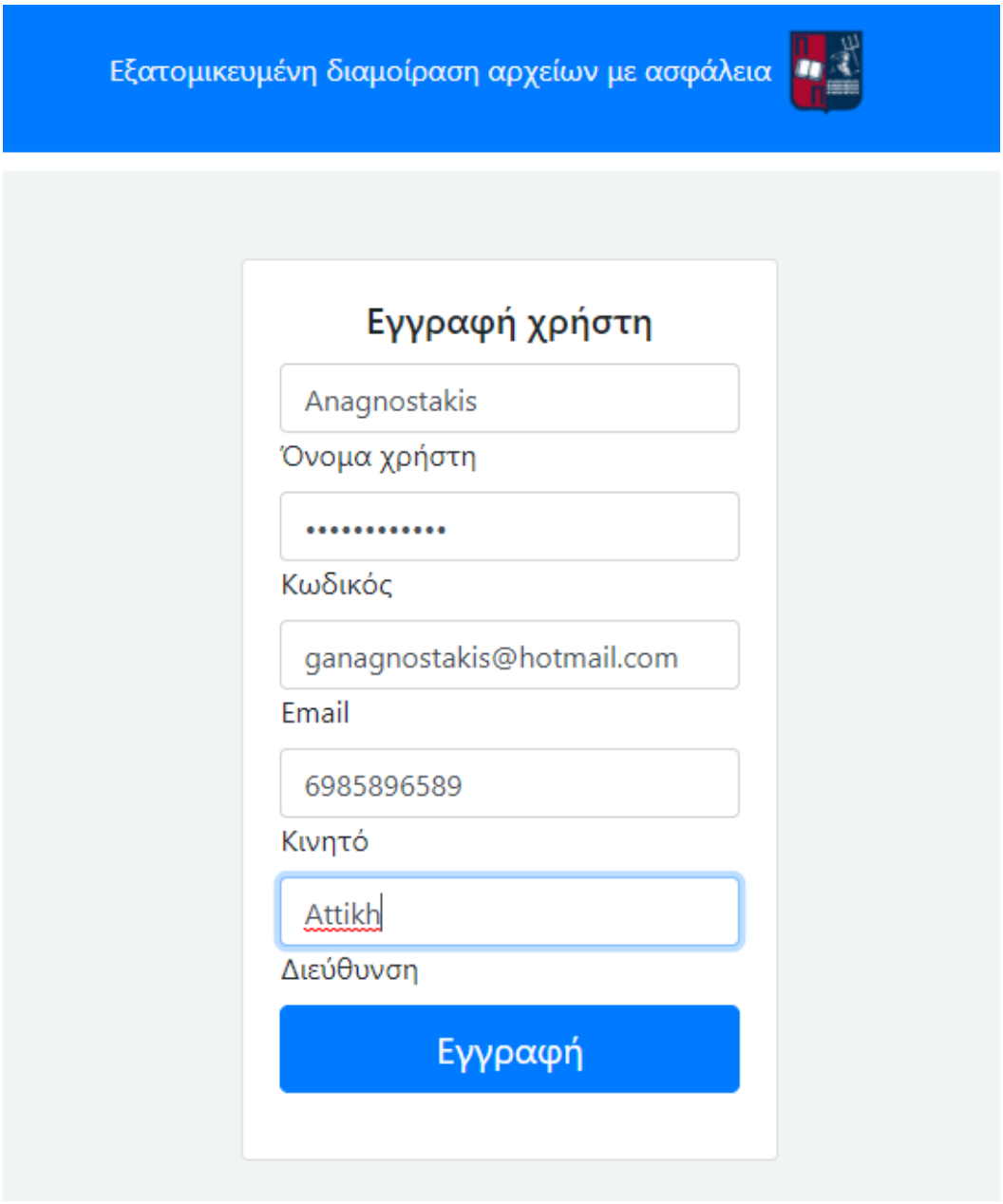
Πληκτρολογώντας την κατάλληλη διεύθυνση βρισκόμαστε στην αρχική οθόνη της εφαρμογής μας όπου μας δίνεται η δυνατότητα εισόδου ή εγγραφής στο σύστημα.



Εικόνα 22 Αρχική οθόνη

6.2. Εγγραφή

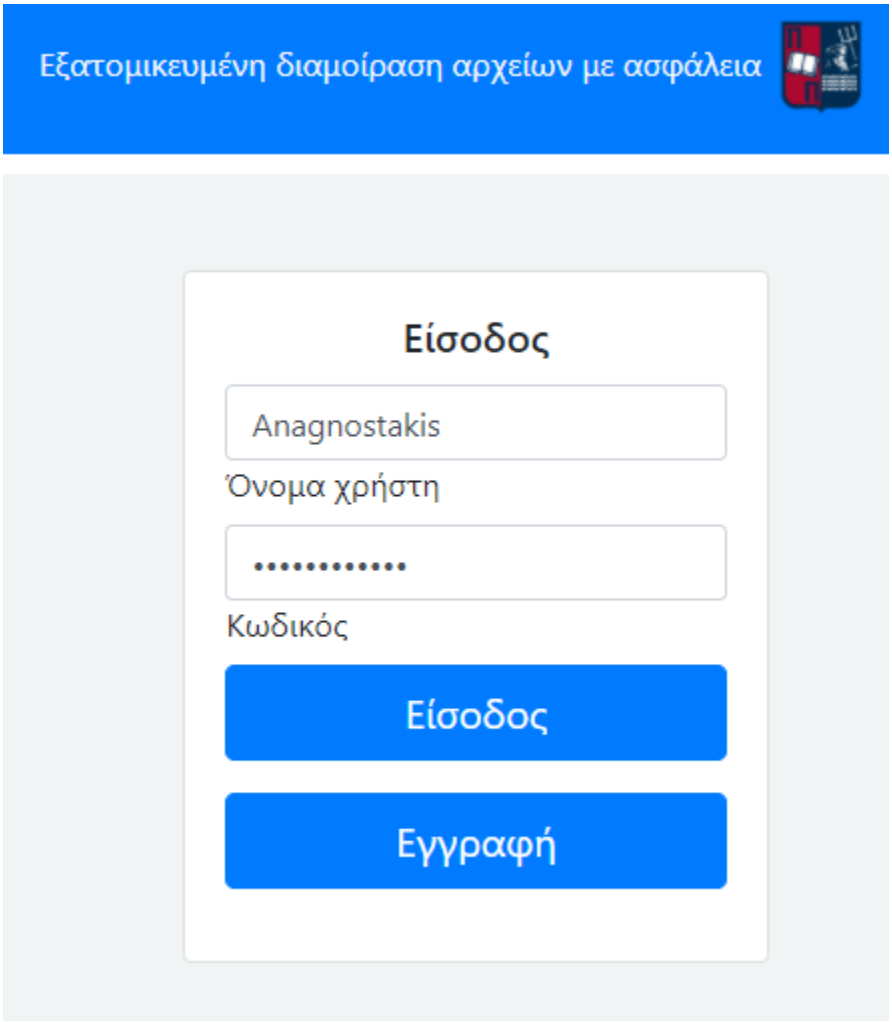
Το πρώτο βήμα για την χρήση των δυνατοτήτων της εφαρμογής μας είναι η εγγραφή στο σύστημα μας. Η εγγραφή επιτυγχάνεται μέσα από την είσοδο βασικών πληροφοριών του χρήστη μεταξύ αυτών όνομα χρήστη, κωδικό και διεύθυνση.



