Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής
Πρόγραμμα Μεταπτυχιακών Σπουδών
«Προηγμένα Συστήματα Πληροφορικής - Ανάπτυξη Λογισμικού και Τεχνητής Νοημοσύνης»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Λογισμικό εξατομικευμένων εντύπων συμπλήρωσης με ασφάλεια** **Secure personalized dynamic form creation software**
Ονοματεπώνυμο Φοιτητή	**Γκότσης-Λογοθέτης Ιωάννης**
Πατρώνυμο	**Παναγιώτης**
Αριθμός Μητρώου	**ΜΠΣΠ/ 18007**
Επιβλέπων	**Ευάγγελος Σακκόπουλος, Επίκουρος καθηγητής**

Ημερομηνία Παράδοσης **Ιούλιος 2021**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Ε. Αλέπης Αναπληρωτής Καθηγητής	Δ. Σωτηρόπουλος Επίκουρος Καθηγητής	Ε. Σακκόπουλος Επίκουρος Καθηγητής

Περιεχόμενα

Παράρτημα Εικόνων

Περίληψη

Στη συγκεκριμένη εργασία παρουσιάζεται μια web based πλατφόρμα ανάπτυξης φορμών εισαγωγής δεδομένων, με δυναμικό χαρακτήρα, με την βοήθεια ενός online text-editor παρέχοντας τη δυνατότητα σε κάθε είδους οργανισμό την ανάπτυξη φορμών εισαγωγής δεδομένων χωρίς καμία απαίτηση σε πόρους (storage,bandwidth,security κλπ.). Χρησιμοποιήθηκαν τεχνολογίες Microsoft ανοικτού κώδικα και κλασσικές τεχνολογίες frontend (html, javascript) καθώς και πολύ γνωστές βάσεις δεδομένων (SQL Server και Oracle). Ο χρήστης της εφαρμογής δημιουργεί φόρμες και τις διαχειρίζεται/επεξεργάζεται καθώς και βλέπει τις συμπληρωμένες φόρμες των πελατών του. Οι πελάτες του χρήστη, εισέρχονται στο περιβάλλον της εφαρμογής μέσω ενός link όπου και τους προβάλλεται η προς συμπλήρωση φόρμα. Η εφαρμογή στοχεύει στην επίλυση του προβλήματος της συχνής ανάγκης για δημιουργία έντυπων προς συμπλήρωση και γρήγορης διανομής αυτών στους τελικούς πελάτες τους. Το Σύστημα εξασφαλίζει την ασφάλεια τήρησης των δεδομένων, την διαθεσιμότητα (για παράδειγμα μπορεί να τρέχει σε έναν server στο cloud) και την ευκολία στη χρήση.

Abstract

This specific postgraduate dissertation presents a web-based platform for the development of data entry forms, with a dynamic character and with the help of an online text-editor, enabling any type of organization to develop data entry forms without any resource requirements (storage, bandwidth, security, etc.). For the development of the proposed platform, Microsoft open source technologies and classic frontend technologies (html, javascript) as well as well-known databases (SQL Server and Oracle) were used. The user of the application creates forms and manages / processes them and is able to view the submitted forms of his clients. The user's customers enter the application environment through a url where the document to be filled is displayed. The application aims to solve the problem of the frequent need to create fill-in forms and distributing them quickly to their end customers. The System ensures data security, availability (for example it can run on a cloud server) and ease of use.

Εισαγωγή – Σύντομη Περιγραφή

Οι ανάγκες των εταιρειών (Τράπεζες, εταιρείες Courier κλπ) για γρήγορη και δυναμική δημιουργία έντυπων συμπλήρωσης από τους πελάτες τους καθώς και η άμεση διανομή τους, για παράδειγμα ηλεκτρονικά σε μορφή ιστοσελίδας και όχι σε κάποιο είδους αρχείο, οδήγησαν στην ανάπτυξη της συγκεκριμένης διατριβής.

Η συγκεκριμένη διατριβή λοιπόν, αφορά την ανάπτυξη μιας web-based εφαρμογής δημιουργίας φορμών συμπλήρωσης και χρήσης τους από τους πελάτες (τελικοί χρήστες) των χρηστών της παρούσας εφαρμογής. Οι εν λόγω φόρμες μπορεί να είναι έντυπα νομικής συμμόρφωσης, έρευνες εργαζομένων, φόρμες εγγραφής και γενικά κάθε είδους έγγραφο που χρειάζεται εισαγωγή δεδομένων χρήστη. Ο χρήστης της εφαρμογής δημιουργεί τις δικές του φόρμες εισάγοντας και επεξεργάζοντας κείμενο και controls σε έναν WYSIWYG editor. Ο τελικός πελάτης μπορεί να εισάγει κείμενο, να επιλέξει από κουμπί πολλαπλής επιλογής, επιλογή σε checkbox καθώς και να εισάγει την υπογραφή του ηλεκτρονικά.

Η παρούσα διατριβή παρουσιάζει τεχνικό ενδιαφέρον ως προς τον τρόπο υλοποίησης καθώς και τις τεχνολογίες που χρησιμοποιήθηκαν. Η υλοποίηση ενός κειμενογράφου για το διαδίκτυο αποδεικνύεται δύσκολη εργασία. Οι ανάγκες για την ασφαλή αποθήκευση και διαχείριση των (προσωπικών) δεδομένων είναι επίσης μεγάλες.

Η εργασία χωρίζεται σε πέντε κεφάλαια. Στο πρώτο κεφάλαιο παρουσιάζεται το θεωρητικό υπόβαθρο πίσω από τους αλγορίθμους που χρησιμοποιήθηκαν. Στο δεύτερο κεφάλαιο παρουσιάζονται παραπλήσια software με το powForms ως προς την λειτουργία, αναλύονται και συγκρίνονται. Το τρίτο κεφάλαιο περιέχει τις τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη του συστήματος δίδοντας βάση στα Razor Pages. Το τέταρτο κεφάλαιο αναλύει την λειτουργία του συστήματος δείχνοντας εικόνες σε Ενοποιημένη Γλώσσα Σχεδίασης Προτύπων UML καθώς και δύο ολοκληρωμένα παραδείγματα χρήσης της εφαρμογής. Το πέμπτο και τελευταίο κεφάλαιο περιέχει τα τελικά συμπεράσματα και πιθανές μελλοντικές επεκτάσεις.

Κεφάλαιο 1. Θεωρητικό υπόβαθρο

1.1 REST

Representational state transfer (REST) είναι ένα λογισμικό που χρησιμοποιεί ένα υποσύνολο του HTTP. Συνήθως χρησιμοποιείται για τη δημιουργία διαδραστικών εφαρμογών που χρησιμοποιούν Web Services. Ένα WebService που ακολουθεί αυτές τις οδηγίες ονομάζεται RESTful. Ως τέτοιο, πρέπει να παρέχει τους πόρους Ιστού της σε μια αναπαράσταση κειμένου και να τους επιτρέπει να διαβάζονται και να τροποποιούνται με ένα πρωτόκολλο χωρίς κατάσταση και ένα προκαθορισμένο σύνολο λειτουργιών. Αυτή η προσέγγιση επιτρέπει τη διαλειτουργικότητα μεταξύ των συστημάτων υπολογιστών στο Διαδίκτυο που παρέχουν αυτές τις υπηρεσίες. Το REST είναι μια εναλλακτική λύση, για παράδειγμα του SOAP, ως τρόπου πρόσβασης σε ένα web service.

Οι "πόροι Ιστού" (Web resources) ορίστηκαν για πρώτη φορά στον Παγκόσμιο Ιστό ως έγγραφα ή αρχεία που προσδιορίζονται από τις διευθύνσεις URL τους. Σήμερα, ο ορισμός είναι πολύ πιο γενικός και αφηρημένος και περιλαμβάνει κάθε πράγμα, οντότητα ή ενέργεια που μπορεί να αναγνωριστεί, να ονομαστεί, να αντιμετωπιστεί ή να εκτελεστεί με οποιονδήποτε τρόπο στον Ιστό. Σε ένα RESTful Web service, αιτήματα που υποβάλλονται σε URI ενός πόρου προκαλούν απάντηση μορφοποιημένη σε HTML, XML, JSON ή κάποια άλλη μορφή. Για παράδειγμα, η απόκριση μπορεί να επιβεβαιώσει ότι η κατάσταση πόρων έχει αλλάξει. Η απάντηση μπορεί επίσης να περιλαμβάνει συνδέσμους υπερκειμένου με σχετικούς πόρους. Το πιο κοινό πρωτόκολλο για αυτά τα αιτήματα και τις απαντήσεις είναι το HTTP. Παρέχει λειτουργίες (μεθόδους HTTP) όπως GET, POST, PUT και DELETE. Χρησιμοποιώντας ένα πρωτόκολλο χωρίς κατάσταση (Stateless) και τυπικές λειτουργίες, τα συστήματα RESTful στοχεύουν στη γρήγορη απόδοση, την αξιοπιστία και την ικανότητα ανάπτυξης, επαναχρησιμοποιώντας στοιχεία που μπορούν να διαχειριστούν και να ενημερωθούν χωρίς να επηρεάσουν το σύστημα στο σύνολό του, ακόμη και όταν λειτουργεί.

Ο στόχος του REST είναι να αυξήσει την απόδοση, την επεκτασιμότητα, την απλότητα, την τροποποίηση, την ορατότητα, τη φορητότητα και την αξιοπιστία. Αυτό επιτυγχάνεται ακολουθώντας τις αρχές REST, όπως αρχιτεκτονική client-server, πρωτόκολλο stateless, δυνατότητα Cache, χρήση ενός συστήματος με στρώσεις, υποστήριξη για κώδικα on demand και χρήση ομοιόμορφης διεπαφής. Αυτές οι αρχές πρέπει να τηρούνται για να ταξινομηθεί το σύστημα ως REST.

Ο όρος representational state transfer εισήχθη και ορίστηκε το 2000 από τον Roy Fielding στη διδακτορική του διατριβή.Η διατριβή του Fielding εξήγησε τις αρχές REST που ήταν γνωστές ως "μοντέλο αντικειμένου HTTP" από το 1994 και χρησιμοποιήθηκαν για το σχεδιασμό των προτύπων HTTP 1.1 και Uniform Resource Identifiers (URI). Ο όρος προορίζεται να προκαλέσει μια εικόνα του τρόπου με τον οποίο συμπεριφέρεται μια καλά σχεδιασμένη εφαρμογή Ιστού: είναι ένα δίκτυο πόρων Ιστού όπου ο χρήστης προχωρά μέσω της εφαρμογής επιλέγοντας αναγνωριστικά πόρων όπως http://www.example.com/άρθρα/21 και λειτουργίες πόρων, όπως το GET ή το POST (προκαλούν μεταβολή της κατάστασης της εφαρμογής), με αποτέλεσμα η αναπαράσταση του επόμενου πόρου (η επόμενη κατάσταση εφαρμογής) να μεταφερθεί στον τελικό χρήστη για χρήση του. [1]

1.2 TOTP

Ο μοναδικός κωδικός πρόσβασης με βάση το χρόνο (Time-based One-time Password TOTP) είναι ένας αλγόριθμος υπολογιστών που δημιουργεί έναν κωδικό πρόσβασης μίας χρήσης (OTP) που χρησιμοποιεί την τρέχουσα ώρα ως πηγή μοναδικότητας. Μια επέκταση του αλγόριθμου One-Time Password που βασίζεται στο HMAC (HOTP), έχει υιοθετηθεί ως πρότυπο RFC 6238 του Internet Engineering Task Force (IETF). Το TOTP είναι ο ακρογωνιαίος λίθος του Initiative for Open Authentication (OATH) και χρησιμοποιείται σε διάφορα συστήματα ελέγχου ταυτότητας δύο παραγόντων (2FA). [2]

Ο αλγόριθμος HOTP καθορίζει έναν αλγόριθμο OTP βάσει συμβάντων, όπου ο κινούμενος παράγοντας είναι ένας μετρητής συμβάντων. Η παρούσα εργασία βασίζει την κίνηση συντελεστή σε μια τιμή χρόνου.

Μια παραλλαγή βάσει του χρόνου του αλγορίθμου OTP παρέχει βραχύβιες τιμές OTP, οι οποίες είναι επιθυμητές για βελτιωμένη ασφάλεια. Ο προτεινόμενος αλγόριθμος μπορεί να χρησιμοποιηθεί σε ένα ευρύ φάσμα εφαρμογών δικτύου, από απομακρυσμένη πρόσβαση εικονικού ιδιωτικού δικτύου (VPN) και σύνδεση δικτύου Wi-Fi σε εφαρμογές Web προσανατολισμένες στις συναλλαγές. Οι συγγραφείς πιστεύουν ότι ένας κοινός και διαμοιρασμένος αλγόριθμος θα διευκολύνει την υιοθέτηση ελέγχου ταυτότητας δύο παραγόντων (2-factor authentication) στο Διαδίκτυο, επιτρέποντας τη διαλειτουργικότητα σε εμπορικές και ανοικτού κώδικα εφαρμογές.

Ιστορικό

Όπως ορίζεται στο [RFC4226], ο αλγόριθμος HOTP βασίζεται στον αλγόριθμο HMAC-SHA-1 (όπως καθορίζεται στο [RFC2104]) και εφαρμόζεται σε μια αυξανόμενη τιμή μετρητή που αντιπροσωπεύει το μήνυμα στον υπολογισμό HMAC. Βασικά, η έξοδος του υπολογισμού HMAC-SHA-1 περικόπτεται για τη λήψη φιλικών προς τον χρήστη τιμών: HOTP (K, C) = Truncate (HMAC-SHA-1 (K, C)) όπου το Truncate αντιπροσωπεύει τη λειτουργία που μπορεί να μετατρέψει μια HMAC-SHA-1 τιμή σε τιμή HOTP. Τα K και C αντιπροσωπεύουν το κοινό μυστικό και την τιμή του μετρητή. Το TOTP είναι η παραλλαγή βάσει του χρόνου αυτού του αλγορίθμου, όπου μια τιμή T, που προέρχεται από μια αναφορά χρόνου και ένα βήμα χρόνου, αντικαθιστά τον μετρητή C στον υπολογισμό HOTP. Οι υλοποιήσεις TOTP ενδέχεται να χρησιμοποιούν συναρτήσεις HMAC-SHA-256 ή HMAC-SHA-512, με βάση τις λειτουργίες κατακερματισμού SHA-256 ή SHA-512 [SHA2], αντί για τη λειτουργία HMAC-SHA-1 που έχει καθοριστεί για τον υπολογισμό HOTP στο [RFC4226].

Απαιτήσεις αλγορίθμου

Αυτή η ενότητα συνοψίζει τις απαιτήσεις που λαμβάνονται υπόψη για το σχεδιασμό του αλγορίθμου TOTP.

R1: Ο prover (π.χ. token, soft token) και verifier (διακομιστής ελέγχου ταυτότητας ή επικύρωσης) πρέπει να γνωρίζει ή να μπορεί να αντλήσει τον τρέχοντα χρόνο Unix (δηλαδή, ο αριθμός των δευτερολέπτων που έχουν παρέλθει από τα μεσάνυχτα UTC της 1ης Ιανουαρίου 1970) για την παραγωγή OTP. Η ακρίβεια του χρόνου που χρησιμοποιείται από τον prover επηρεάζει το πόσο συχνά θα πρέπει να γίνεται συγχρονισμός ρολογιού.

R2: Ο prover και ο verifier πρέπει να μοιράζονται το ίδιο μυστικό ή τη γνώση ενός μυστικού μετασχηματισμού, για να δημιουργήσουν ένα κοινό μυστικό.

R3: Ο αλγόριθμος πρέπει να χρησιμοποιεί το HOTP [RFC4226] ως βασικό δομικό στοιχείο.

R4: Ο prover και ο verifier πρέπει να χρησιμοποιούν την ίδια τιμή χρονικού βήματος X.

R5: πρέπει να υπάρχει ένα μοναδικό μυστικό (κλειδί) για κάθε prover.

R6: Τα κλειδιά πρέπει να δημιουργούνται τυχαία χρησιμοποιώντας αλγόριθμους παραγωγής κλειδιών.

R7: Τα κλειδιά πρέπει να αποθηκεύονται σε μια συσκευή ανθεκτική σε παραβιάσεις και πρέπει να προστατεύονται από μη εξουσιοδοτημένη πρόσβαση και χρήση.

Αλγόριθμος TOTP

Αυτή η παραλλαγή του αλγορίθμου HOTP καθορίζει τον υπολογισμό ενός OTP με βάση την αναπαράσταση του μετρητή ως παράγοντα χρόνου.

Σημειώσεις

Το X αντιπροσωπεύει το βήμα χρόνου σε δευτερόλεπτα (προεπιλεγμένη τιμή X = 30 δευτερόλεπτα) και είναι μια παράμετρος συστήματος.

Το T0 είναι ο χρόνος Unix για να ξεκινήσει η μέτρηση των βημάτων χρόνου (η προεπιλεγμένη τιμή είναι 0, δηλαδή η εποχή του Unix) και είναι επίσης μια παράμετρος συστήματος.

Περιγραφή

Βασικά, ορίζουμε το TOTP ως TOTP = HOTP (K, T), όπου το T είναι ακέραιος και αντιπροσωπεύει τον αριθμό των βημάτων χρόνου μεταξύ του αρχικού χρόνου μετρητή T0 και του τρέχοντος χρόνου Unix. Πιο συγκεκριμένα, T = (Τρέχων χρόνος Unix - T0) / X, όπου η προεπιλεγμένη συνάρτηση κατωφλίου χρησιμοποιείται στον υπολογισμό.

Για παράδειγμα, με T0 = 0 και Time Step X = 30, T = 1 εάν ο τρέχων χρόνος Unix είναι 59 δευτερόλεπτα και T = 2 εάν ο τρέχων χρόνος Unix είναι 60 δευτερόλεπτα.

Η εφαρμογή αυτού του αλγορίθμου πρέπει να υποστηρίζει τιμή χρόνου T μεγαλύτερη από έναν ακέραιο 32-bit όταν είναι πέρα από το έτος 2038. Η τιμή των παραμέτρων του συστήματος X και T0 είναι προκαθορισμένη κατά τη διαδικασία παροχής και κοινοποιείται μεταξύ ενός prover και ενός verifier ως μέρος του βήματος παροχής.

Ζητήματα ασφάλειας

Γενικά

Η ασφάλεια και η ισχύς αυτού του αλγορίθμου εξαρτώνται από τις ιδιότητες του υποκείμενου δομικού στοιχείου HOTP, το οποίο είναι μια κατασκευή που βασίζεται στο HMAC [RFC2104] χρησιμοποιώντας το SHA-1 ως hash function.

