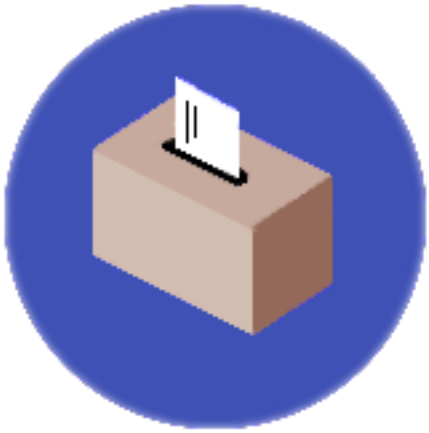
ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Εφαρμογή android για σύστημα e-voting



Του φοιτητή

Σάββα Σταματιάδη

Επιβλέπων καθηγητής

Κλεφτούρης Δημήτριος

Αρ. Μητρώου: 113797

Θεσσαλονίκη 2017

ΠΡΟΛΟΓΟΣ

Η παρούσα πτυχιακή εργασία έχει ως αντικείμενο την έρευνα της φύσης των ηλεκτρονικών ψηφοφοριών (e-voting) και την ανάπτυξη μιας πλατφόρμας ηλεκτρονικής ψηφοφορίας μαζί με μια εφαρμογή για κινητές συσκευές smartphones/tablets για το λειτουργικό σύστημα Android. Μετά από μελέτη και ανάλυση συστημάτων ηλεκτρονικής ψηφοφορίας καθώς και περιπτώσεων χρήσης αυτών, αναπτύχθηκε ένα σύστημα το οποίο επιτρέπει μια ασφαλή, ορθή και απλή διεξαγωγή ψηφοφοριών.

Πέρα από την ανάπτυξη του λογισμικού, σκοπός της εργασίας ήταν η μελέτη και η κατανόηση των προκλήσεων στον τομέα της ηλεκτρονικής ψηφοφορίας αλλά και η εξοικείωση με τις τεχνολογίες που χρησιμοποιήθηκαν.

Λέξεις κλειδιά: E-voting, Android, Play Framework, Restful API

ABSTRACT

This dissertation aims to investigate the nature of electronic voting (e-voting) and to develop an electronic voting platform along with a mobile application for the Android operating system. After studying and analyzing electronic voting systems and their use cases, a system has been developed which provides a way for safe, correct and simple voting.

Apart from the software development, the purpose of this effort was to study and understand the challenges posed in the field of electronic voting, as well as to become familiar with the technologies used.

Keywords: E-voting, Android, Play Framework, Restful API

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω την οικογένεια μου για την στήριξη που μου παρείχε όλα αυτά τα χρόνια καθώς και τον επιβλέπων καθηγητή μου Κλεφτούρη Δημήτριο για την καθοδήγηση στην συγγραφή της παρούσας πτυχιακής εργασίας.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

ΕΙΣΑΓΩΓΗ

Σκοπός της παρούσας πτυχιακής είναι η δημιουργία μια εφαρμογής e-voting για το λειτουργικό σύστημα Android η οποία θα παρέχει μια απλή, σωστή και ασφαλή διαδικασία εκλογών. Παράλληλα αναπτύχθηκε και μια πλατφόρμα ηλεκτρονικής ψηφοφορίας πάνω στην οποία θα στηρίζεται η εφαρμογή.

Για την κατανόηση των απαιτήσεων και του σχεδιασμού ενός τέτοιου συστήματος απαραίτητη προϋπόθεση αποτελεί και η μελέτη της βιβλιογραφίας που αφορά το e-voting. Η ανάλυση εστιάζει στα βασικά χαρακτηριστικά των συστημάτων όπως για παράδειγμα τα πλεονεκτήματα και τις προκλήσεις των ηλεκτρονικών εκλογών. Η αναλυτική περιγραφή του τεχνικού μέρους των συστημάτων που περιγράφονται στο πρώτο κεφάλαιο δεν αποτελεί αντικείμενο αυτής της έρευνας.