Εικόνα 23 Εγγραφή στο σύστημα

6.3. Είσοδος

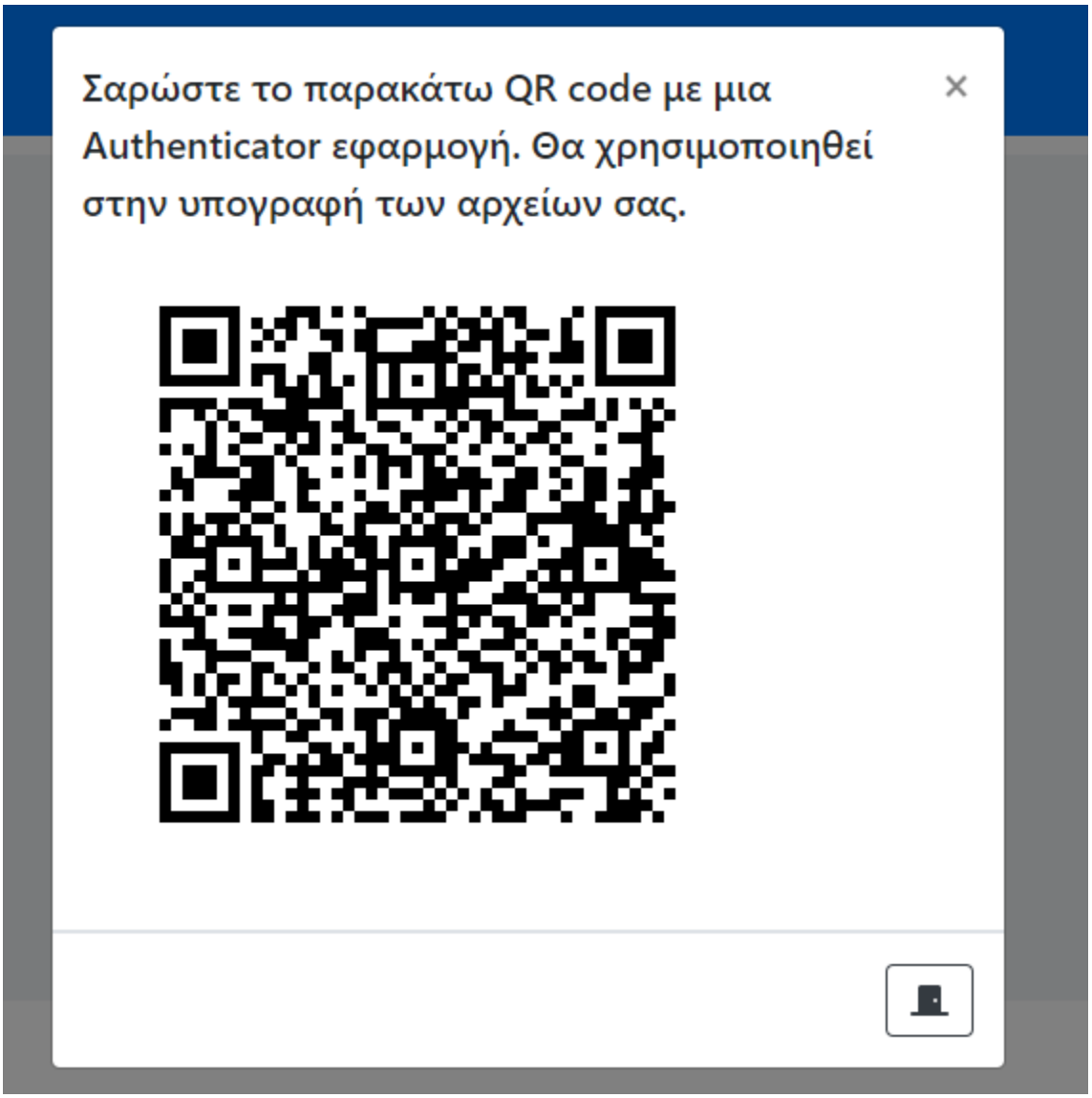
Εισάγοντας το όνομα χρήστη και τον κωδικό μας είμαστε έτοιμοι για την είσοδο στην εφαρμογή.