Το συμπέρασμα της ανάλυσης ασφαλείας που περιγράφεται λεπτομερώς στο [RFC4226] είναι ότι, για όλους τους πρακτικούς σκοπούς, οι έξοδοι της δυναμικής περικοπής σε ξεχωριστές εισόδους είναι ομοιόμορφα και ανεξάρτητα κατανεμημένες συμβολοσειρές.

Η ανάλυση δείχνει ότι η καλύτερη δυνατή επίθεση κατά της λειτουργίας HOTP είναι η επίθεση brute force.

Όπως υποδεικνύεται στην ενότητα απαιτήσεων αλγορίθμου, τα κλειδιά πρέπει να επιλέγονται τυχαία ή να χρησιμοποιούν μια κρυπτογραφικά ισχυρή ψευδοτυχαία γεννήτρια με κατάλληλα δοσμένη τυχαία τιμή.

Τα κλειδιά πρέπει να είναι του μήκους της εξόδου HMAC για τη διευκόλυνση της διαλειτουργικότητας.

Όλες οι επικοινωνίες πρέπει να πραγματοποιούνται μέσω ενός ασφαλούς καναλιού, π.χ., Secure Socket Layer / Transport Layer Security (SSL / TLS) [RFC5246] ή IPsec [RFC4301].

Τα κλειδιά πρέπει να βρίσκονται σε ασφαλή περιοχή, για να αποφευχθεί, όσο το δυνατόν περισσότερο, άμεση επίθεση στο σύστημα επικύρωσης και στη βάση δεδομένων μυστικών. Συγκεκριμένα, η πρόσβαση στο βασικό υλικό πρέπει να περιορίζεται σε προγράμματα και διαδικασίες που απαιτούνται μόνο από το σύστημα επικύρωσης. [3], [4]

1.3 SHA-512

Το SHA-512 είναι ένας αλγόριθμος κατακερματισμού που εκτελεί μια λειτουργία κατακερματισμού σε ορισμένα δεδομένα που του έχουν δοθεί.

Οι αλγόριθμοι κατακερματισμού χρησιμοποιούνται σε πολλά πράγματα όπως η ασφάλεια στο Διαδίκτυο, τα ψηφιακά πιστοποιητικά και ακόμη και στο blockchain. [5]

Απαιτήσεις συναρτήσεων κατακερματισμού

Ο σκοπός μιας συνάρτησης κατακερματισμού είναι η παραγωγή ενός «αποτυπώματος» (fingerprint) για ένα αρχείο, ένα μήνυμα, ή κάποιο άλλο τμήμα δεδομένων. Για να είναι χρήσιμη για πιστοποίηση αυθεντικότητας, μια συνάρτηση κατακερματισμού H θα πρέπει να έχει τις ακόλουθες ιδιότητες:

1. Να μπορεί να εφαρμόζεται σε τμήματα δεδομένων οποιουδήποτε μεγέθους.
2. Να παράγει έξοδο συγκεκριμένου σταθερού μεγέθους.
3. Να είναι σχετικά εύκολη στον υπολογισμό για οποιοδήποτε δεδομένο χ, καθιστώντας πρακτικά εφαρμόσιμες τις υλοποιήσεις σε υλικό και λογισμικό.
4. Για κάθε δεδομένη τιμή h,να είναι υπολογιστικά αδύνατο να βρεθεί το χ για το οποίο H(χ) = h. Αυτή η ιδιότητα αναφέρεται συχνά στη βιβλιογραφία ως μονόδρομη (one-way) ιδιότητα.
5. Για κάθε δεδομένο τμήμα x, να είναι υπολογιστικά αδύνατο να βρεθεί ένα y≠x με H(x)=H(y).Αυτή αναφέρεται συχνά ως ιδιότητα ασθενούς ανθεκτικότητας σε διένεξη (weak collision resistance).
6. Να είναι υπολογιστικά αδύνατο να βρεθεί ένα ζευγάρι (x,y) τέτοιο ώστε H(x)=H(y).Αυτή αναφέρεται συχνά ως ιδιότητα ισχυρής ανθεκτικότητας σε διένεξη (strong collision resistance).

Οι τρεις πρώτες ιδιότητες αποτελούν απαιτήσεις για την πρακτική εφαρμογή μιας συνάρτησης κατακερματισμού για πιστοποίηση αυθεντικότητας μηνυμάτων.Η τέταρτη είναι η «μονόδρομη»

ιδιότητα. Είναι εύκολο να δημιουργήσει κανείς ένα κωδικό με δεδομένο το μήνυμα, αλλά είναι στην πραγματικότητα αδύνατο να δημιουργήσει ένα μήνυμα με δεδομένο τον κωδικό. Αυτή η ιδιότητα είναι σημαντική αν η τεχνική πιστοποίησης αυθεντικότητας περιλαμβάνει την χρήση μιας μυστικής τιμής.

Η Πέμπτη ιδιότητα εξασφαλίζει ότι είναι αδύνατο να βρεθεί ένα εναλλακτικό μήνυμα με ίδια τιμή κατακερματισμού (hash value) με το δεδομένο μήνυμα. Αυτό αποτρέπει την πλαστογραφία όταν χρησιμοποιείται κρυπτογραφημένος κώδικας κατακερματισμού.

Μια συνάρτηση κατακερματισμού που ικανοποιεί τις πέντε πρώτες ιδιότητες της προηγούμενης λίστας αναφέρεται ως ασθενής συνάρτηση κατακερματισμού. Αν ικανοποιείται επίσης και η έκτη ιδιότητα, τότε αναφέρεται ως ισχυρή συνάρτηση κατακερματισμού.

Εκτός από την ήδη παρεχόμενη πιστοποίηση αυθεντικότητας, η σύνοψη μηνύματος εξασφαλίζει επίσης την ακεραιότητα δεδομένων. Παρέχει την ίδια λειτουργία όπως μια ακολουθία ελέγχου πλαισίου (frame check sequence):Αν κάποια bit του μηνύματος αλλάξουν καταλάθος στη διάρκεια της μετάδοσης, η σύνοψη του μηνύματος θα είναι λανθασμένη.

Ο Ασφαλής Αλγόριθμος Κατακερματισμού (Secure Hash Algorithm, SHA) αναπτύχθηκε από το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογιών των Η.Π.Α. (NIST) και δημοσιεύτηκε ως Ομοσπονδιακό πρότυπο επεξεργασίας πληροφοριών (FIPS PUB 180) το 1993. Μια αναθεωρημένη έκδοση εκδόθηκε ως FIPS PUB 180-1 το 1995, και αναφέρεται γενικά ως SHA-1. Ο αλγόριθμος SHA-1 καθορίζεται επίσης στο RFC 3174,το οποίο είναι ουσιαστικά ένα αντίγραφο του FIPS PUB 180-1 με την προσθήκη της υλοποίησης σε γλώσσα C.

Ο αλγόριθμος παράγει μια τιμή κατακερματισμού με μήκος 160 bit.Το 2002 το NIST εξέδωσε μια νέα έκδοση του προτύπου, το FIPS PUB 180-2,οπου καθόριζε τρεις νέες εκδόσεις του αλγορίθμου SHA με μήκος τιμής κατακερματισμού 256, 384 και 512 bit.Οι εκδόσεις αυτές είναι γνωστές ως SHA-256, SHA-384, και SHA-512 αντίστοιχα. Οι εκδόσεις αυτές έχουν την ίδια βασική δομή και χρησιμοποιούν τις ίδιες αριθμητικές και λογικές πράξεις με τον αλγόριθμο SHA-1.

Ο αλγόριθμος SHA-512 παίρνει ως είσοδο ένα μήνυμα με μέγιστο μήκος 2^{128} bit και παράγει μια σύνοψη με μήκος 512 bit. Η επεξεργασία της εισόδου γίνεται σε μπλόκ των 1024 bit. [6]

Κεφάλαιο 2. Προηγούμενες και Σχετικές Μέθοδοι και Λύσεις

2.1 Εισαγωγή

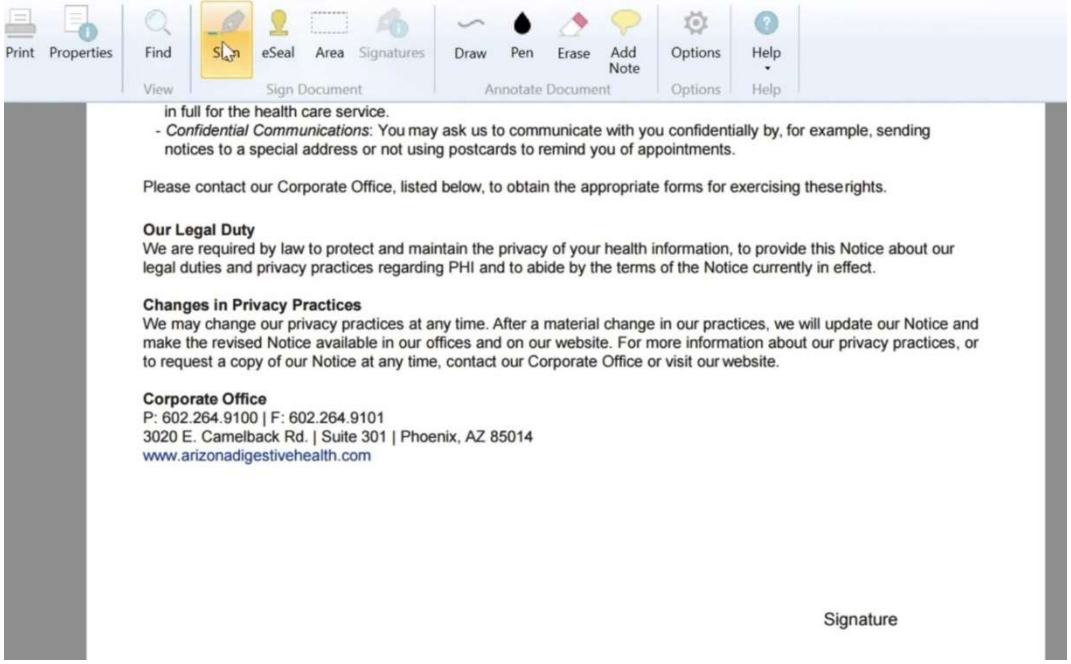
Έχουν δημιουργηθεί πολλές εφαρμογές Form Builder. Κάποιες διατίθενται δωρεάν μέσω ίντερνετ ενώ άλλες είναι επί πληρωμή και έχουν μεγάλο πελατολόγιο. Τέλος, άλλες τέτοιες εφαρμογές χρησιμοποιούν, πέρα από το software, δικό τους hardware .

Σε αυτό το κεφάλαιο θα αναλυθούν κάποιες από αυτές τις εφαρμογές οι οποίες είναι χρήσιμες και δημοφιλείς.

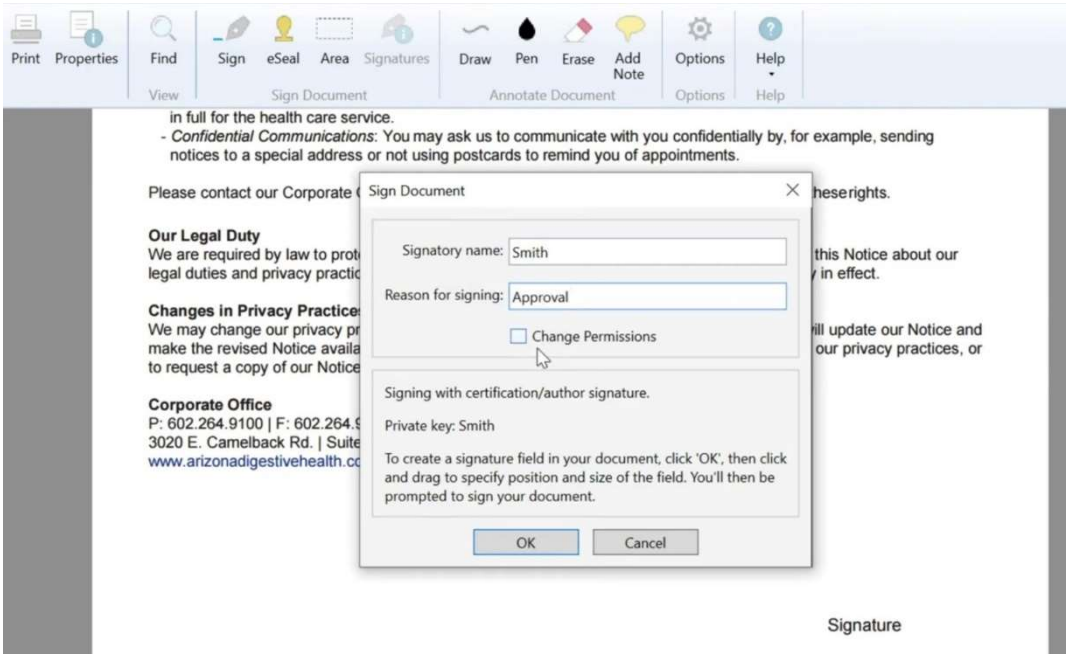
2.2 Wacom

Ένα solution που χρησιμοποιούν ευρέως οι ελληνικές τράπεζες είναι το Wacom. Το Wacom είναι εξειδικευμένο hardware που χρησιμεύει στις ψηφιακές τέχνες, ταινίες, ειδικά εφέ, μόδα και κάθε είδους design καθώς επίσης και για επεξεργασία ψηφιακών εγγράφων. Οι τράπεζες χρησιμοποιούν την τελευταία λειτουργία που αναφέρθηκε, καλώντας τον πελάτη τους να υπογράψει κάποιο έγγραφο. Ο πελάτης με τη σειρά του, χρησιμοποιώντας το ειδικό πενάκι που προσφέρει η Wacom με την ταμπλέτα της, βάζει την υπογραφή του. Παρακάτω βλέπουμε screenshots διαδικασίας υπογραφής εγγράφου με το Wacom.

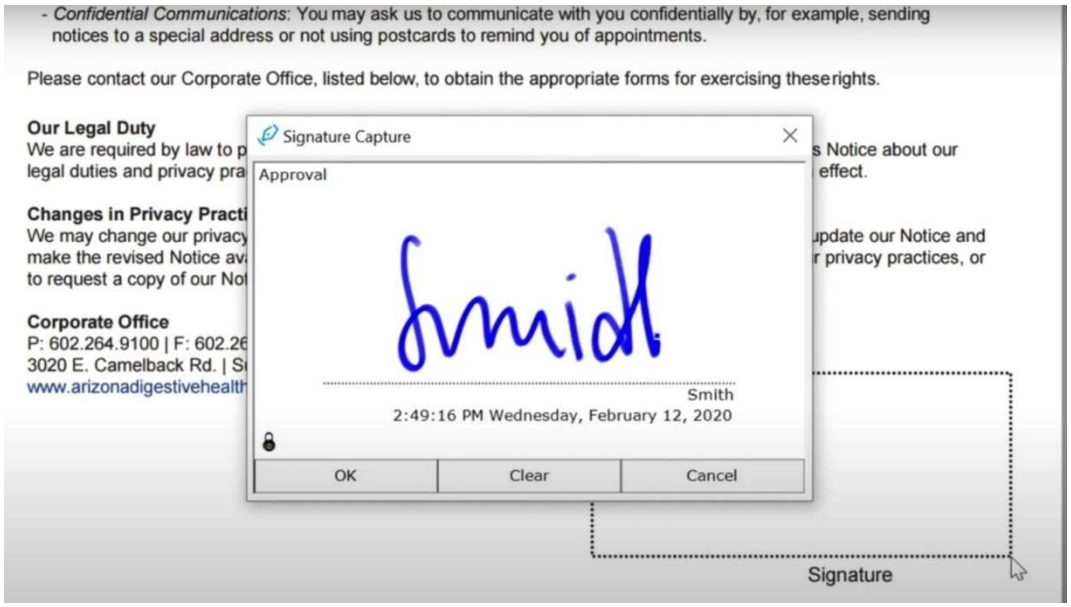
Αρχικά ο χειριστής έχει σαρωμένο το έγγραφο στον υπολογιστή του, το επιλέγει και το ανοίγει στο γραφικό περιβάλλον του Wacom. Εκεί, επιλέγει Sign (Εικόνα 2.1)και ξεκινάει η διαδικασία υπογραφής. Ζητείται από τον χρήστη να δώσει κάποια στοιχεία για την προς συμπλήρωση υπογραφή (Εικόνα 2.2) και τέλος ορίζει μια περιοχή όπου θα μπεί το σκίτσο (υπογραφή) του πελάτη. Ο πελάτης υπογράφει (Εικόνα 2.3) και, τέλος, η υπογραφή μπαίνει στο έγγραφο και έτσι η διαδικασία ολοκληρώνεται.(Εικόνα 2.4).Ο χειριστής μπορεί να επιλέξει να στείλει με email το υπογεγραμμένο έγγραφο (Εικόνα 2.5). [7, 8]