Εικόνα 24 Είσοδος στο σύστημα

6.4. QR Code

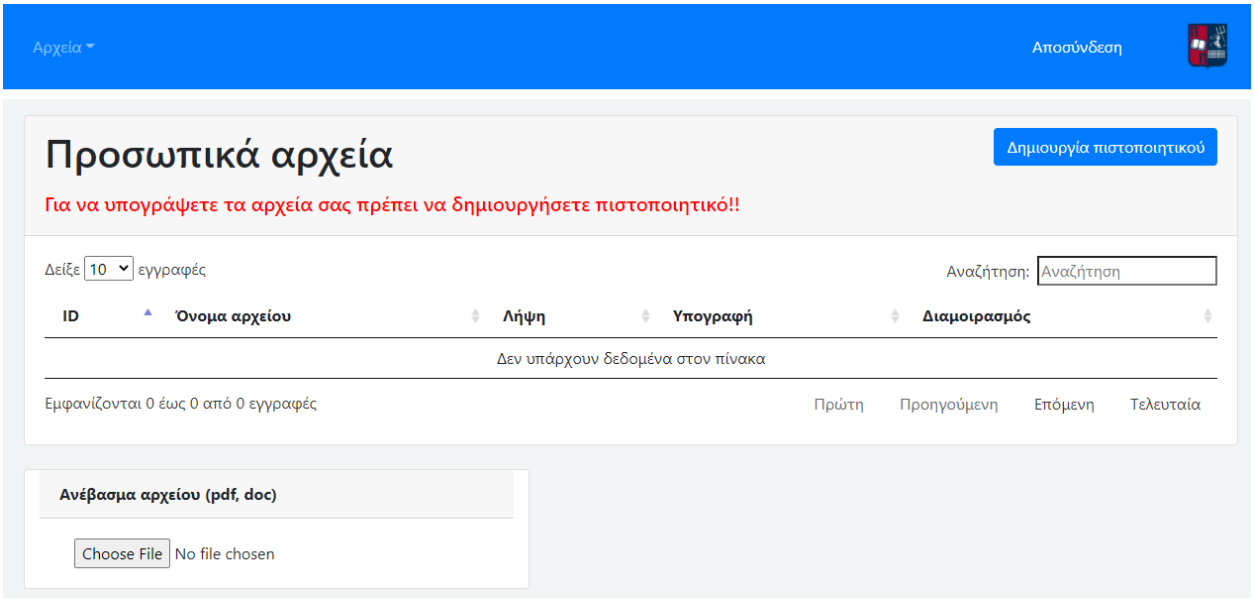
Σε συνέχεια, αν η είσοδος του χρήστη είναι πετυχημένη και αν είναι η πρώτη φορά που μπαίνει στην εφαρμογή μας θα του εμφανιστεί το παρακάτω παράθυρο με το QR Code το οποίο και θα πρέπει να σκανάρει με τον Authenticator.



Εικόνα 25 QR code μετά την επιτυχημένη είσοδο στο σύστημα

6.5. Σελίδα προσωπικών αρχείων

Αφού κλείσουμε το παράθυρο του QR code, θα οδηγηθούμε στη σελίδα των προσωπικών μας αρχείων

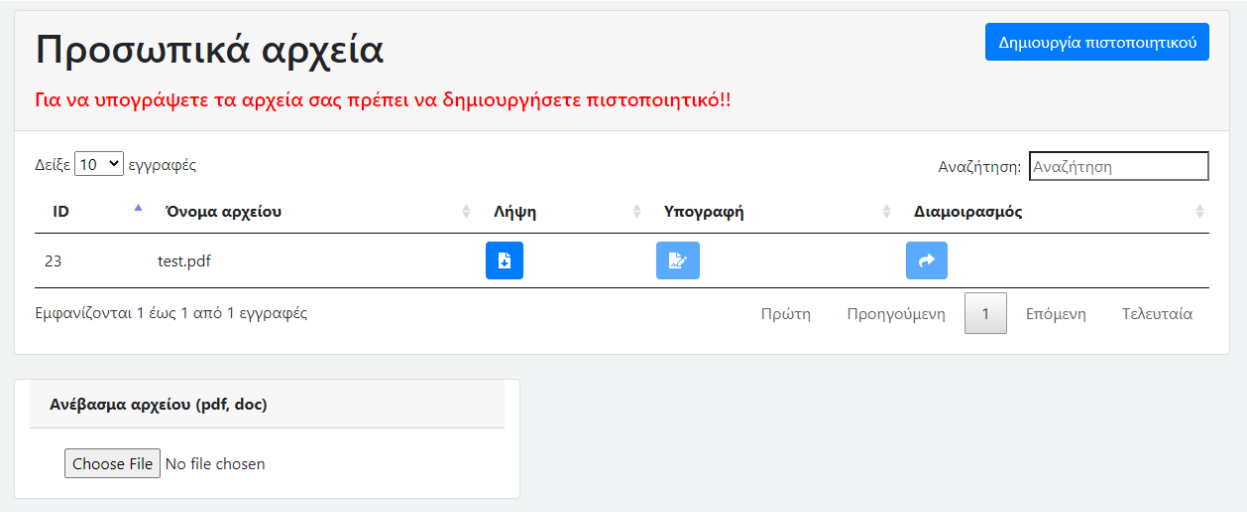


Εικόνα 26 Προσωπικά αρχεία

Σε αυτή τη σελίδα ο χρήστης μπορεί να δει τα προσωπικά του αρχεία, να ανεβάσει νέα να τα υπογράψει και να τα αποστείλει. Παράλληλα όπως δείχνει και η ενημέρωση με τα κόκκινα γράμματα, ο χρήστης πρέπει να ξεκινήσει τη διαδικασία δημιουργίας πιστοποιητικού πατώντας το κουμπί πάνω δεξιά.