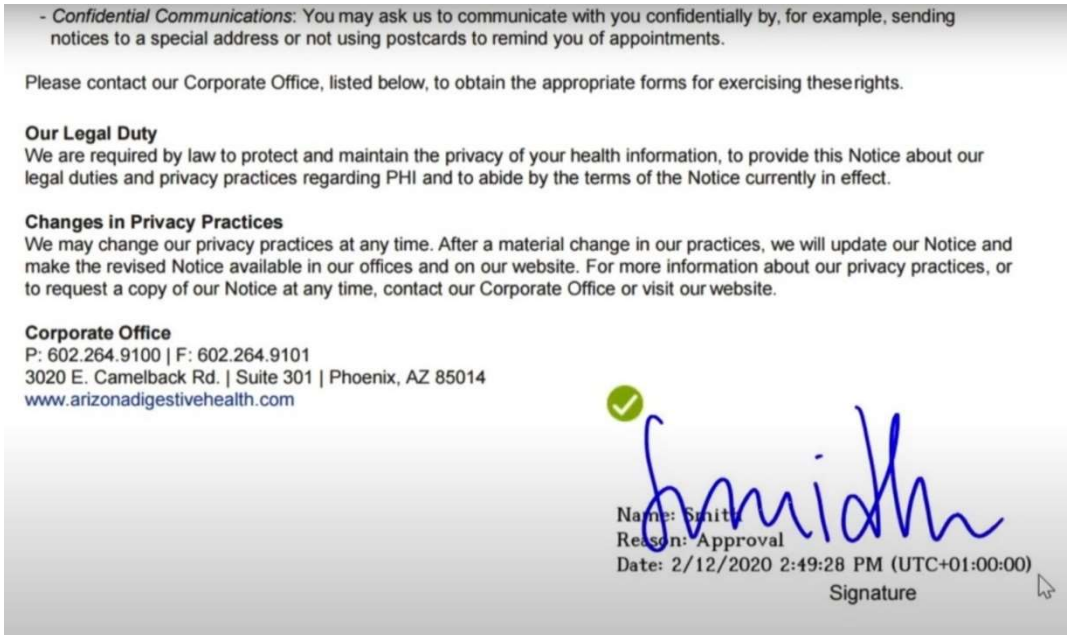
Εικόνα 2.1 Επιλογή Sign



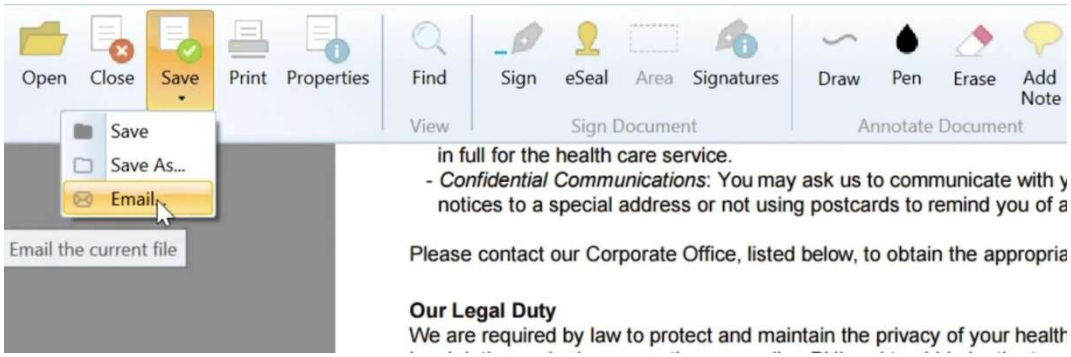
Εικόνα 2.2 Συμπλήρωση στοιχείων



Εικόνα 2.3 Υπογραφή πελάτη



Εικόνα 2.4 Υπογραφή στο έγγραφο

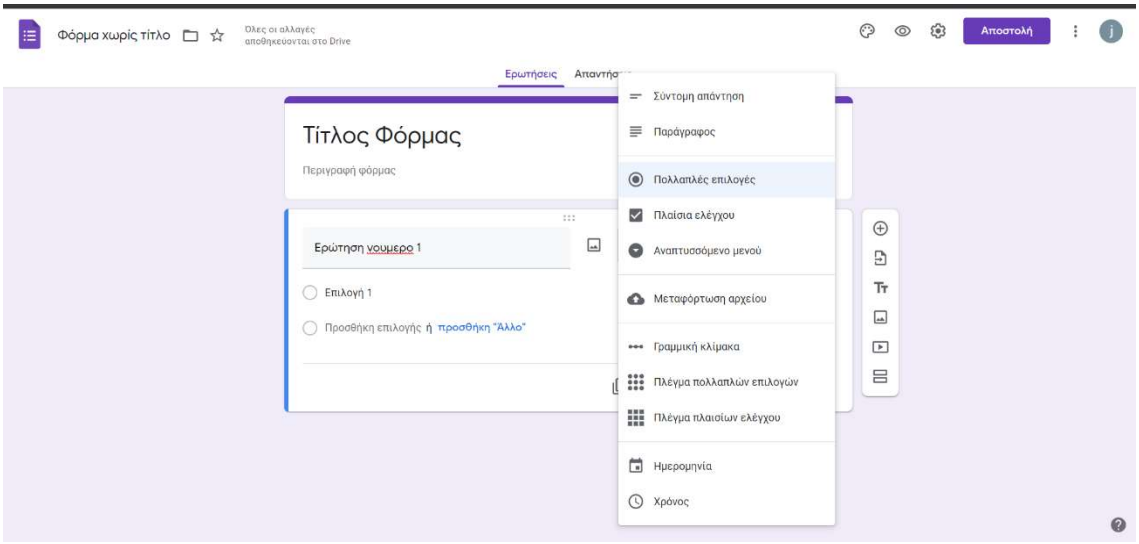


Εικόνα 2.5 Αποστολή με Email

2.3 Google Forms

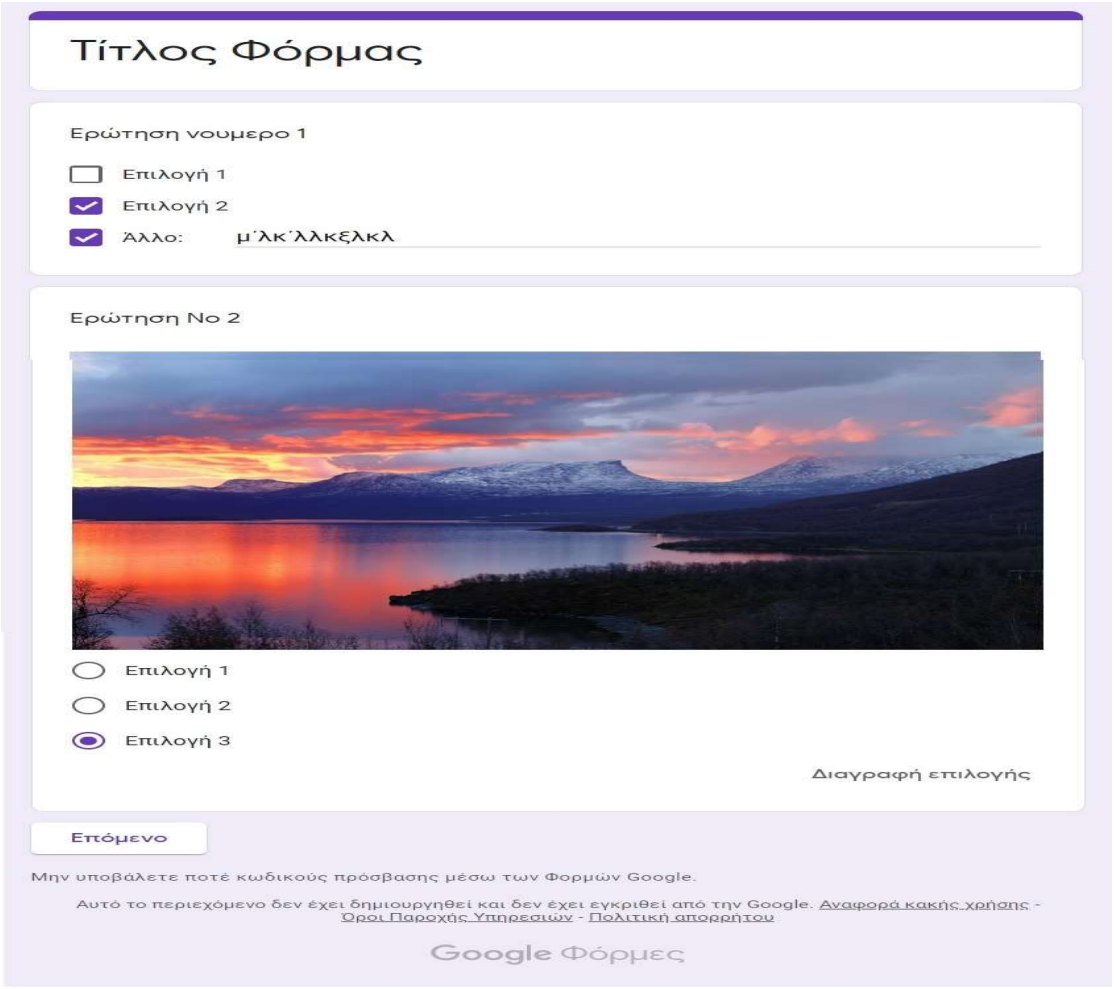
Τα google forms, διατίθενται δωρεάν από την Google.Αν και η Google το εργαλείο αυτό το προτείνει για χρήση διεκπεραίωσης ερευνών, μπορεί να χρησιμοποιηθεί και ως ερωτηματολόγιο και απλή φόρμα συμπλήρωσης πεδίων. Ο χρήστης κατασκευάζει την φόρμα του με προκαθορισμένα πεδία και εργαλεία .Στέλνει την προς υποβολή φόρμα με email και μπορεί να δει στατιστικά των απαντήσεων της φόρμας. [9, 10]

Στην παρακάτω εικόνα (Εικόνα 2.6) βλέπουμε παράδειγμα δημιουργίας φόρμας. Ο χρήστης γράφει τίτλο, περιγραφή και επιλέγει κάποιο από τα συγκεκριμένα controls όπως πολλαπλές επιλογές (Radio Button),πλαίσια ελέγχου (Checkboxes) κλπ.



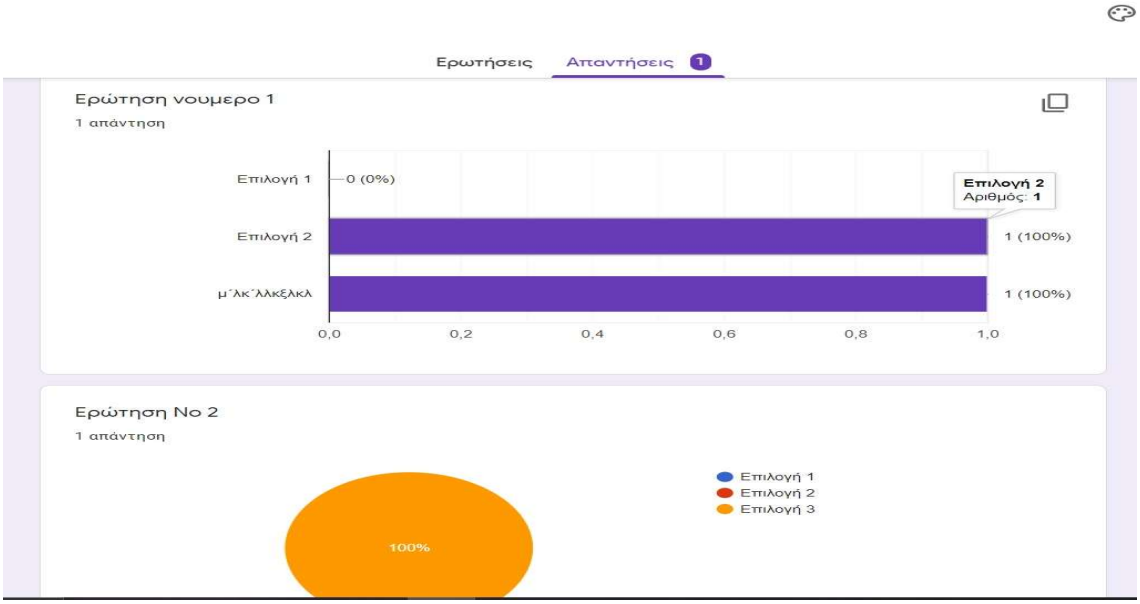
Εικόνα 2.6 Δημιουργία φόρμας

Έπειτα, τη στέλνει με email σε έναν ή περισσότερους πελάτες προς υποβολή. Ο πελάτης συμπληρώνει τα πεδία (Εικόνα 2.7) και την υποβάλλει.



Εικόνα 2.7 Υποβολή φόρμας από χρήστη

Τέλος, ο χρήστης αποκομίζει μια συνολική εικόνα μέσω στατιστικών στοιχείων (Εικόνα 2.8)



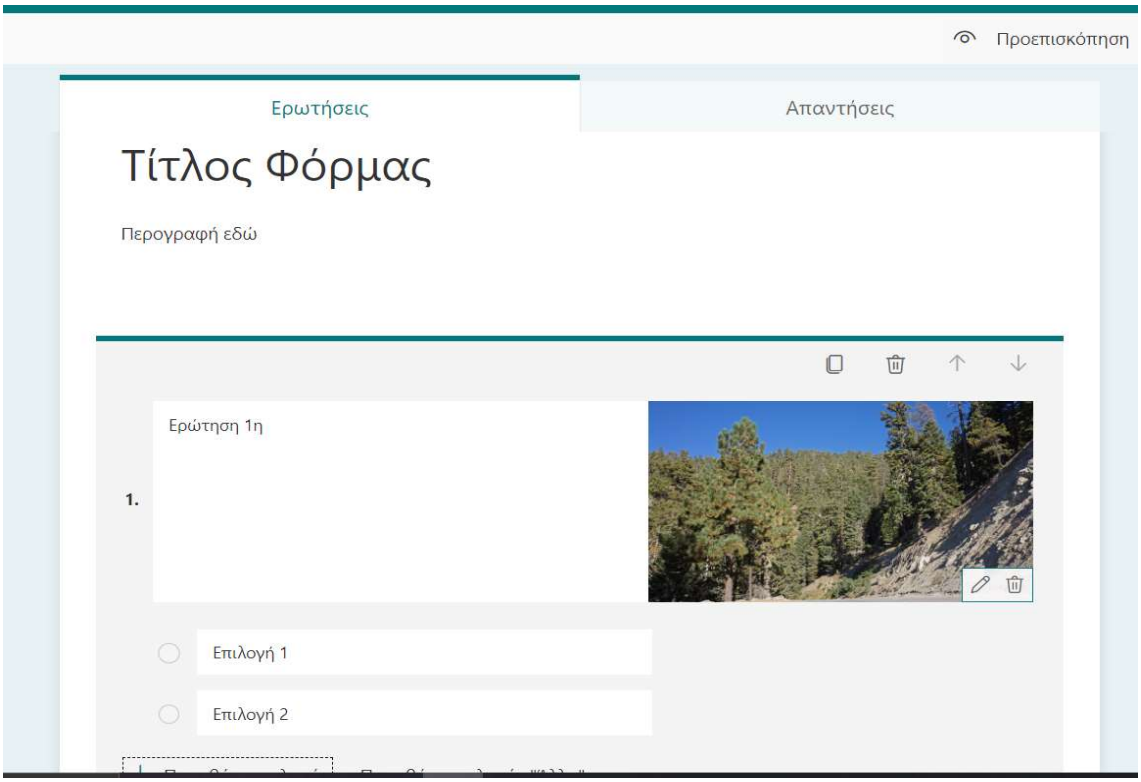
Εικόνα 2.8 Στατιστικά Στοιχεία

2.4 Microsoft Forms

Τα Microsoft Forms μοιάζουν με τα Google Forms και παρουσιάζουν παρόμοια δομή και τρόπο λειτουργίας. Με τα Microsoft Forms ο χρήστης μπορεί να δημιουργήσετε μια φόρμα, όπως έρευνα ή κουίζ, να προσκαλέσετε άλλους να απαντήσουν σε αυτό χρησιμοποιώντας σχεδόν οποιοδήποτε πρόγραμμα περιήγησης ιστού ή κινητή συσκευή, να δεί τα αποτελέσματα σε πραγματικό χρόνο καθώς υποβάλλονται, να χρησιμοποιήσετε ενσωματωμένα εργαλεία analytics για να αξιολογήσει τις απαντήσεις και εξάγει τα αποτελέσματα στο Excel για πρόσθετη ανάλυση ή βαθμολόγηση. [11, 12]

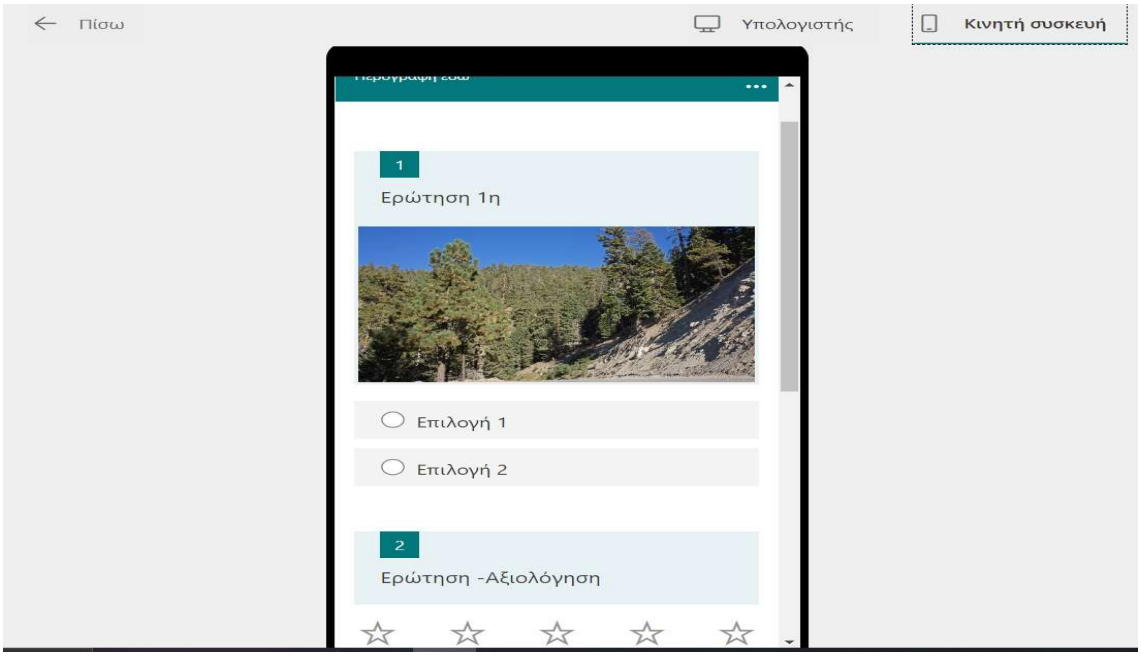
Παρακάτω βλέπουμε ένα αντιπροσωπευτικό παράδειγμα σε screenshots.

Στην εικόνα 2.9 βλέπουμε την δημιουργία φόρμας από τον χρήστη.



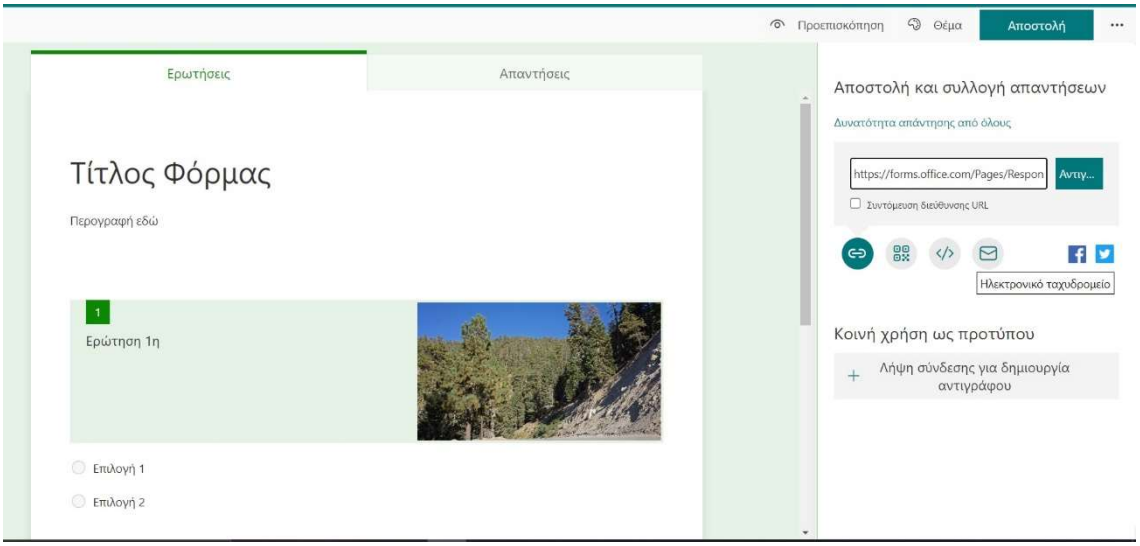
Εικόνα 2.9 Δημιουργία νεας φόρμας

Μόλις ο χρήστης ολοκληρώσει, μπορεί να κάνει μια προεπισκόπηση της φόρμας που δημιούργησε και να δει πως φαίνεται από desktop αλλά και από mobile οθόνη (Εικόνα 2.10)



Εικόνα 2.10 Preview

Ολοκληρώνοντας, ο χρήστης μπορεί να στείλει τη φόρμα μέσω email αλλά και με ένα url (Εικόνα 2.11)

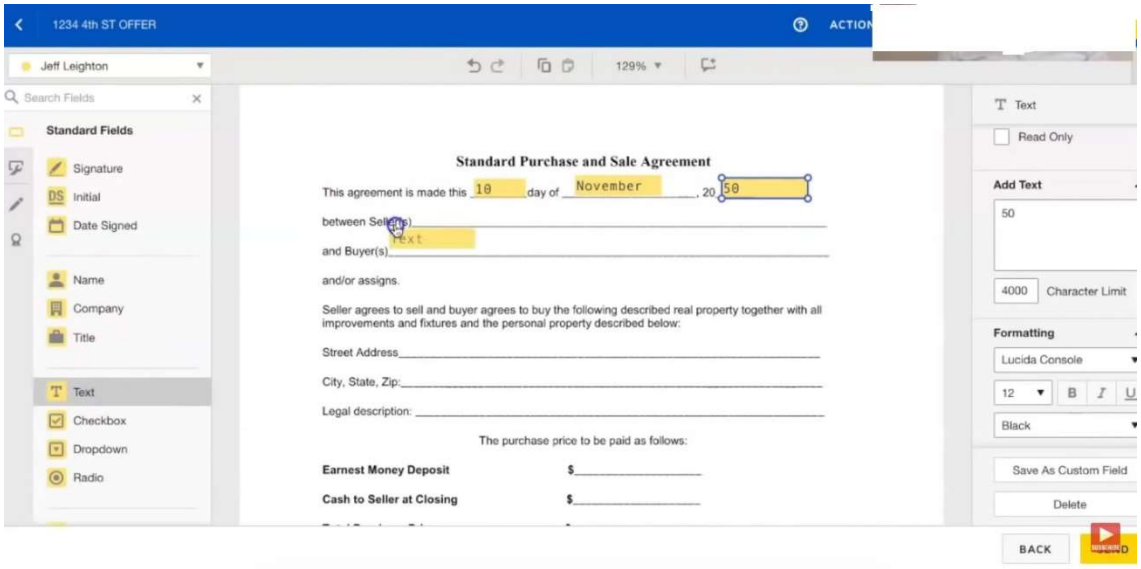


Εικόνα 2.11 Ολοκλήρωση και αποστολή

2.5 DocuSign

Η DocuSign είναι μια αμερικανική εταιρεία με πολυετή εμπειρία στο χώρο του digital signing. Προσφέρει διαφορετικά πακέτα για τις υπηρεσίες τις και έχει και ξεχωριστό section για developers. [13]

Εδώ, η λειτουργία που επιτελεί ο χρήστης είναι η εξής. Πρώτα ανεβάζει το έγγραφο που θέλει να υπογραφεί στην πλατφόρμα. Σημειώνει κάποια στοιχεία του χρήστη που θα παραλάβει το προς υπογραφή έγγραφο και τοποθετεί πεδία ελεύθερα πάνω στο έγγραφο (Εικόνα). Τα πεδία αυτά μπορεί να είναι ελεύθερο κείμενο, ημερομηνία, υπογραφή (free drawing), τίτλος, check και άλλα. Τέλος, στέλνει μέσω email ένα url που ανοίγει μέσα στην πλατφόρμα την φόρμα και εκεί ο τελικός χρήστης αλληλεπιδρά με το έγγραφο. [14]



Εικόνα 2.12 Δημιουργία πεδίων στο προς υπογραφή έγγραφο

2.6 Συμπεράσματα

Είδαμε αρκετά παραδείγματα Form Builder software. Οι λειτουργίες τους μοιάζουν και κάθε ένα επιλύει μια πλειάδα προβλημάτων. Παρακάτω βλέπουμε έναν πίνακα σύγκρισης μεταξύ των software's.

	Wacom	**DocuSign**	**Microsoft Forms**	**Google Forms**	**PowForms**
Free Edition	ΟΧΙ	ΟΧΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ
Χρησιμοποιεί Hardware	ΝΑΙ	ΟΧΙ	ΟΧΙ	ΟΧΙ	ΟΧΙ
Αποστέλλεται ηλεκτρονικά	ΟΧΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ
Δημιουργείς Δικό σου έγγραφο	ΟΧΙ	ΟΧΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ
Χρησιμοποιείς ήδη υπάρχον έγγραφο	ΝΑΙ	ΝΑΙ	ΟΧΙ	ΟΧΙ	ΟΧΙ

Τα θετικά πλεονεκτήματα του PowForms σε σύγκριση με τα υπόλοιπα solutions είναι ότι δεν χρειάζεται συγκεκριμένο hardware και άρα την φυσική σου παρουσία για την συμπλήρωση της φόρμας. Η λειτουργία επιτελείται μέσω οποιουδήποτε web browser. Ακόμα, δεν χρειάζεται να έχεις σαρώσει κάποιο έγγραφο, οπότε και να το έχεις δημιουργήσει σε κάποιο άλλο κειμενογράφο, αλλά σου προσφέρεται η δυνατότητα δημιουργίας στον PowForms κειμενογράφο καθώς και η επιλογή έτοιμων templates. Επίσης, η φόρμα είναι διαθέσιμη μέσω URL το οποίο και ενσωματώνεται στο website του πελάτη, ο οποίος με ελάχιστο programming έχει έτοιμη τη φόρμα για κάθε πελάτη του εν αντιθέσει με τα υπόλοιπα software που πρέπει να δηλώσεις εξ αρχής ποιος θα είναι ο αποδέκτης της φόρμας και να την στείλεις με email. Τέλος, λόγω του custom κειμενογράφου, ο χρήστης φτιάχνει το δικό του έγγραφο, το οποίο έχει όποια μορφή θέλει ο ίδιος σε αντίθεση με τα άλλα software (Google Forms, WinForms) που είναι τυποποιημένα και έχουν μορφή controls το ένα κάτω από το άλλο σε ένα επαναλαμβανόμενο μοτίβο.

Κεφάλαιο 3. Τεχνολογίες Ανάπτυξης

3.1 Razor Pages

Το ASP.NET Razor Pages είναι ένα server-side, page-focused framework. Παρουσιάστηκε ως μέρος του ASP.NET Core, και τώρα περιλαμβάνεται στο .NET 5. Το Razor Pages επιτρέπει τη δημιουργία δυναμικών ιστότοπων που βασίζονται σε δεδομένα με καθαρό διαχωρισμό των λειτουργιών (separation of concerns).Ως μέρος του ASP.NET Core web development Framework της Microsoft, υποστηρίζει ανάπτυξη πολλαπλών πλατφορμών (cross platform) και μπορεί να αναπτυχθεί σε λειτουργικά συστήματα Windows, Unix και Mac.

Το Razor Pages Framework είναι ελαφρύ και πολύ ευέλικτο. Παρέχει στον προγραμματιστή πλήρη έλεγχο του HTML που παράγεται και είναι το προτεινόμενο framework για παραγωγή cross-platform server-side HTML.

Το Razor Pages χρησιμοποιεί τη δημοφιλή γλώσσα προγραμματισμού C # για τον προγραμματισμό από την πλευρά του Server και την εύχρηστη σύνταξη Razor templating syntax για την ενσωμάτωση της C# σε HTML για τη δημιουργία δυναμικού περιεχομένου για τους Browsers.

Από αρχιτεκτονική σκοπιά, το Razor Pages υλοποιεί το MVC pattern και ενθαρρύνει τον διαχωρισμό των λειτουργιών. [15]

3.2 .Net 5

Το σχέδιο ήταν να ενοποιηθούν όλα τα .NET runtimes σε μία πλατφόρμα .NET με ενοποιημένες βιβλιοθήκες base class (BCL) για όλα τα μοντέλα εφαρμογών. Έτσι το .NET 5 θα είναι μια κοινόχρηστη code base για τα .NET Core, Mono, Xamarin και μελλοντικές υλοποιήσεις του .NET.

Αυτό θα έκανε το .NET μια ενοποιημένη πλατφόρμα για όλους τους τύπους εφαρμογών. Ένα project για μια πλατφόρμα θα υποστηρίζει και την ανάπτυξη Windows desktop, Android, and iOS. Ένα Blazor project θα υποστηρίζει εφαρμογές που εκτελούνται σε προγράμματα περιήγησης ιστού, σε κινητά ή σε desktop εφαρμογές.

Πριν από το .NET 5, υπήρχαν εντελώς διαχωρισμένες υλοποιήσεις του .NET (.NET Framework, .NET Core, Xamarin κ.λπ.). Προκειμένου να γράψουμε μια class library που μπορεί να εκτελεστεί σε όλες αυτές τις υλοποιήσεις, έπρεπε να δοθεί ένας στόχος που καθορίζει το σύνολο των κοινόχρηστων API, το οποίο είναι ακριβώς το .NET Standard.

Αυτό σημαίνει ότι κάθε φορά που θέλουμε να προσθέσουμε ένα νέο API, πρέπει να δημιουργήσουμε μια νέα έκδοση του .NET Standard και στη συνέχεια να εργαστούμε με όλες τις πλατφόρμες .NET για να διασφαλίσουμε ότι προσθέτουν υποστήριξη για αυτήν την έκδοση. Το πρώτο πρόβλημα είναι ότι αυτή η διαδικασία δεν είναι πολύ γρήγορη. Το άλλο πρόβλημα είναι ότι αυτό απαιτεί μια αποκωδικοποίηση ως προς το ποια version ποιας .Net πλατφόρμας η υποστήριξη χρειάζεται ποια version της έκδοσης.

Αλλά με το .NET 5, η κατάσταση είναι πολύ διαφορετική. Έχουμε πλέον μια κοινόχρηστη code base για όλα τα .NET workloads, είτε πρόκειται για desktop εφαρμογές, cloud services είτε για εφαρμογές για κινητά. Και σε έναν κόσμο όπου όλα τα .NET workloads λειτουργούν στην ίδια .NET stack, δεν χρειάζεται να διαχωρίσουμε τεχνητά το έργο σε ορισμό API και εργασία υλοποίησης. Προσθέτουμε απλώς το API στο .NET και την επόμενη φορά που το stack αποστέλλεται, η υλοποίηση είναι άμεσα διαθέσιμη για όλα τα workloads.

Αυτό δεν σημαίνει ότι όλα τα workloads θα έχουν την ίδια επιφάνεια API, γιατί απλά δεν θα λειτουργούσε. Για παράδειγμα, το Android και το iOS διαθέτουν τεράστιο αριθμό OS APIs. Θα μπορούν να κληθούν μόνο όταν τρέχουν στις κατάλληλες συσκευές.

Βελτιώσεις απόδοσης στο .NET 5

Πολύ δουλειά έγινε για να επιτευχθεί καλύτερη και με μεγαλύτερη συνέπειας, απόδοση.

- Το κλιμακωτό compile είναι μια προσέγγιση που επιτρέπει ταχύτερο αρχικό compilation χωρίς να εκτελεί βελτιστοποιήσεις την πρώτη φορά που μια μέθοδος μεταγλωττίζεται. Οι μέθοδοι που χρησιμοποιούνται συχνά μπορούν στη συνέχεια να μεταγλωττιστούν με υψηλότερη ποιότητα εφαρμόζοντας περαιτέρω βελτιστοποιήσεων. Στο .NET 5, τέτοιες μέθοδοι αναγνωρίζονται πλέον καλύτερα με την καταμέτρηση κλήσεων. Οι μέθοδοι με βρόχους λαμβάνουν ειδική μεταχείριση με βελτιστοποίηση αμέσως λόγω του υψηλού αντίκτυπου που θα μπορούσαν να έχουν ακόμη και αν κληθούν μόνο μία φορά.
- Ο RyuJIT compiler περιέχει πολλές άλλες βελτιώσεις δημιουργίας κώδικα. Μια ειδική κατηγορία αυτών των βελτιώσεων είναι στο hardware, δηλ. Υποστήριξη για στόχευση συγκεκριμένων συνόλων οδηγιών συγκεκριμένων επεξεργαστών (π.χ. SSE και AVX).
- Οι βελτιώσεις στον garbage collector συμβάλλουν στη μείωση των χρόνων παύσης της εφαρμογής όταν εκτελείται η συλλογή απορριμμάτων και βελτιστοποιούν τη σάρωση της διαχειριζόμενης μνήμης (managed memory) του garbage collector για τους υποψηφίους να υποβληθούν σε συλλογή.
  Εκτός από αυτό, πολλά API βελτιστοποιούνται επίσης για καλύτερη απόδοση, κυρίως:
  • Υλοποίηση HTTP 1.1 και HTTP / 2,
  • Regular Expressions και
  • λειτουργίες με Strings.

Υποστήριξη για το Cloud

Αρκετές αλλαγές αποσκοπούν ειδικά στη βελτίωση της απόδοσης στα σενάρια των container workload.

• Γενικά καλύτερες επιδόσεις .NET σε περιβάλλοντα container.

• Μείωση του μεγέθους των δημοσιευμένων Images και μεγαλύτερη επιλογή διαθέσιμων Images.

• Υποστήριξη API για orchestration όπως το OpentelEmetry για διευκόλυνση της εργασίας με το .NET σε τέτοια περιβάλλοντα.

Υποστήριξη για ARM64 αρχιτεκτονική

Το .NET Core 3.1 ήταν ήδη διαθέσιμο για Linux ARM64 ήταν πλήρως λειτουργικό με την έκδοση X64. Από την άποψη της απόδοσης, ωστόσο, είχε μείνει πίσω. Για να αλλάξει αυτό στο .NET 5, πραγματοποιήθηκαν αρκετές σημαντικές επενδύσεις σε αυτόν τον τομέα:

• Βελτιστοποίηση μεταγλωττιστών JIT για την ARM64.

• Υποστήριξη για το Hardware Arm64, δηλ. Ειδικά σύνολα εντολών.

• Προσαρμοσμένοι αλγόριθμοι κρίσιμων επιδόσεων για το ARM64.

Δεν υπήρξε native υποστήριξη για το Windows Arm64 στο .NET Core 3.1 που σήμαινε ότι όλες οι εφαρμογές .NET Core και .NET θα μπορούσαν να τρέχουν μόνο σε λειτουργία εξομοίωσης X86 σε αυτές τις συσκευές. Στο .NET 5, παρέχεται πλήρης native υποστήριξη για το Windows Arm64 (σε συσκευές όπως το Surface Pro X), τόσο για ανάπτυξη λογισμικού όσο και για τις εφαρμογές. Η αρχική έκδοση δεν περιλαμβάνει το στοιχείο των Windows Desktop (Forms Windows και WPF). Υπάρχουν σχέδια να προστεθούν σε μια μελλοντική ενημέρωση.

Ενημερώσεις στις γλώσσες προγραμματισμού

Το .NET 5 συνοδεύεται επίσης από ορισμένες αλλαγές στις υποστηριζόμενες γλώσσες προγραμματισμού:

• Το C # 9 φέρνει πολλά νέα features που επικεντρώνονται στη βελτίωση της εμπειρίας της εργασίας με immutable δομές δεδομένων. Οι πιο αξιοσημείωτες αλλαγές είναι η εισαγωγή των long awaited record types και περαιτέρω βελτιώσεις για pattern matching.

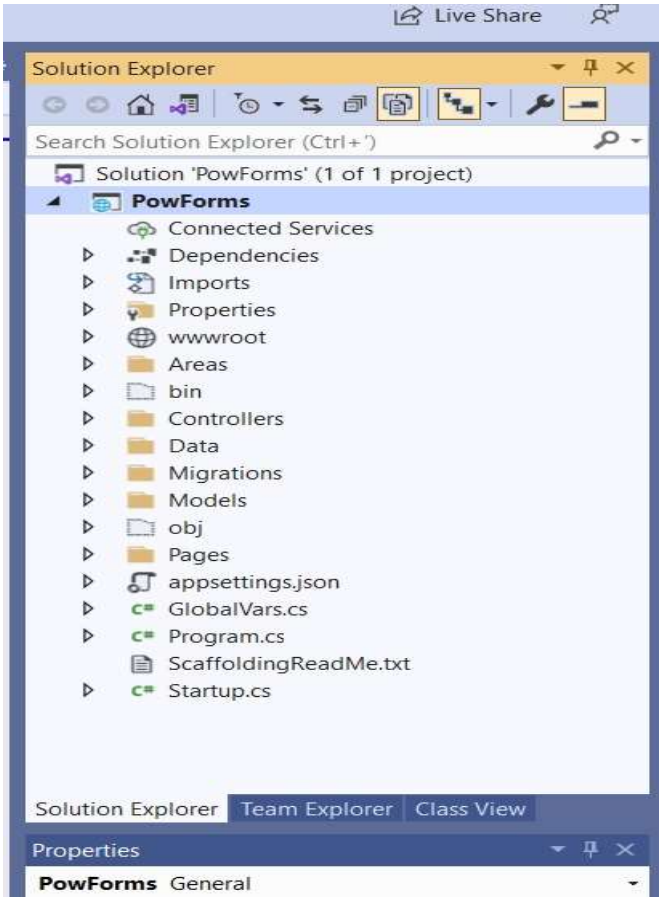
• Η F # 5 επικεντρώνεται στις βελτιώσεις στον διαδραστικό και αναλυτικό προγραμματισμό για να κάνει την εμπειρία του Jupyter Notebooks ακόμα καλύτερα.