6.6. Ανέβασμα αρχείου

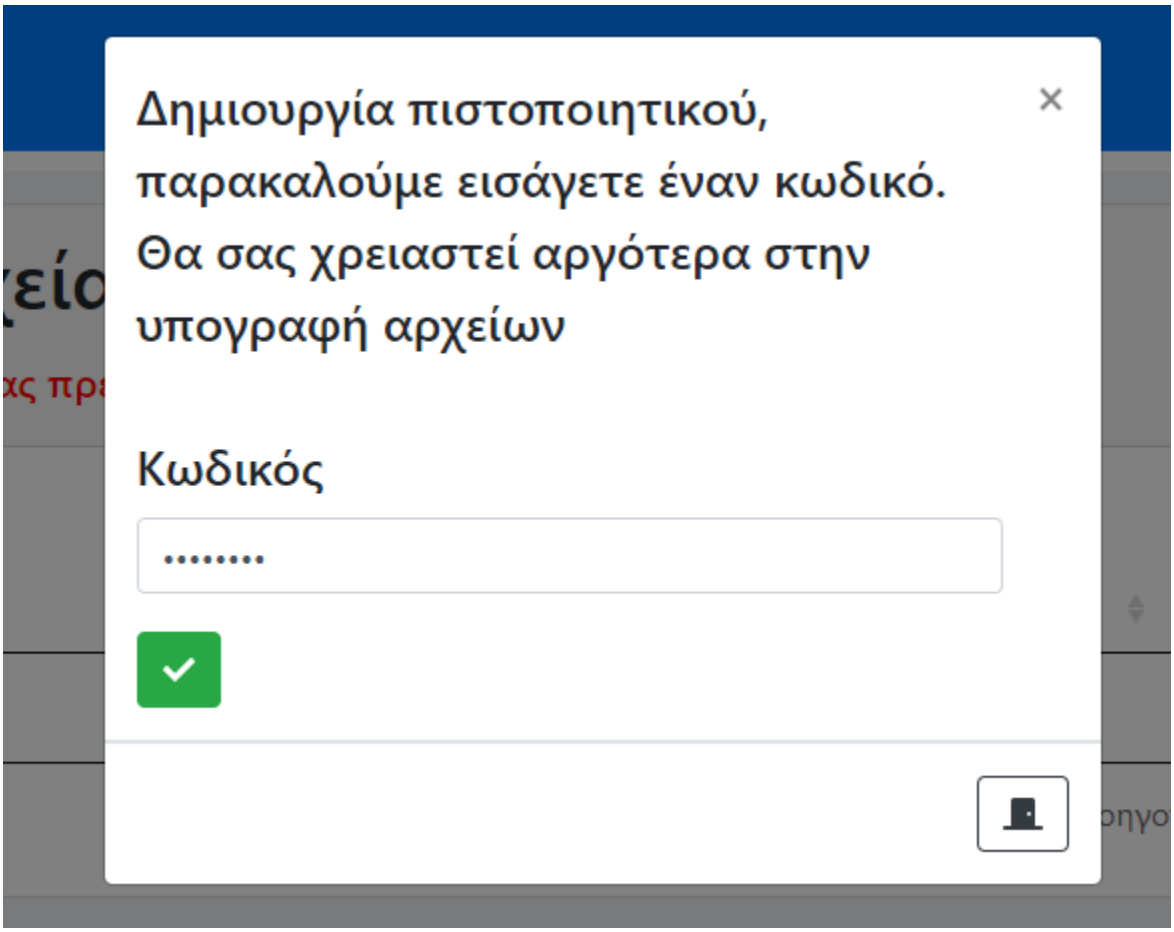
Παρακάτω βλέπουμε τη διαδικασία ανεβάσματος αρχείου. Πατάμε την επιλογή αρχείου και διαλέγουμε ένα αρχείο της μορφής pdf ή docx. Στη συνέχεια ο πίνακας μας ανανεώνεται δείχνοντας το αρχείο μας.



Εικόνα 27 Προβολή ανεβασμένων αρχείων

6.7. Δημιουργία ψηφιακής υπογραφής/πιστοποιητικού

Παρατηρούμε ότι η επιλογή «υπογραφή αρχείου» είναι απενεργοποιημένη. Αυτό συμβαίνει διότι δεν έχει δημιουργηθεί το πιστοποιητικό/ψηφιακή υπογραφή. Πατάμε δημιουργία και βλέπουμε την παρακάτω οθόνη.

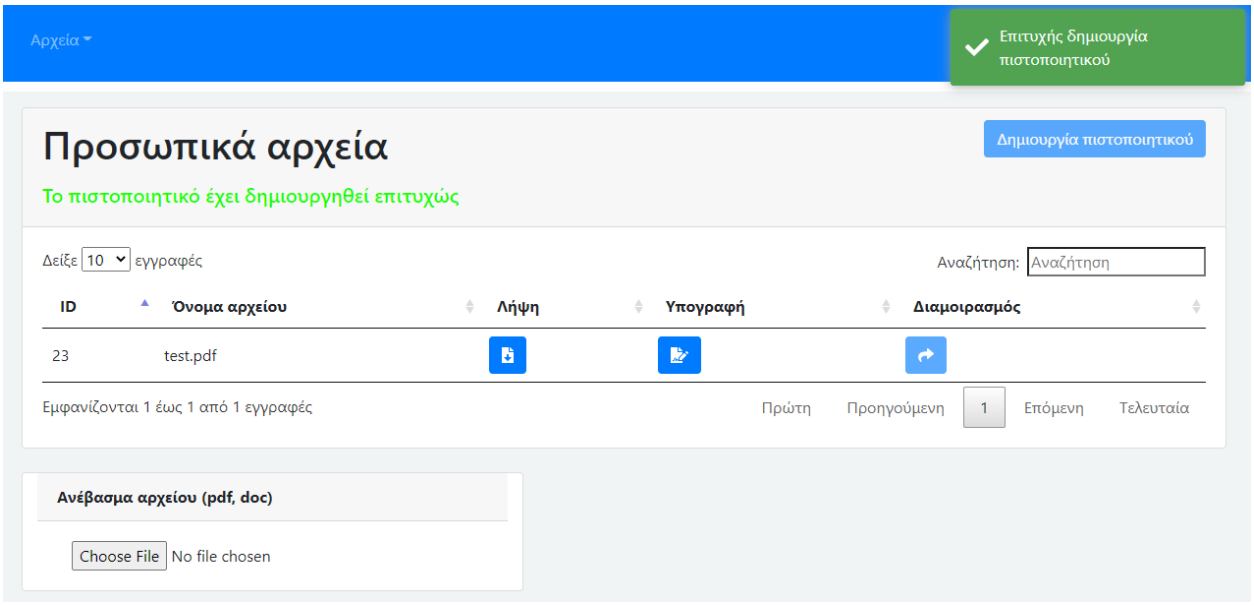


Εικόνα 28 Δημιουργία ψηφιακής υπογραφής/πιστοποιητικού

Εισάγουμε δύο φορές για επιβεβαίωση έναν κωδικό για το πιστοποιητικό μας και πατάμε το πράσινο κουμπί.