• Η Visual Basic δεν θα αναπτυχθεί περαιτέρω ως γλώσσα. Για να διευκολυνθεί η μεταφορά εφαρμογών που βασίζονται στο Visual Basic .NET, περισσότεροι τύποι project θα λειτουργούν με την Visual Basic στο .NET 5: Τα Windows Forms και WPF, πέρα από τις βιβλιοθήκες κλάσης και τις εφαρμογές κονσόλας που έχουν ήδη υποστηριχθεί στο.NET Core 3.1. Δεν θα υποστηριχθούν νέες λειτουργίες του .NET Core που θα απαιτούσαν μια αλλαγή στη Visual Basic. [16], [17]

3.3 Medium-editor

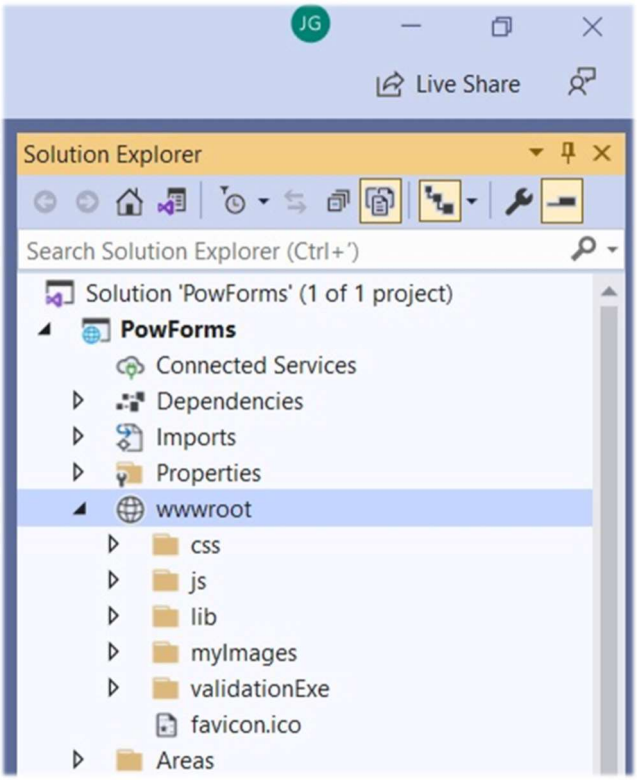
Το medium text editor (κειμενογράφος) είναι ένας WYSIWYG editor ανοικτού κώδικα, γραμμένος σε javascript. Δίδει τη δυνατότητα στον προγραμματιστή να εισάγει html, είτε σε επιλεγμένο κείμενο είτε στον κέρσορα, οπότε και την δυνατότητα να πραγματοποιήσει κάθε είδους μορφοποίηση στο κείμενο.

3.4 Δομή Ενός ASP.NET Core Web App



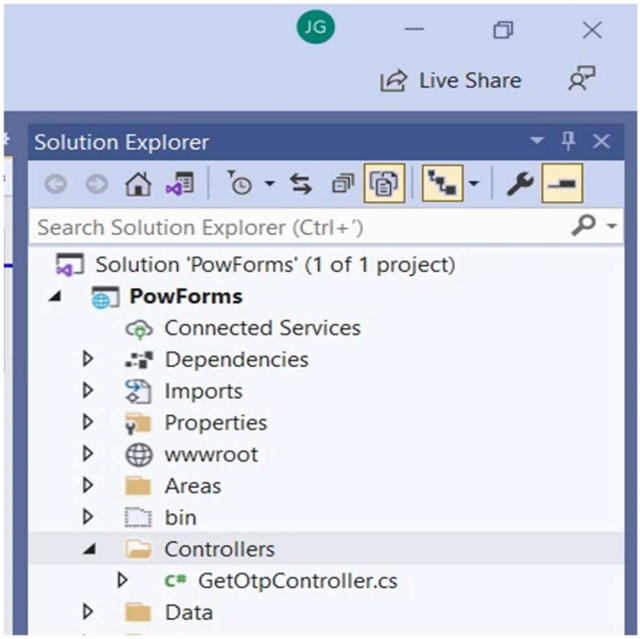
Εικόνα 3.1 Δομή μιας .NET Core Webb App

Το κυρίως σώμα μιας ASP.NET Core Razor Pages Web App αποτελείται από τους φακέλους Properties, wwwroot και Pages.Οι υπόλοιποι φάκελοι που φαίνονται στην εικόνα 3.1 έχουν προστεθεί από τον προγραμματιστή για την καλύτερη δόμηση του project.Εξίσου σημαντικό ρόλο παίζουν τα αρχεία appsettings.json και Startup.cs. Το πρώτο κρατάει κάποιες παραμέτρους για το project καθώς και τα connection Strings με τις DataBases. Το δεύτερο αρχείο περιέχει όλο το configuration του project, όπως τα cookie, configs για τα razor pages, το authorization σε περίπτωση που χρησιμοποιηθεί το Identity, το DatabaseContext για το Entity Framework (το default ORM framework της Microsoft) κ.α. Τέλος, εκεί μπορείς να τρέξεις κώδικα που τρέχει κατά την πρώτη εκτέλεση της εφαρμογής (Όπως δημιουργία ρόλων κλπ).



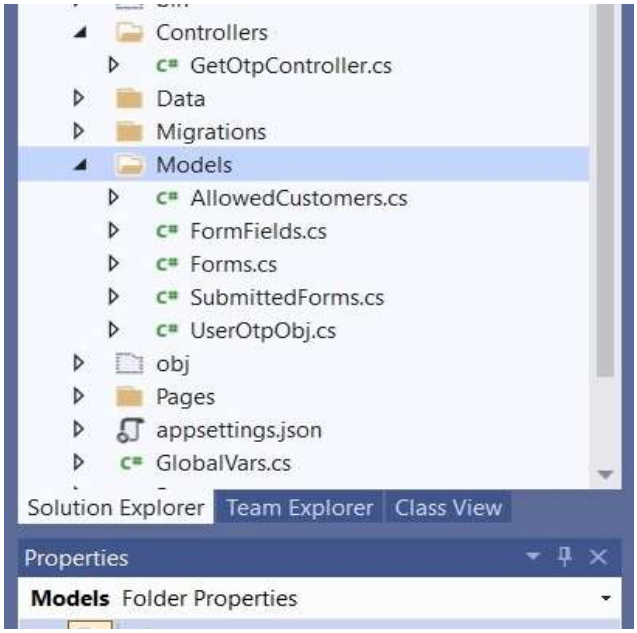
Εικόνα 3.2 Φάκελος wwwroot

Στο φάκελο wwwroot (Εικόνα 3.2) κρατούνται όλα τα αρχεία css, javascript, external images καθώς και στο συγκεκριμένο project, το validation.exe που κατόπιν εντολής του admin, ελέγχει αν τα data στην database έχουν υποστεί αλλαγές πιθανώς για κακόβουλο σκοπό.



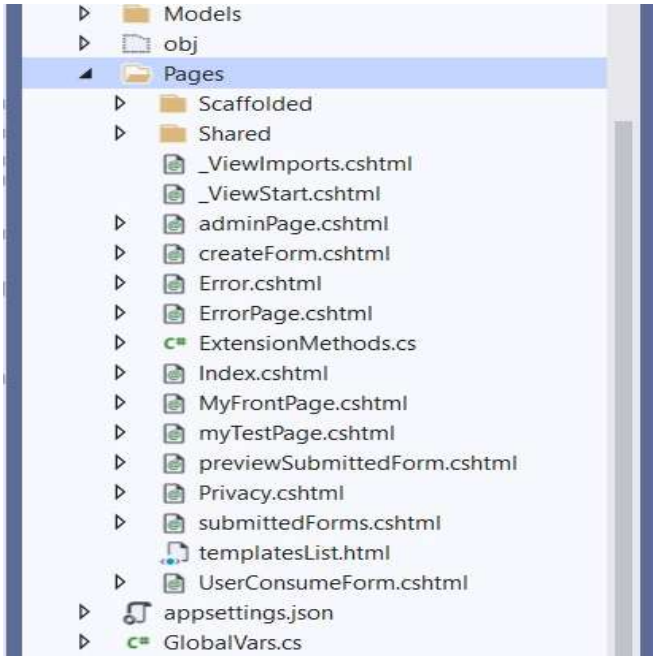
Εικόνα 3.3 Φάκελος Controllers

Ο φάκελος Controllers (Εικόνα 3.3) περιέχει έναν controller που εξυπηρετεί ένα RESTfull HTTP Get service. Παράγει και επιστρέφει έναν One Time Password για τους τελικούς χρήστες της PowForms πλατφόρμας.



Εικόνα 3.4 Φάκελος Models

Ο φάκελος Models (Εικόνα 3.4) περιέχει κλάσεις οι οποίες αντικατοπτρίζουν τους αντίστοιχους πίνακες της DataBase. Object αυτών των κλάσεων χρησιμοποιούνται εκτενώς για λειτουργίες CRUD καθώς και για DataBinding του Backend με το Frontend.



Εικόνα 3.5 Φάκελος Pages

Στον φάκελο Pages (Εικόνα 3.5) περιέχονται όλα τα Razor Pages που δημιουργεί ο προγραμματιστής. Δηλαδή οι ιστοσελίδες-frontend μίξη HTML και C# με το backend τους σε C#.

Κεφάλαιο 4 Σχεδίαση συστήματος

4.1 Σύντομη περιγραφή

Η εφαρμογή που αναπτύχθηκε είναι μια web based πλατφόρμα κατασκευής φορμών συμπλήρωσης καθώς και χρήσης τους από τελικούς χρήστες.

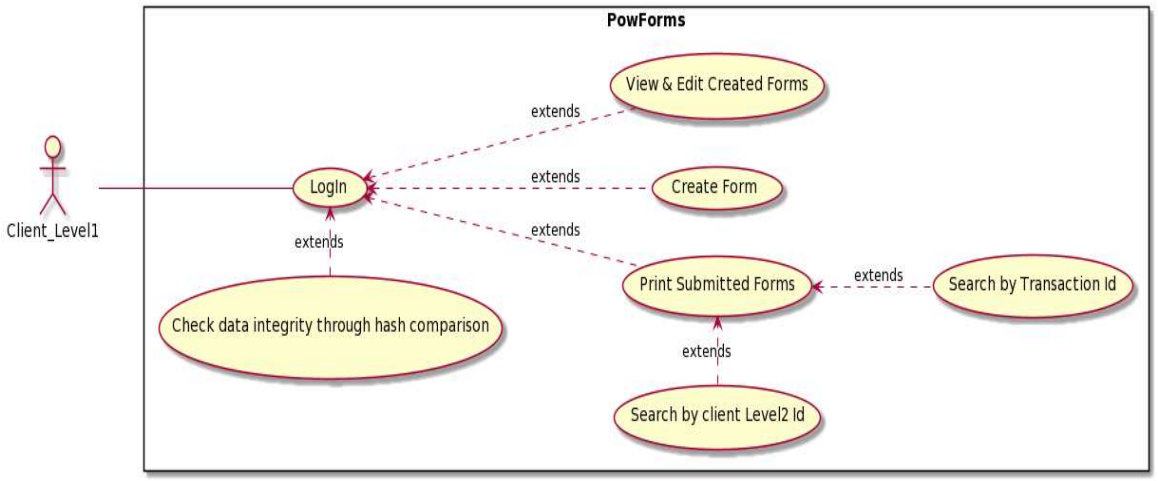
Ο πελάτης της εφαρμογής, χρησιμοποιεί την πλατφόρμα για να δημιουργήσει τις δικές του φόρμες συμπλήρωσης, για παράδειγμα φόρμα GDPR Consent, χρησιμοποιώντας τον HTML Editor της εφαρμογής. Εφόσον ολοκληρώσει τη φόρμα του, λαμβάνει ένα παραγόμενο link. Βάζοντάς το στη δική του εφαρμογή, οι δικοί του χρήστες μπορούν να χρησιμοποιήσουν την φόρμα. Ακόμα, μπορεί να δει το σύνολο των συμπληρωμένων φορμών των πελατών του καθώς και να τις εκτυπώσει.

Οι τελικοί χρήστες, έρχονται στην εφαρμογή και βλέπουν την προς συμπλήρωση φόρμα χρησιμοποιώντας το παραγόμενο link. Εκεί, είτε συμπληρώνουν τα πεδία και υποβάλλουν την φόρμα είτε την απορρίπτουν. Για κάθε περίπτωση, γίνεται redirect πίσω στην σελίδα του πελάτη σε προσυμφωνημένη σελίδα (success page, failure page).

Οι Συμπληρωμένες φόρμες κρατούνται στην εφαρμογή, απομπλέκοντας έτσι τον πελάτη από το κόστος της διαδικασίας τήρησης δεδομένων σε δικό του server, backup και ασφάλειας δεδομένων.

4.2 Διαγράμματα Use Case

Use Case 1 – Αλληλεπίδραση χρήστη Level 1 με την πλατφόρμα PowForms:



Εικόνα 4.1 Use Case Diagram 1

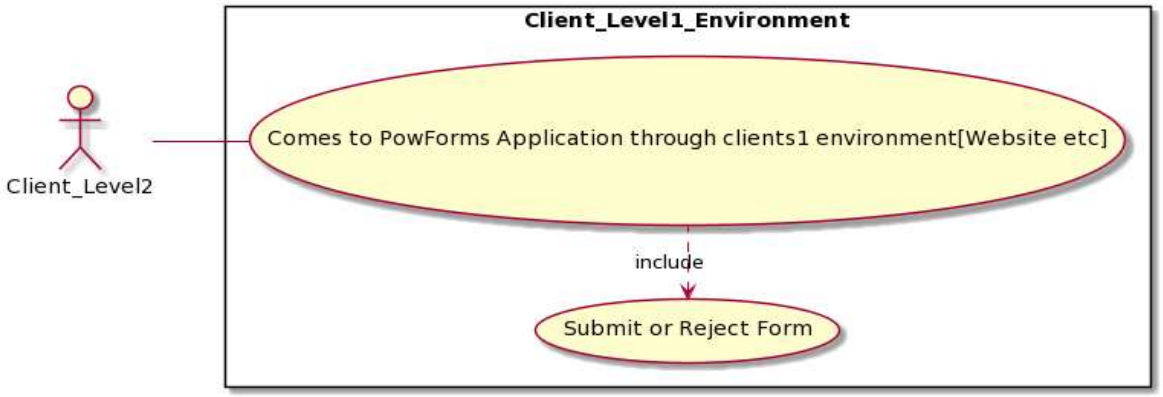
Βασικό σενάριο επιτυχίας

1. Ο Client_Level1 κάνει Log in στο σύστημα.
2. Φτιάχνει μια φόρμα

Εναλλακτικά σενάρια

1. Προβολή και επεξεργασία των φορμών που έχει δημιουργήσει.
2. Εκτύπωση συμπληρωμένων φορμών των πελατών του.
3. Διαπίστωση αλλοίωσης δεδομένων στη Database.

Use Case 2 – Αλληλεπίδραση χρήστη Level 2 με την πλατφόρμα PowForms:



Εικόνα 4.2 Use Case Diagram 2

Βασικό σενάριο επιτυχίας

1. Ο Χρήστης επιτυχώς, ανακατευθύνεται στο περιβάλλον της εφαρμογής
2. Αποδέχεται ή όχι τη φόρμα.

Για τα διαγράμματα χρησιμοποιήθηκε η Plant Uml. Ο κώδικας για τα παραπάνω διαγράμματα είναι ο εξής.

Use Case Diagram 1:

```
@startuml
left to right direction
skinparam packageStyle rectangle
actor Client_Level1
rectangle PowForms {
  Client_Level1 -- (LogIn)
(Create Form) as UC3
  UC3 -up.> (LogIn) : extends
  (View & Edit Created Forms) -up.> (LogIn) : extends
  (Print Submitted Forms) -up.> (LogIn) : extends
  (Check data integrity through hash comparison) .> (LogIn) : extends
          (Search by client Level2 Id).> (Print Submitted Forms)  : extends
          (Search by Transaction Id) -up.> (Print Submitted Forms)  : extends
}
@enduml
```

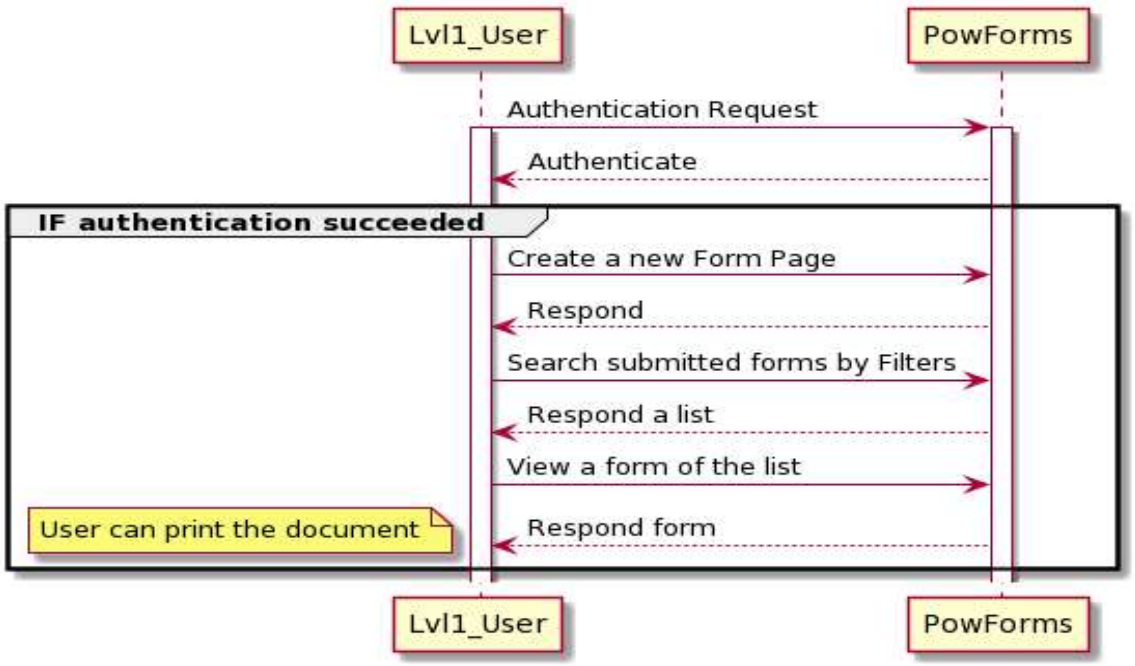
Use Case 1 Plant Uml Code

Use Case Diagram 2:

```
@startuml
left to right direction
skinparam packageStyle rectangle
actor Client_Level2
rectangle Client_Level1_Environment{
  Client_Level2 -- (Comes to PowForms Application through clients1 environment[Website etc])
(Comes to PowForms Application through clients1 environment[Website etc]) -left.> (Submit or Reject
Form): include
}
@enduml
```

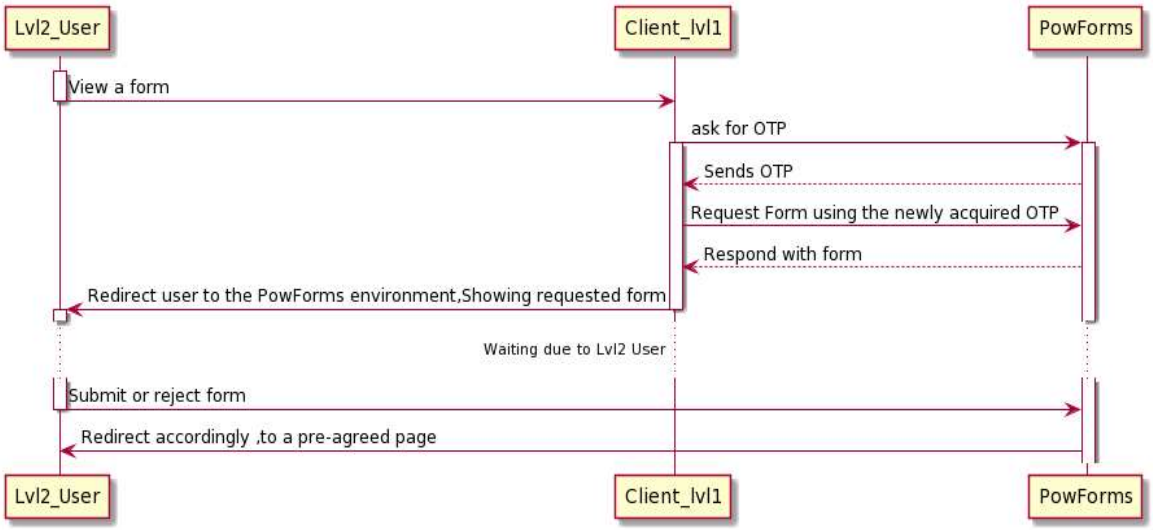
Use Case 2 Plant Uml Code

4.3 Ακολουθιακά Διαγράμματα



Εικόνα 4.3 Sequence Diagram 1.