6.8. Επιτυχής δημιουργία υπογραφής/πιστοποιητικού

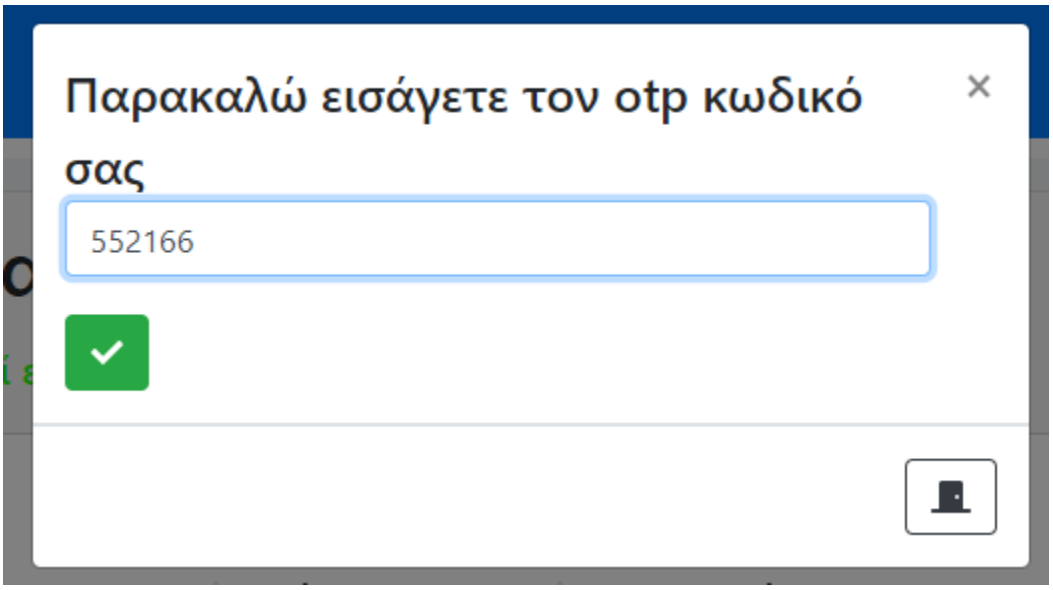
Ύστερα από την επιτυχή δημιουργία του πιστοποιητικού, κλείνει το παράθυρο που βλέπαμε και οδηγούμαστε πίσω στη σελίδα που ήμασταν. Παρατηρούμε πως πλέον επισημαίνεται με πράσινα γράμματα πως «Το πιστοποιητικό έχει δημιουργηθεί επιτυχώς» και ενεργοποιείται η υπογραφή του αρχείου μας.



Εικόνα 29 Επιτυχής δημιουργία υπογραφής/πιστοποιητικού

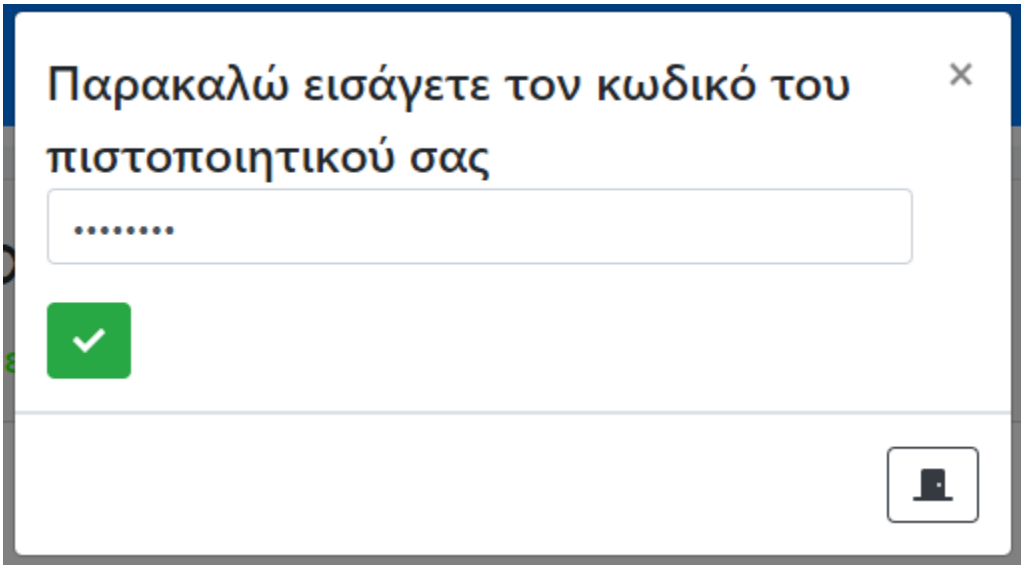
6.9. Υπογραφή αρχείου

Ας δούμε τώρα πως γίνεται η υπογραφή ενός αρχείου. Αρχικά πατάμε το κουμπί «υπογραφή» του αρχείου που επιθυμούμε να υπογράψουμε. Μας εμφανίζεται το παρακάτω παράθυρο που ζητάει τον otp κωδικό.

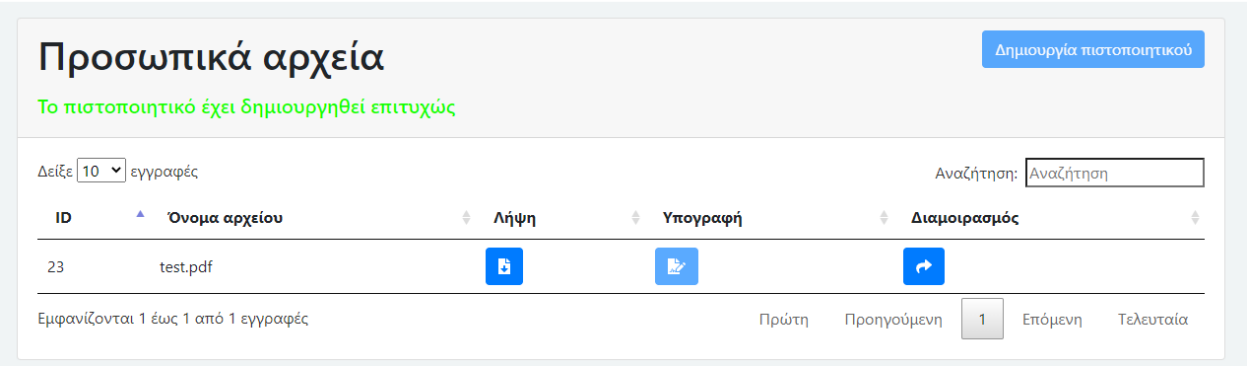


Εικόνα 30 Εισαγωγή otp κωδικού

Στη συνέχεια ζητείται ο κωδικός της ψηφιακής υπογραφής.

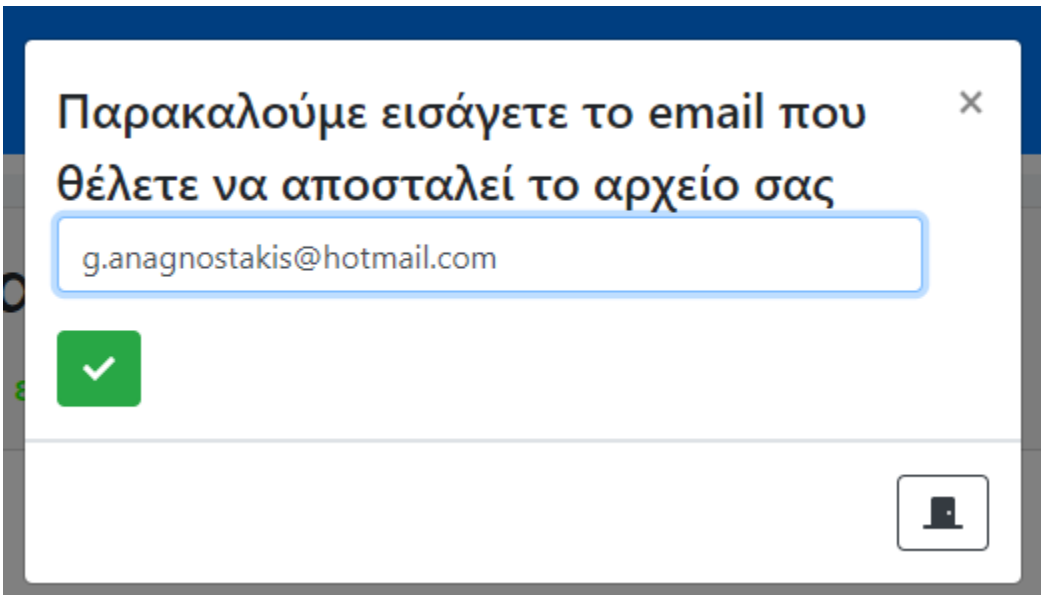


Μετά την επιτυχή εισαγωγή των παραπάνω κωδικών βλέπουμε πως ενεργοποιείται ο διαμοιρασμός του νέου υπογεγραμμένου αρχείου μας



6.10. Αποστολή αρχείου μέσω mail

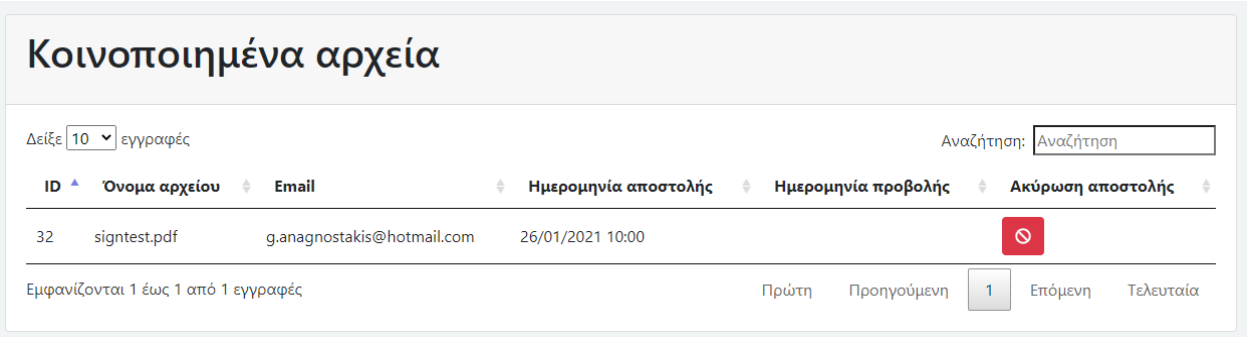
Πατάμε στο κουμπί «διαμοιρασμός και μας ανοίγει το παράθυρο εισαγωγής email όπως βλέπουμε παρακάτω.



Πληκτρολογούμε το email και πατάμε αποστολή. Το αρχείο έχει αποσταλεί επιτυχώς.

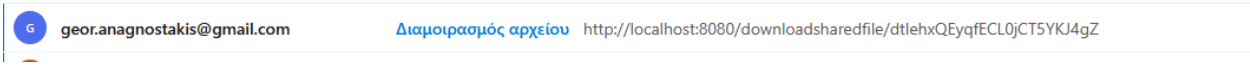
6.11. Προβολή κοινοποιημένων αρχείων

Στη συνέχεια επιλέγουμε από το μενού Αρχεία -> Κοινοποιημένα. Μας ανοίγει η σελίδα όπου προβάλλονται όλες οι αποστολές των αρχείων που έχουν γίνει, πότε προβλήθηκαν καθώς και η δυνατότητα ακύρωσης σε περίπτωση που δεν έχει ανοιχτεί ακόμα.