Στο διάγραμμα ακολουθίας 1, βλέπουμε ένα πιθανό σενάριο αλληλεπίδρασης του χρήστη με την εφαρμογή. Ο χρήστης αφού αυθεντικοποιηθεί επιτυχώς, εισέρχεται στο σύστημα. Δημιουργεί νέα φόρμα και μετά, ζητάει να δει μια φόρμα από τις συμπληρωμένες την οποία μπορεί και να την εκτυπώσει.



Εικόνα 4.4 Sequence Diagram 2

Στο διάγραμμα ακολουθίας 2, βλέπουμε ολοκληρωμένο παράδειγμα χρήσης της εφαρμογής από τον τελικό πελάτη (Lvl2_User) και πώς η εφαρμογή μας αλληλεπιδρά με την εφαρμογή του πελάτη της (Client_lvl1). Μέσα από την εφαρμογή του Client_lvl1, ο τελικός πελάτης ζητάει να δει την φόρμα.

Τότε η εφαρμογή του Client_lvl1, ζητάει ένα One time password από την PowForms και βάση αυτού, ζητάει την φόρμα προς συμπλήρωση. Κατόπιν, ανακατευθύνει τον τελικό πελάτη σε αυτήν την φόρμα όπου και τελικά θα την αποδεχθεί με εισαγωγή δεδομένων ή θα την απορρίψει.

Για τα διαγράμματα χρησιμοποιήθηκε η Plant Uml. Ο κώδικας για τα παραπάνω διαγράμματα είναι ο εξής.

Sequence diagram 1:

```
@startuml
Lvl1_User-> PowForms: Authentication Request
activate Lvl1_User
activate PowForms
PowForms --> Lvl1_User: Authenticate
group IF authentication succeeded
   Lvl1_User -> PowForms : Create a new Form Page
   PowForms --> Lvl1_User : Respond
   Lvl1_User-> PowForms : Search submitted forms by Filters
   PowForms --> Lvl1_User : Respond a list
   Lvl1_User-> PowForms : View a form of the list
   PowForms --> Lvl1_User : Respond form
   note left
      User can print the document
   end note
end
@enduml
```

Sequence Diagram 1 Plant Uml Code

Sequence diagram 2:

```
@startuml
Activate Lvl2_User
Lvl2_User-> Client_lvl1: View a form
Deactivate Lvl2_User
Client_lvl1 -> PowForms : ask for OTP
Activate Client_lvl1
activate PowForms
PowForms --> Client_lvl1: Sends OTP
Client_lvl1 -> PowForms : Request Form using the newly acquired OTP
PowForms --> Client_lvl1:Respond with form
Client_lvl1 -> Lvl2_User : Redirect user to the PowForms environment,Showing requested form
DEActivate Client_lvl1
Activate Lvl2_User
...Waiting due to Lvl2 User...
 Lvl2_User -> PowForms:Submit or reject form
 DEActivate Lvl2_User
 PowForms -> Lvl2_User:Redirect accordingly, to a pre-agreed page
@enduml
```

Sequence Diagram 2 Plant Uml Code

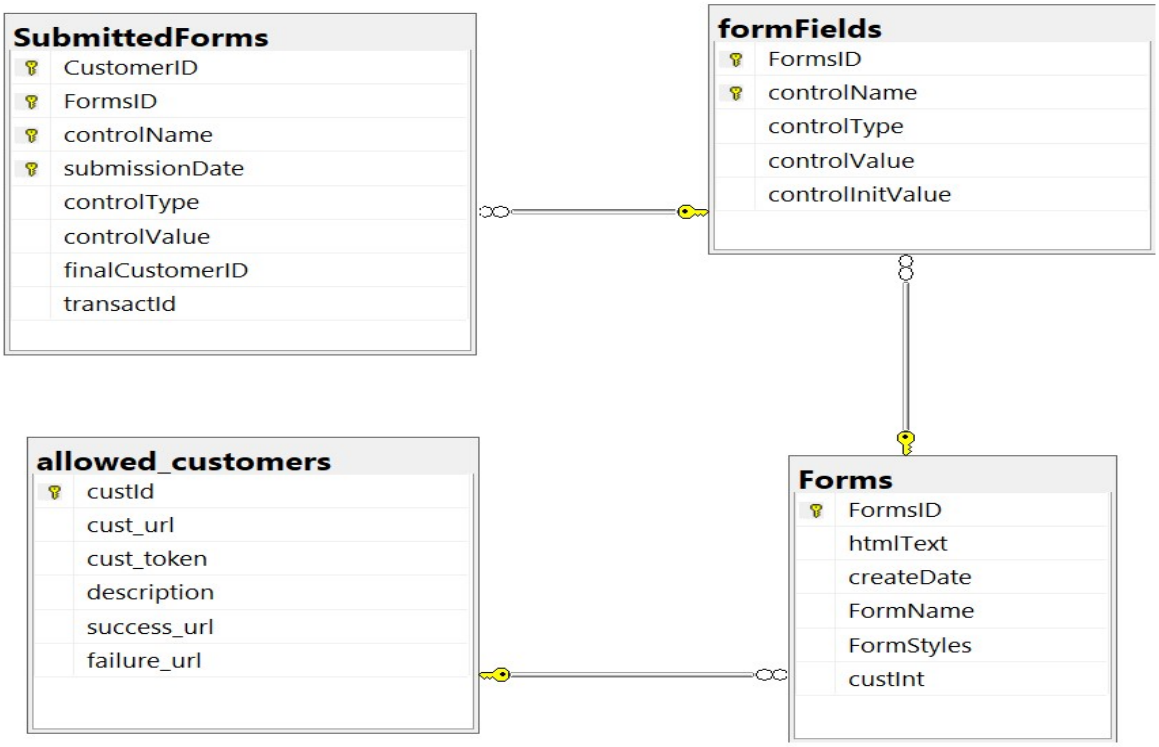
4.4 Ανάλυση Tables και Models

Οι σχεσιακές βάσεις δεδομένων που έχουν χρησιμοποιηθεί για το συγκεκριμένο project είναι δύο.

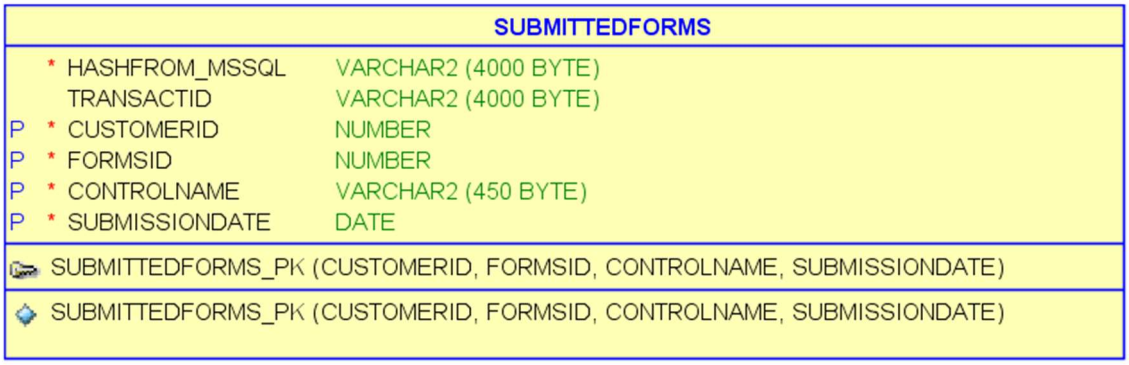
Η κύρια βάση που τρέχει όλο το project είναι η Microsoft SQL Server. Περιέχει όλους τους πίνακες για την λειτουργία της εφαρμογής καθώς και τα data των χρηστών.

Η δευτερεύουσα βάση είναι η Oracle Database. Σκοπός αυτής, είναι να κρατούνται τα primary keys καθώς και όλο το row του πίνακα submittedForms σε μορφή Hash για έλεγχο, τυχόν κακόβουλης η μη, αλλαγής δεδομένων στην κύρια βάση.

Παρακάτω βλέπουμε τα ER Diagrams για τις 2 αυτές βάσεις.



Εικόνα 4.5 SQLServer E-R Diagram



Εικόνα 4.6 Oracle Db E-R Diagram

Κεφάλαιο 5 Υλοποίηση Συστήματος

5.1 Γενική Αρχιτεκτονική

Στις ενότητες που ακολουθούν θα αναλυθούν περαιτέρω οι Εικόνα 3.3 και Εικόνα 3.4 , που είναι τα μοντέλα που χρησιμοποιούνται στα PowForms καθώς και ο controller που χρησιμοποιείται για την λειτουργία του Timed OTP. Τέλος, ακολουθεί η περιγραφή ροής που δείχνει αναλυτικά δύο παραδείγματα χρήσης των PowForms.

5.2 Υλοποίηση Μοντέλου

Τα models που χρησιμοποιήθηκαν αντικατοπτρίζουν τη βάση και ο κώδικας που χρησιμοποιήθηκε είναι ο εξής:

Πίνακας Forms (Περιέχει όλη τη φόρμα που δημιούργησε ο χρήστης σε Html μορφή):

```
public class Forms
{
    public int FormsID { get; set; }
    public String htmlText { get; set; }
    public String  FormName { get; set; }
    public DateTime createDate { get; set; }
    public String FormStyles { get; set; }
    public int? custInt { get; set; }
    public ICollection<FormFields> formFieldsList { get; set; }
    public ICollection<SubmittedForms> submitedList { get; set; }
}
```

- **htmlText:** Η φόρμα που έχει δημιουργήσει ο χρήστης σε Html text
- **FormName:** Τίτλος φόρμας που δίνει ο χρήστης
- **CreateDate:** Ημερομηνία δημιουργίας φόρμας
- **FormStyles:** Styles για την φόρμα, όπως τα περιθώρια,περίγραμμα κ.α.
- **custInt:**Ο κωδικός πελάτη που δημιούργησε τη φόρμα

Πίνακας FormFields (Περιέχει τα controls της φόρμας πχ textbox,checkbox κλπ.):

```
public class FormFields
    {
        public int FormsID { get; set; }
        public String controlType { get; set; }
        public String controlName { get; set; }
        public String controlValue { get; set; }
        public string controlInitValue { get; set; }

        public Forms momForm { get; set; }
    }
```

- **controlType:**Τύπος Control (textbox,checkbox,swich-options)
- **controlName:**Αυτόματο όνομα που παίρνει το κάθε control
- **controlInitValue:**Αρχική τιμή που έχει το κάθε control (πχ το αρχικό κείμενο που θα εμφανίζεται σε ένα textbox)

Πίνακας SubmittedForms (Περιέχει τις συμπληρωμένες φόρμες από τους χρήστες):

```
public class SubmittedForms
    {

        public int CustomerID { get; set; }
        public int FormsID { get; set; }
        public String controlType { get; set; }
```

```
        public String controlName { get; set; }
        public String controlValue { get; set; }
        public DateTime submissionDate { get; set; }
        public String finalCustomerID { get; set; }
        public String transactId { get; set; }
        public Forms formUsed { get; set; }


    }
```

- **transactId:** Είναι ένας μοναδικός κωδικός που παράγεται κατά την αποθήκευση της συμπληρωμένης φόρμας στην βάση (Ο χρήστης κάνει submit επιτυχώς).
- Τα υπόλοιπα πεδία είναι πεδία υπολοίπων πινάκων και εξηγούνται σαυτούς αντίστοιχα.

Πίνακας AllowdCustomers (Περιέχει δεδομένα για τους πελάτες της εφαρμογής απαραίτητα για την ορθή λειτουργία καθώς και τις σελίδες success page και failure page που γίνονται redirect οι χρήστες ανάλογα με την υποβολή ή μη της φόρμας):

```
public class AllowedCustomers
    {
        public int custId { get; set; }
        public String cust_url { get; set; }
        public String cust_token { get; set; }
        public String description { get; set; }
        public String success_url { get; set; }
        public String failure_url { get; set; }


    }
```

- **cust_url:**Το url για το domain του πελατη μου
- **cust_token:**Ένα μοναδικό token για κάθε πελάτη
- **description:** Περιγραφή πελάτη μόνο για την πλατφόρμα μου
- **success_url:**Η σελίδα που κάνεις Redirect τον πελάτη όταν υποβάλλει επιτυχώς τη φόρμα.
- **failure_url:** Η σελίδα που κάνεις Redirect τον πελάτη όταν ακυρώσει την υποβολή.

Το επόμενο model δεν έχει να κάνει με την Βάση Δεδομένων αλλά χρησιμοποιείται κατά την λειτουργία έκδοσης One Time Password πριν την προβολή της φόρμας από τον τελικό χρήστη.

```
public class UserOtpObj
    {
        public Totp OtpObject { get; set; }
        public String UrlRequest { get; set; }
        public DateTime datetimeComputedOtp { get; set; }

        public int MyCustomerUniqueId { get; set; }

    }
```

5.3 Υλοποίηση Βασικής Λογικής

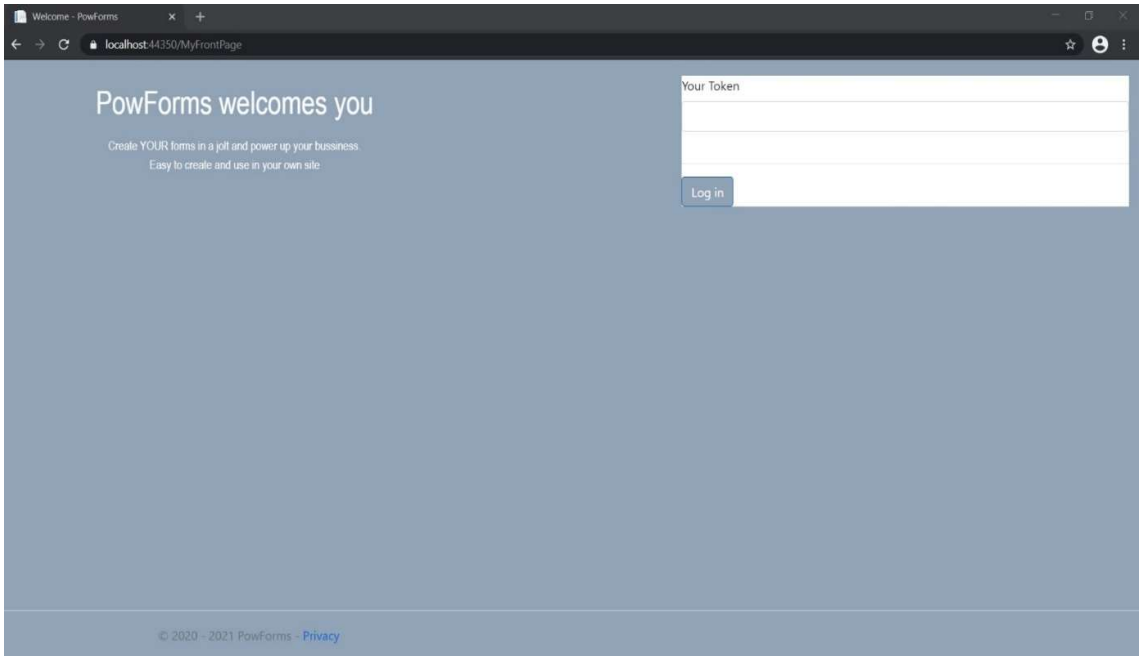
Ο controller που χρησιμοποιεί το PowForms εξυπηρετεί σαν ένα REST Web Service όπου στην GET επιστρέφει ένα μοναδικό OTP password. Ο OTP που παράγεται κρατείται σε μια λίστα με χρόνο, ώστε να

είναι εφικτή, μέσω του OTP η αναγνώριση του χρήστη καθώς και αν το OTP ισχύει ακόμα ή έχει λήξει και το πρόγραμμα δεν πρέπει να επιστρέψει την φόρμα.

```
[HttpGet("{key}", Name = "Get")]
public string Get(String key)
{
       return totpPass(Convert.ToString(key));
}

private String totpPass(String secKey)
        {
            byte[] passInBytes = System.Text.Encoding.Default.GetBytes(secKey);

            var totpObj = new Totp(passInBytes, mode: OtpHashMode.Sha512, step:
5, totpSize: 8);
            DateTime dtComputed = DateTime.Now;
            string totpCode = totpObj.ComputeTotp(dtComputed);
            addTo_Otplist(totpCode,totpObj,dtComputed,secKey);

            return totpCode;
        }
```