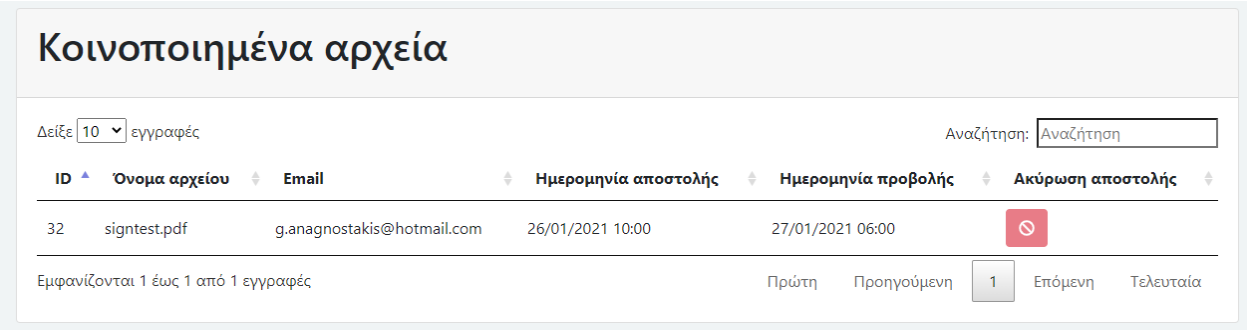
Εικόνα 31 Κοινοποιημένα αρχεία

Στα email του χρήστη βλέπουμε το mail που στάλθηκε



Εικόνα 32 Εισερχόμενο mail

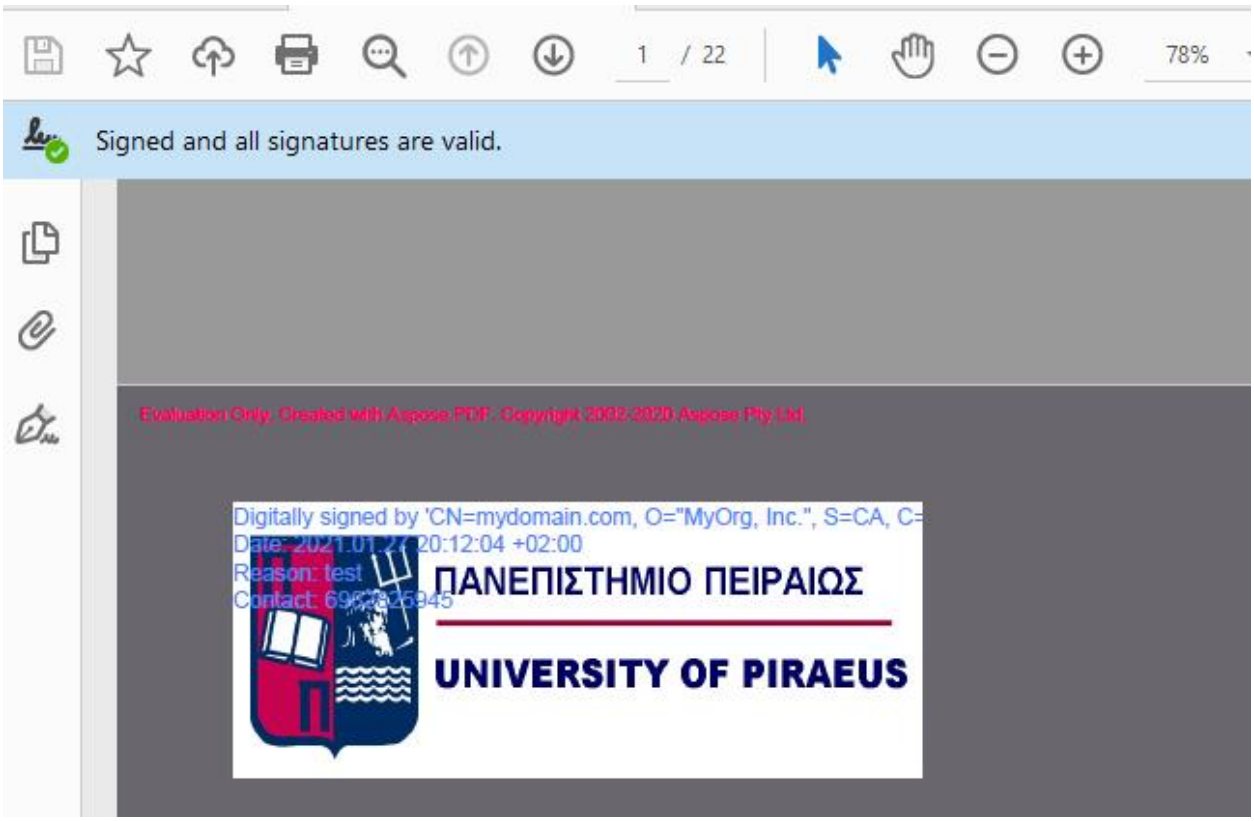
Όπως ήδη αναφέραμε, όταν ανοιχτεί το αρχείο προβάλλεται στον πίνακα και η ημερομηνία προβολής



Εικόνα 33 Ημερομηνία προβολής κοινοποιημένου αρχείου

6.12. Προβολή αρχείου στον adobe reader

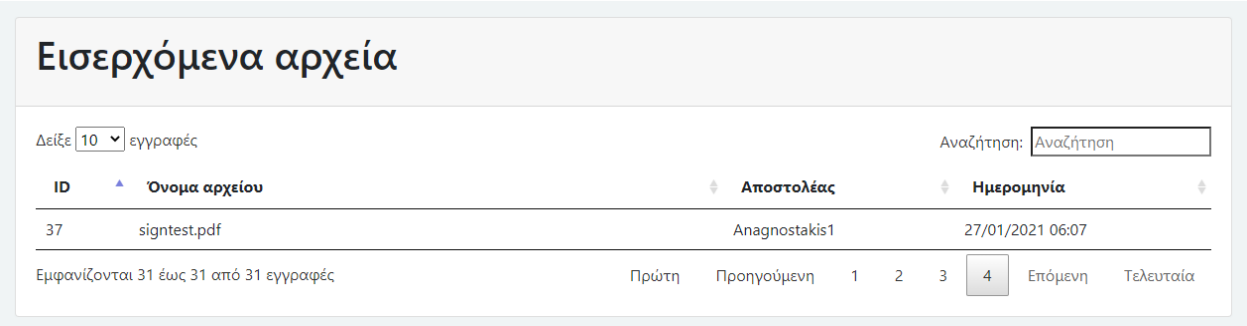
Επιλέγοντας να δούμε το αρχείο παρατηρούμε ότι αναγνωρίζεται ως επιτυχώς υπογεγραμμένο. Προϋπόθεση για αυτό είναι στον adobe reader να έχει εισαχθεί με επιτυχία το πιστοποιητικό του CA που έχει υπογράψει το δικό μας.



Εικόνα 34 Επιτυχώς υπογεγραμμένο αρχείο

6.13. Προβολή εισερχομένων αρχείων

Τέλος, αν ο χρήστης εγγραφεί με το email που του στάλθηκε το αρχείο, επιλέγοντας Αρχεία -> Εισερχόμενα θα δει τα αρχεία που του έχουν κοινοποιηθεί με αποστολέα και ημερομηνία.



Εικόνα 35 Εισερχόμενα αρχεία

7. Συμπεράσματα και μελλοντικές επεκτάσεις

7.1. Συμπεράσματα

Η συνεχής και αδιάκοπη εξέλιξη της τεχνολογίας μπορεί να μας λύνει τα χέρια αλλά αξιοσημείωτα είναι και τα προβλήματα ασφάλειας που δημιουργούνται. Η παρούσα διπλωματική διατριβή ασχολήθηκε με ένα από αυτά, την ασφάλεια στον διαμοιρασμό των αρχείων ηλεκτρονικά. Η τελική εφαρμογή που αναπτύχθηκε καλύπτει τους παρακάτω θεματικούς άξονες:

1. **Ασφάλεια**

   Η ψηφιακή υπογραφή, η χρήση one time password καθώς και ιδιωτικών κλειδιών προσέφεραν αρκετά επίπεδα ασφαλείας σε ότι αφορά την υπογραφή και αποστολή αρχείων. Παράλληλα η χρήση του Spring Security σε συνδυασμό με το JWT token προσέφεραν τεράστιες δυνατότητες διαχείρισης, όχι μόνο στο API μας αλλά και στο interface.

2. **Χαμηλές απαιτήσεις hardware και ευέλικτη εγκατάστασης**

   Η επιλογή της Java ως γλώσσα προγραμματισμού έγινε σκόπιμα ώστε να μπορεί να εγκατασταθεί και να λειτουργήσει εύκολα σε οποιαδήποτε συσκευή και λειτουργικό σύστημα απαιτηθεί. Τέλος η επιλογή της Angular ως γλώσσα υλοποίησης του frontend, κάνει το interface φιλικό και διαθέσιμο τόσο για υπολογιστές όσο και για tablet ή κινητές συσκευές.

7.2. Μελλοντικές επεκτάσεις

Καθημερινά οι ανάγκες για ασφάλεια μεταφοράς αρχείων μεγαλώνουν και εξελίσσονται. Η εφαρμογή δύναται να αναπτυχθεί περαιτέρω ώστε να καταστεί ευκολότερη στη χρήση και να εξυπηρετεί κάθε νέα ανάγκη του χρήστη. Παρακάτω αναφέρονται κάποιες βελτιώσεις οι οποίες κρίνονται απαραίτητες για τη εξέλιξη της εφαρμογής:

1. Βελτίωση του user interface για χρήση από κινητή συσκευή
2. Ανέβασμα και χρήση υπάρχουσας υπογραφής από το χρήστη
3. Μαζική αποστολή αρχείων μέσω mail
4. Δημιουργία λίστα επαφών από τα email που έχει αποσταλεί αρχείο στο παρελθόν
5. Ενημέρωση χρήστη αν το mail στου παραλήπτη αντιστοιχεί σε κάποιον εγγεγραμμένο στο σύστημα μας λογαριασμό.

7.3. Γνώσεις που αποκτήθηκαν

Αρχικά, η ανάπτυξη της παρούσας εφαρμογής συνέβαλε στην απόκτηση δεξιοτήτων και γνώσεων στο πεδίο της ανάπτυξης εφαρμογών με σύγχρονες μεθόδους. Συγκεκριμένα αποκτήθηκε η γνώση χρήσης κάποιων από των πιο διαδεδομένων framework για ανάπτυξη εφαρμογών, το Spring boot και η Angular.

Παράλληλα το αντικείμενο της διπλωματικής αυτής μας ώθησε στη μελέτη της ασφάλειας αρχείων και επικοινωνίας, κάτι που με την εξέλιξη της τεχνολογίας γίνεται όλο και πιο κρίσιμο. Μελετήθηκαν οι λειτουργίες των ιδιωτικών και δημοσίων κλειδιών, η ψηφιακή εφαρμογή και αποκτήθηκαν γνώσεις γύρω από την χρήση του ssl για την παραγωγή τους.

Τέλος μελετήθηκαν και χρησιμοποιήθηκαν τεχνικές που συνέβαλλαν σε κάποια επιπλέον επίπεδα ασφάλειας της εφαρμογής μας. Συγκεκριμένα η εκμάθηση των JWT και OTP βοήθησαν στην θωράκιση της εφαρμογής μας από τρίτους και κακόβουλες επιθέσεις.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Βασίλης, Γ., 2006. Αντικειμενοστραφής ανάπτυξη λογισμικού με τη UML. Αθήνα: Κλειδάριθμος.
2. Γιακουμάκης Μανώλης, Δ. Ν., 2009. Τεχνολογία Λογισμικού. s.l.:UNIBOOKS.
3. Κάτσικας,Κ., Γκρίτζαλης.Α. και Γκριτζάλης Σ. (2004). *Ασφάλεια πληροφοριακών συστήματων.* Αθήνα: Εκδόσεις νέων τεχνολογιών.
4. Wikipedia. 2021. JSON Web Token - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/JSON_Web_Token. [Accessed 24 January 2021].
5. Wikipedia. 2021. Public-key cryptography - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Public-key_cryptography. [Accessed 24 January 2021].
6. Wikipedia. 2021. OpenSSL - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/OpenSSL. [Accessed 24 January 2021].
7. Μάθημα: Θέματα Ασφάλειας στο Διαδίκτυο και στον Η/Υ. 2021. Μάθημα: Θέματα Ασφάλειας στο Διαδίκτυο και στον Η/Υ. [ONLINE] Available at: https://demetra.afs.edu.gr/moodle3/course/view.php?id=21. [Accessed 24 January 2021].
8. Medium. 2021. Build A One Time Password API. [ONLINE] Available at: https://medium.com/@__diffa/create-one-time-password-api-2285237f8656 [Accessed 25 January 2021].
9. Medium. 2021. How JSON Web Token(JWT) Authentication Works?. [ONLINE] Available at: https://medium.com/@sureshdsk/how-json-web-token-jwt-authentication-works-585c4f076033 [Accessed 25 January 2021].
10. Pacific Jour. 2021. Τι είναι η HTML. [ONLINE] Available at: http://pacific.jour.auth.gr/html/. [Accessed 25 January 2021].
11. E-learning-education. 2021. Τι είναι CSS. [ONLINE] Available at: https://www.e-learning-education.gr/%CF%84%CE%B9-%CE%B5%CE%AF%CE%BD%CE%B1%CE%B9-css/. [Accessed 25 January 2021].