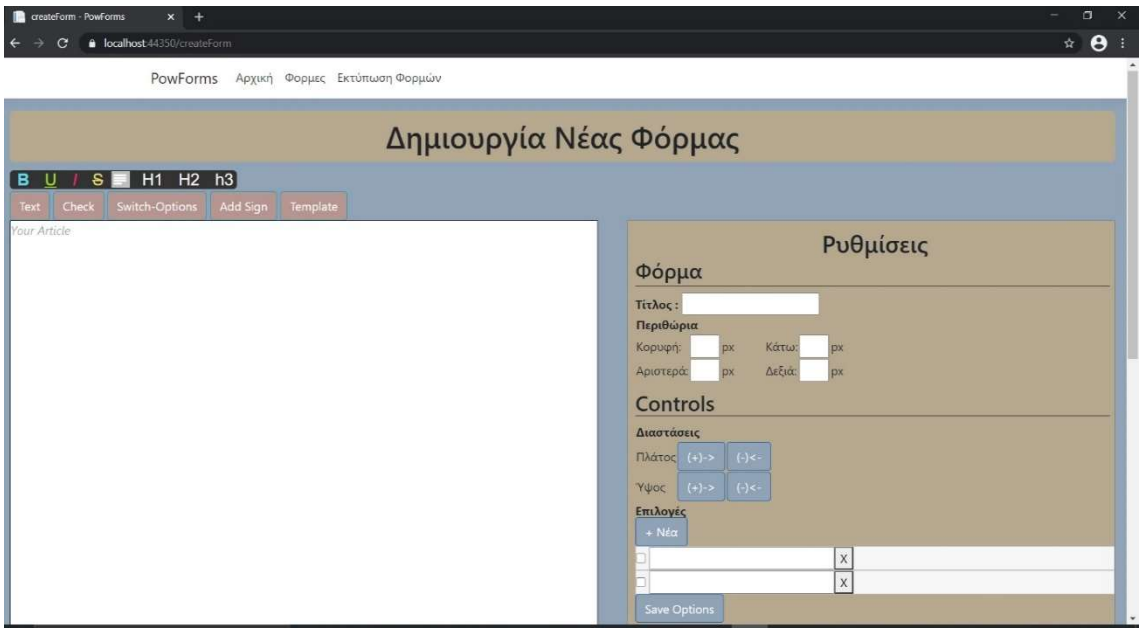
5.4 Περιγραφή Ροής

Ερχόμενος στην εφαρμογή, ο χρήστης καλείται να δώσει ένα μοναδικό κωδικό – Token που του έχει δοθεί για να κάνει LogIn.(Εικόνα 5.1)



Εικόνα 5.1 LogIn Page

Έπειτα από το επιτυχές Log In, ο χρήστης κατευθύνεται στην σελίδα όπου μπορεί να δημιουργήσει νέες φόρμες για τους πελάτες του όπως φαίνεται στην παρακάτω εικόνα (Εικόνα 5.2).

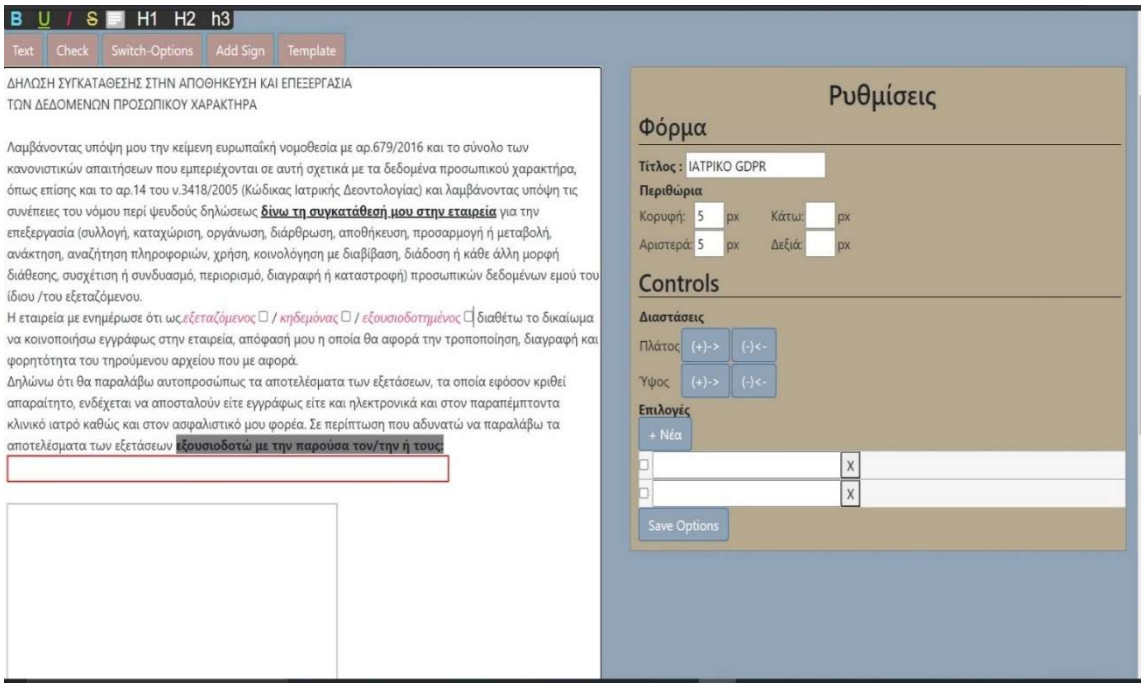


Εικόνα 5.2 Html Editor - Form Builder

Σε αυτόν τον Html Editor κατασκευάζει εύκολα, γρήγορα και ευέλικτα την φόρμα που απαιτούν οι ανάγκες του. Ο χρήστης γράφει ελεύθερο κείμενο και έχει την επιλογή να εισάγει κουτί εισαγωγής κειμένου, κουτί για τσέκ, λίστα επιλογών καθώς και κουτί εισαγωγής υπογραφής. Επίσης, η εφαρμογή προσφέρει την δυνατότητα εισαγωγής ολόκληρης φόρμας (Template) για ευκολία, σε συχνά επαναλαμβανόμενες φόρμες μεταξύ των χρηστών.

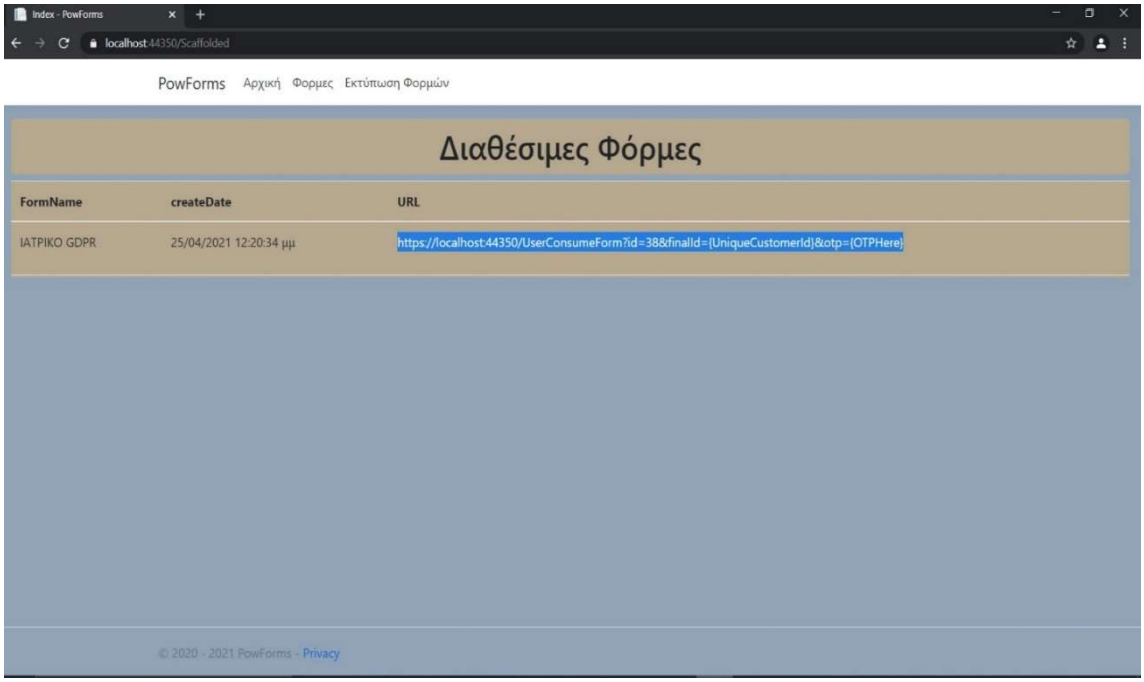
Στο δεξί κομμάτι της εικόνας 5.1 βλέπουμε τις ρυθμίσεις που μπορεί ο χρήστης να κάνει στην φόρμα, όπως Τίτλο και Περιθώρια καθώς και τις προσαρμογές που μπορεί να κάνει στα διάφορα controls (Αύξηση – Μείωση μεγέθους, Δημιουργία επιλογών για την εκάστοτε λίστα επιλογών)

Επίσης, ο χρήστης μπορεί στο ελεύθερο κείμενο να επηρεάσει το font κάνοντας Bold, Italics, Underline, να δώσει κεντρική στοίχιση καθώς και να μαρκάρει κείμενο ως επικεφαλίδα (html H1, H2 ή H3). Παράδειγμα δημιουργίας φόρμας βλέπουμε στην παρακάτω εικόνα. (Εικόνα 5.3)



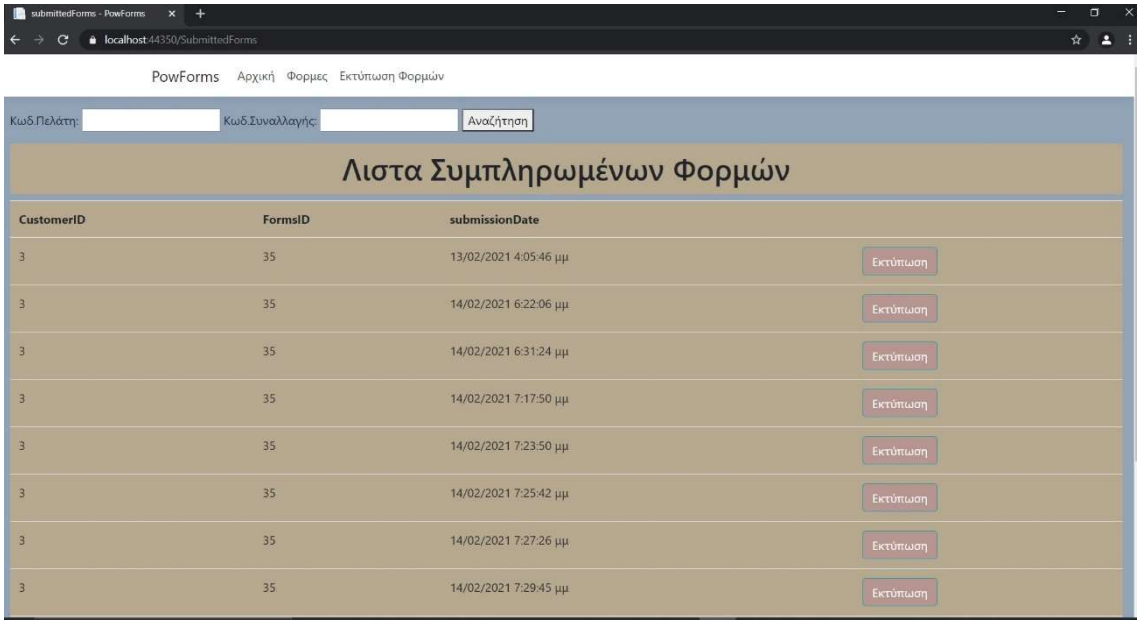
Εικόνα 5.3 Δημιουργία Φόρμας

Ο χρήστης, από την επιλογή Φόρμες του menu, βλέπει τις φόρμες που έχει δημιουργήσει καθώς και το URL που του δίδεται για να χρησιμοποιήσει την εκάστοτε φόρμα στην εφαρμογή του (Εικόνα 5.4)



Εικόνα 5.4 Προβολή Δημιουργημένων Φορμών

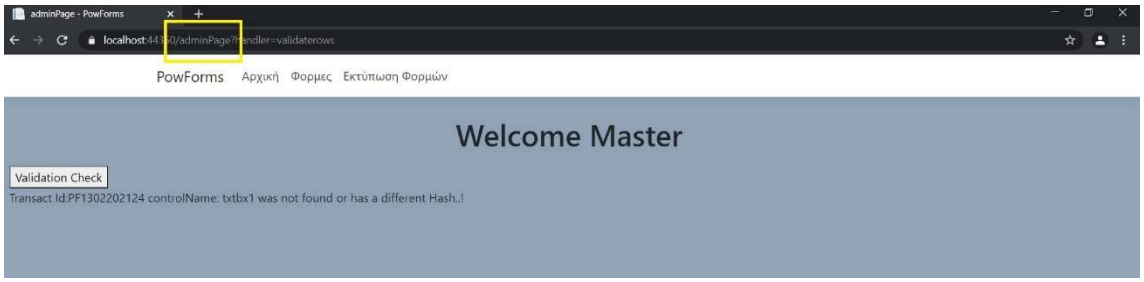
Από την επιλογή Εκτύπωση Φορμών, ο χρήστης βλέπει τις συμπληρωμένες φόρμες των πελατών του. Έχει την δυνατότητα αναζήτησης φόρμες πελάτη του ή συγκεκριμένη φόρμα βάση κωδικού συναλλαγής (Παράγεται κατά την επιτυχή υποβολή φόρμας και επιστρέφεται στον χρήστη κατά το Redirect στην ιστοσελίδα του) (Εικόνα 5.5).



Εικόνα 5.5 Συμπληρωμένες Φόρμες

Βλέπει την ημερομηνία υποβολής και πατώντας εκτύπωση, εκτυπώνει την φόρμα.

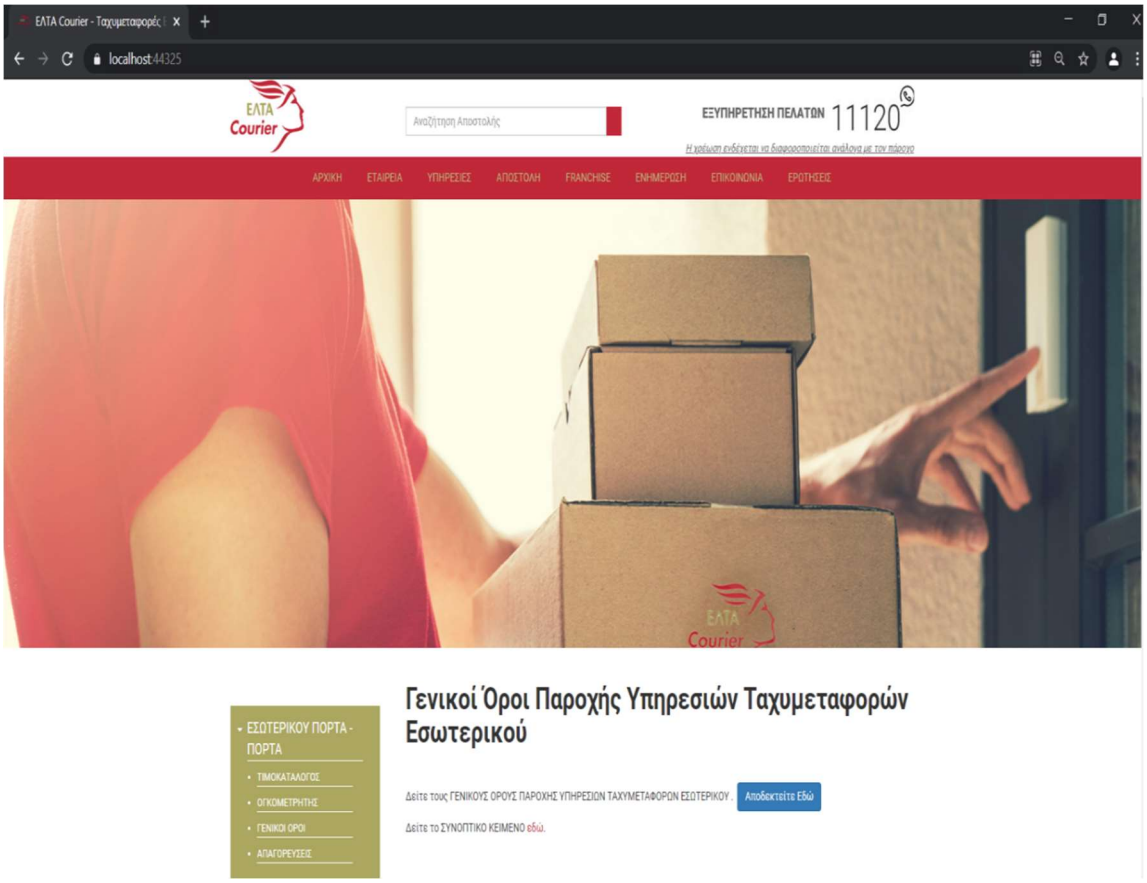
Τέλος, εξίσου σημαντική είναι η σελίδα του admin. Από εκεί ο χρήστης με το πάτημα ενός κουμπιού, ελέγχει αν όλα τα δεδομένα των συμπληρωμένων φορμών στη βάση του είναι ακέραια ή έχουν αλλαχθεί (Εικόνα 5.6).



Εικόνα 5.6 Admin Page - Validation Check

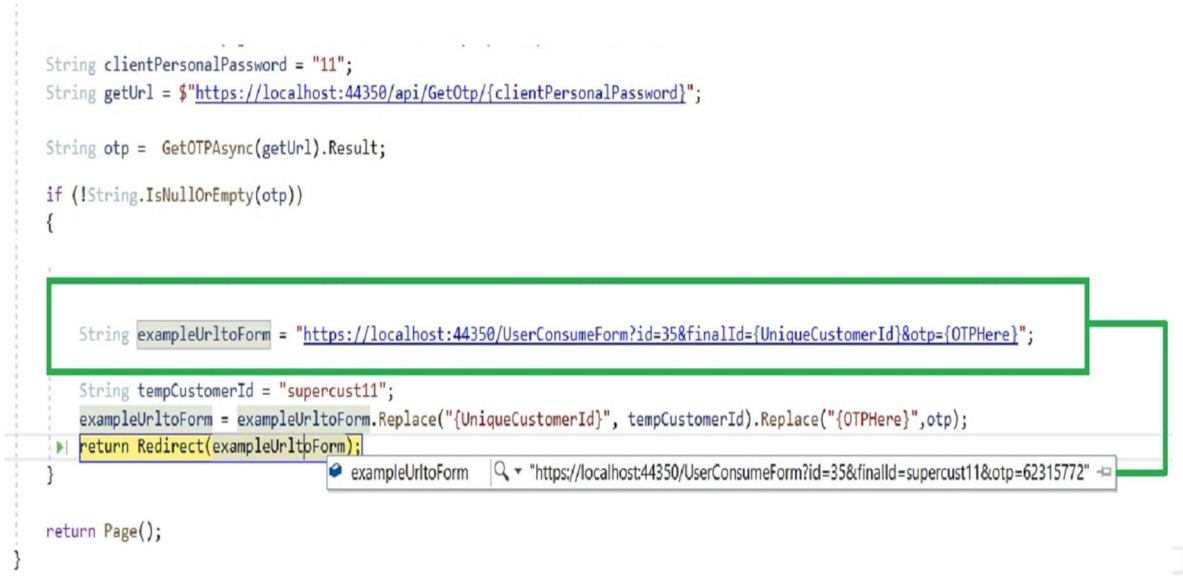
Κεφάλαιο 6. Τρόπος χρήσης της λύσης - Σενάρια

Έστω ότι πελάτης της εφαρμογής PowForms είναι τα ΕΛΤΑ. Ως πελάτης, έχει δημιουργήσει μια φόρμα για τις ανάγκες των χρηστών της. Η δημιουργία έχει γίνει ως άνω (Κεφάλαιο 5) και εξυπηρετεί την αποδοχή των *Γενικών όρων παροχής υπηρεσιών ταχυμεταφορών εσωτερικού*. Ένα Button, οδηγεί τον πελάτη των ΕΛΤΑ στην πλατφόρμα PowForms.



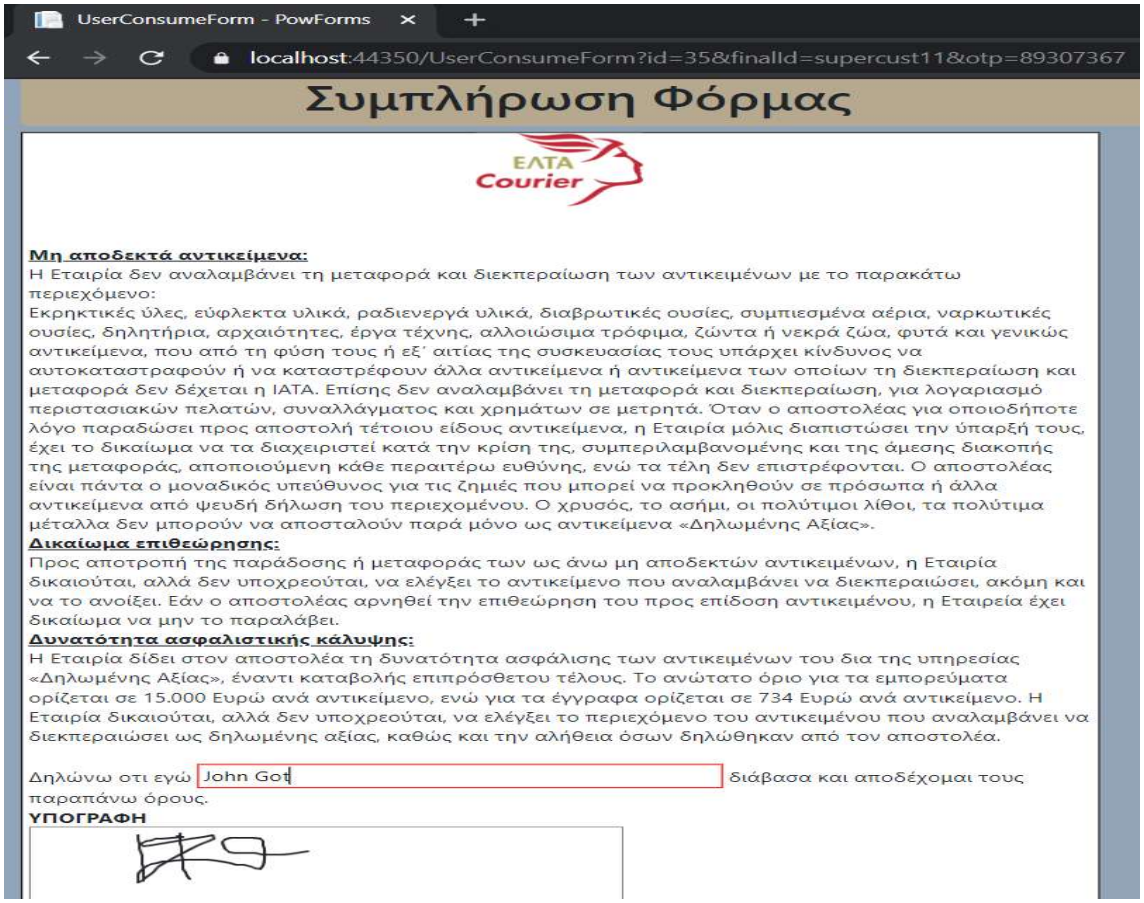
Εικόνα 6.1 Πελάτης ΕΛΤΑ

Οι λειτουργίες που επιτελούνται στο παρασκήνιο είναι οι εξής. Τα ΕΛΤΑ με GET Request παίρνουν ένα μοναδικό OTP κωδικό. Συμπληρώνουν με αυτόν το URL, καθώς και με τον μοναδικό κωδικό του πελάτη τους (Προϋποθέτει LogIn στο site του ΕΛΤΑ).(Εικόνα 6.2)

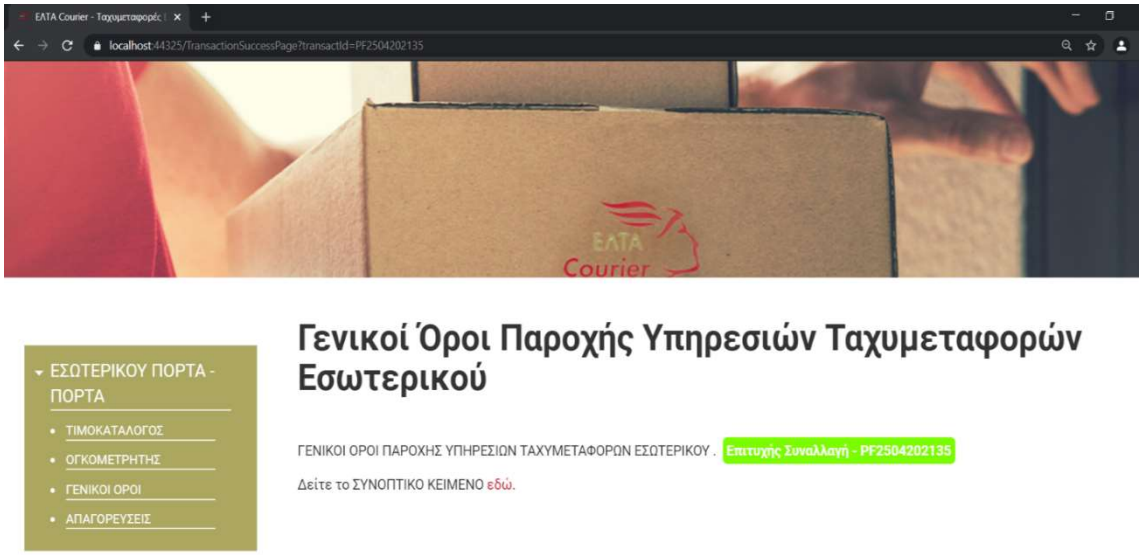


Εικόνα 6.2 Received Otp Password

Κατόπιν, κάνουν Redirect τον χρήστη τους στο URL. Εκεί, ο χρήστης βλέπει την φόρμα την συμπληρώνει (Εικόνα 6.3) και μεταφέρεται πίσω στο περιβάλλον των ΕΛΤΑ (Εικόνα 6.4).

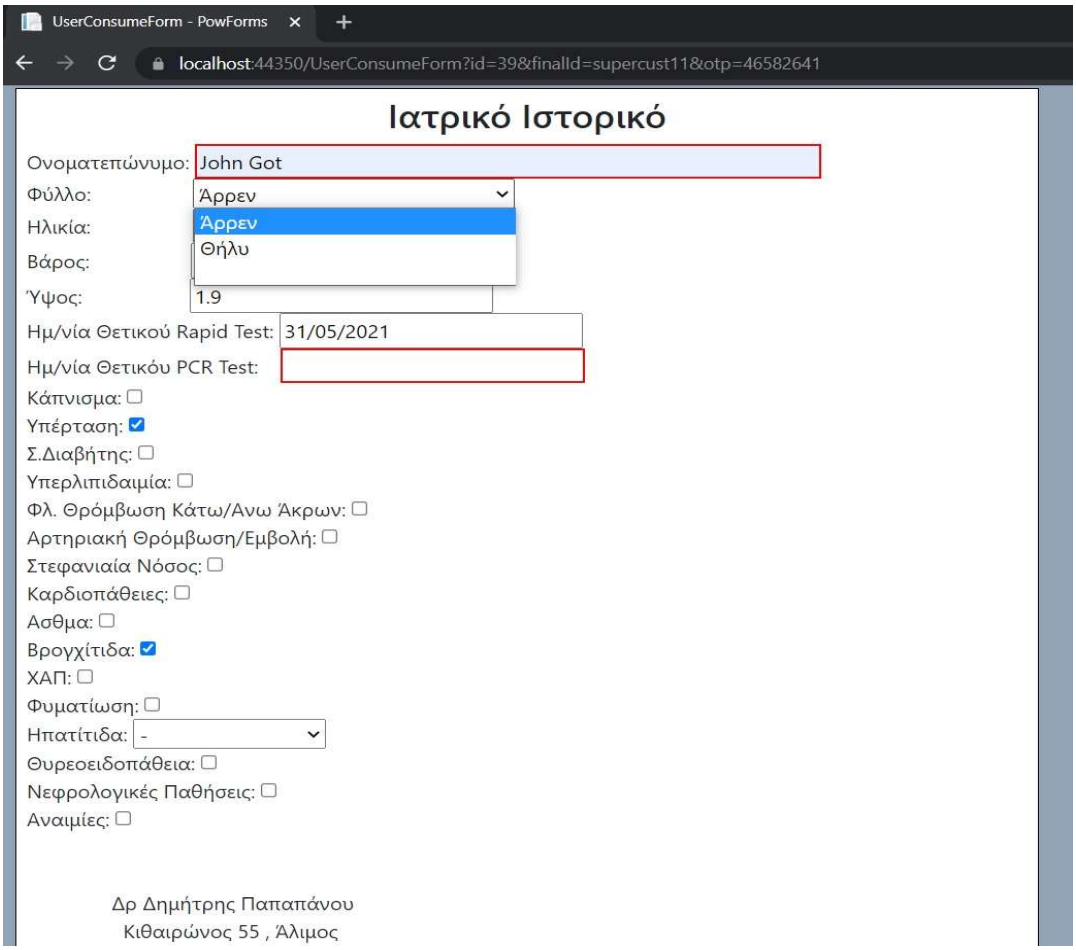


Εικόνα 6.3 υποβολή Φόρμας



Εικόνα 6.4 Επιτυχής Συναλλαγή

Ένα δεύτερο παράδειγμα χρήσης της πλατφόρμας είναι ένας ιδιώτης Ιατρός, που δημιουργεί ερωτηματολόγια – φόρμες για παρακολούθηση των ασθενών του. Η πλατφόρμα, θα μπορούσε να χρησιμοποιηθεί ως μέρος του προγράμματος που χρησιμοποιεί ο ίδιος για να κρατάει τα στοιχεία των ασθενών του. Ακολουθεί παράδειγμα παρακολούθησης ασθενή με Covid-19 σε φωτογραφία.



Εικόνα 6.5 Παράδειγμα Χρήσης Ιατρικής Φόρμας

Κεφάλαιο 7. Συμπεράσματα και μελλοντικές επεκτάσεις

7.1 Συμπεράσματα

Έπειτα από σύντομη έρευνα στο διαδίκτυο και ανάλυση στο κεφάλαιο 2, φαίνεται ότι οι υλοποιήσεις form builders με WYSIWYG Editor είναι σπάνιες. Οι μεγαλύτερες εταιρείες (Adobe form builder,Google Forms,Microsoft Forms κ.α.) επιλέγουν πιο απλές υλοποιήσεις, δηλαδή φόρμες με εισαγωγή controls το ένα κάτω από το άλλο ενώ οι υπόλοιπες υλοποιήσεις απαιτούν από τον χρήστη να έχει σαρωμένο έγγραφο (DocuSign,Panda Docs, Wacom).Λαμβάνοντας υπόψιν αυτό το δεδομένο, το υπάρχον project έχει μεγάλες δυνατότητες επέκτασης καθώς και εμπορευματοποίησης. Οι δυσκολίες που αντιμετωπίζονται είναι πολλές. Κυριότερες όμως είναι η επικοινωνία και χρήση της παρούσας υλοποίησης από τον καταναλωτή (παίρνει ένα url και το χρησιμοποιεί στο website του όμως θα πρέπει με κώδικα να το τροποποιήσει ώστε να εισάγει στοιχεία του πελάτη του, αναγκαία για την ταυτοποίηση και ανάκτηση στο PowForms) καθώς και τα ζητήματα ασφαλείας (αποθήκευση και επικύρωση δεδομένων, χρήση φόρμας σε σύντομο χρονικό διάστημα με τη χρήση Timed One Time Password - TOTP).Οι τεχνολογίες που χρησιμοποιήθηκαν είναι οι πιο καινούριες τη δεδομένη χρονική στιγμή, βοηθώντας τον προγραμματιστή να αποκτήσει νέες γνώσεις καθώς και δίδοντας στο σύστημα τις βέλτιστες επιδόσεις.

7.2 Μελλοντικές Επεκτάσεις

Η παραπάνω εφαρμογή θα μπορούσε να εξελιχθεί και να εμπορευματοποιηθεί υλοποιώντας αρχικά κάποιες περαιτέρω δυνατότητες. Κάποιες από αυτές αναφέρονται περιληπτικά παρακάτω :

- Να αναπτυχθούν παραπάνω δυνατότητες για τον Text Editor, όπως εισαγωγή εικόνας, πίνακα και άλλα.
- Έξυπνα πεδία με λειτουργικότητα και Format mask. Για παράδειγμα εισαγωγή μόνο αριθμού ή εισαγωγή password με αστερίσκους.
- Βελτιστοποίηση Login στην αρχική οθόνη. Χρήση Identity για ολοκληρωμένη διαχείριση χρηστών.
- Καλυτέρευση γραφικού περιβάλλοντος στο GUI
- Μορφοποίηση συμπληρωμένων πεδίων κατά την εκτύπωση ώστε να μη φαίνεται ότι είναι controls

7.3 Γνώσεις που αποκτήθηκαν

Η ανάπτυξη του παρόντος συστήματος απαιτούσε γνώσεις σε ευρύ φάσμα πεδίων όπως τεχνολογίες που χρησιμοποιούνται για το Frontend και το Backend καθώς επίσης και σοβαρό θεωρητικό μέρος για συναρτήσεις κατακερματισμού και One Time Password.Παρακάτω παρουσιάζονται σε μορφή λίστας:

Frontend

- Javascript και ο medium-editor
- Bootstrap και css

Backend

- C# με .Net core Razor Pages
- Entity Framework Core

Θεωρητικό μέρος

- OTP και διαχείριση κωδικών μιας χρήσης
- Hash Functions

Bibliography

[1] t. f. e. Wikipedia, «Representational state transfer,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Representational_state_transfer.

[2] t. f. e. Wikipedia, «Time-based One-Time Password,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Time-based_One-Time_Password.

[3] I. E. T. F. (IETF), «TOTP: Time-Based One-Time Password Algorithm,» [Ηλεκτρονικό]. Available: https://tools.ietf.org/html/rfc6238.

[4] GitHub, «Otp.NET C#,» [Ηλεκτρονικό]. Available: https://github.com/kspearrin/Otp.NET.

[5] Z. Khaishagi, «Cryptography: Explaining SHA-512,» Medium, [Ηλεκτρονικό]. Available: https://medium.com/@zaid960928/cryptography-explaining-sha-512-ad896365a0c1.

[6] Ε. Μ. Πανεπιστήμιο. [Ηλεκτρονικό]. Available: https://eclass.hmu.gr/modules/document/file.php/TP122/02.%CE%92%CE%B9%CE%B2%CE%BB%CE%AF%CE%BF%20%CE%98%CE%B5%CF%89%CF%81%CE%AF%CE%B1%CF%82/ch3-Asymmetrh-Kryptografhsh-Hashes.pdf.

[7] Wacom, «Wacom Website,» Wacom, [Ηλεκτρονικό]. Available: https://www.wacom.com/en-us.

[8] YouTube, «YouTube - Wacom Example,» [Ηλεκτρονικό]. Available: https://www.youtube.com/watch?v=Jun578JDLpU&ab_channel=WacomforBusiness.

[9] «WikiPedia - Google Forms,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Google_Forms.

[10] «Google Forms,» [Ηλεκτρονικό]. Available: https://www.google.com/forms/about/.

[11] «Introduction to Microsoft Forms,» [Ηλεκτρονικό]. Available: https://support.microsoft.com/en-us/office/introduction-to-microsoft-forms-bb1dd261-260f-49aa-9af0-d3dddcea6d69.

[12] «Wikipedia - Microsoft Forms,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Microsoft_Forms.

[13] «DocuSign,» [Ηλεκτρονικό]. Available: https://www.docusign.com/.

[14] «YouTube - DocuSign,» [Ηλεκτρονικό]. Available: https://www.youtube.com/watch?v=CZeSIbBlbIY&ab_channel=KnowledgeByMarcus.

[15] M. Brind, 21 Feb 2021. [Ηλεκτρονικό]. Available: https://www.learnrazorpages.com/.

[16] I. Landwerth, «From .NET Standard to .NET 5,» *Code Magazine,* τόμ. 17, αρ. 1, 2020.

[17] D. Arh, «What is .NET 5? (Overview, Features, Performance, Download),» 20 November 2020. [Ηλεκτρονικό]. Available: https://www.dotnetcurry.com/dotnetcore/dotnet-5-features-download.