Βαθύτερος σκοπός ήταν η τριβή του εξεταζόμενου φοιτητή με τις νέες τεχνολογίες για την αντιμετώπιση του προβλήματος της ηλεκτρονικής ψηφοφορίας. Η πλατφόρμα και η εφαρμογή παρόλο που λειτουργούν ορθά, δεν αποτελούν σε καμία περίπτωση λύση για διεξαγωγή κρίσιμων εκλογών. Η συγκεκριμένη υλοποίηση αρμόζει για εκπαιδευτικούς σκοπούς καθώς αναλύονται οι διάφορες πτυχές της αρχιτεκτονικής εξυπηρετητή-πελάτη.

Στο πρώτο κεφάλαιο παρουσιάζονται τα βασικά χαρακτηριστικά του e-voting και γίνεται περεταίρω ανάλυση των προκλήσεων του χώρου.

Στα επόμενα δυο κεφάλαια αναλύεται η υλοποίηση της πλατφόρμας και της εφαρμογής παράλληλα με την επεξήγηση των τεχνολογικών λύσεων που χρησιμοποιήθηκαν.

Στο τελευταίο κεφάλαιο γίνεται η σύνοψη της πτυχιακής εργασίας και αναφέρονται κάποια συμπερασματικά σχόλια του φοιτητή σχετικά με τις ηλεκτρονικές ψηφοφορίες.

ΚΕΦΑΛΑΙΟ 1: Ηλεκτρονική ψηφοφορία E- voting

ΕΙΣΑΓΩΓΗ

Στο πρώτο κεφάλαιο γίνεται μια εισαγωγή στο θέμα της ηλεκτρονικής ψηφοφορίας (e-voting) αναφέροντας τα βασικά χαρακτηριστικά και αναλύοντας τις προκλήσεις του χώρου. Στόχος του κεφαλαίου είναι η πληροφόρηση του αναγνώστη όσον αφορά την φύση του e-voting για την ομαλότερη μετάβαση στο τεχνικό κομμάτι των επόμενων κεφαλαίων.

ΥΠΟΚΕΦΑΛΑΙΟ 1.1: Ορισμός

Ο όρος «ηλεκτρονική ψηφοφορία» αναφέρεται στην διεξαγωγή μιας εκλογής ή ψηφοφορίας με την χρήση ηλεκτρονικών μέσων με σκοπό την διευκόλυνση ή ακόμα και την εξ ολοκλήρου αντικατάσταση των διαδικασιών που εκτελούνται κατά την διάρκεια αυτής. Δύο είναι οι βασικές κατηγορίες που διακρίνονται:

- **Ηλεκτρονική ψηφοφορία (e-voting):** Στην πρώτη κατηγορία ανήκουν οι εκλογές στις οποίες χρησιμοποιούνται ηλεκτρονικές μηχανές και συστήματα τα οποία βρίσκονται στα εκλογικά κέντρα υπό την επίβλεψη κάποιας αρχής ή της επιλεγμένης εφορευτικής επιτροπής. Τα συστήματα που χρησιμοποιούνται μπορεί να περιέχουν έξυπνες κάρτες (smart cards), οπτικούς σαρωτές (optical scanners), συστήματα απευθείας άμεσης εγγραφής (DRE), οθόνες αφής, εκτυπωτές αποδείξεων κ.α.

- **Ηλεκτρονική ψηφοφορία εξ αποστάσεως (i-voting):** Στην δεύτερη κατηγορία κατατάσσονται οι εκλογές για τις οποίες χρησιμοποιούνται συστήματα βασισμένα στο διαδίκτυο και δεν είναι απαραίτητη η φυσική παρουσία του ψηφοφόρου σε κάποιο κέντρο. Τα συστήματα σε αυτή τη κατηγορία βασίζονται κατά κύριο λόγο στην αρχιτεκτονική πελάτη διακομιστή (client-server) και είναι εξοπλισμένα με κρυπτογραφικούς αλγορίθμους για την διασφάλιση των αρχών που θα παρουσιαστούν παρακάτω.

ΥΠΟΚΕΦΑΛΑΙΟ 1.2: Ιστορική αναδρομή

Το 1869 ο Thomas Edison έθεσε τα πρώτα θεμέλια κατοχυρώνοντας την ευρεσιτεχνία για μια μηχανή ηλεκτρονικής ψηφοφορίας, η οποία όμως δεν τράβηξε το ενδιαφέρον των επενδυτών. Στα τέλη του 19ου αιώνα γίνονται οι πρώτες κατοχυρώσεις ευρεσιτεχνιών για μηχανικά συστήματα ψηφοφορίας που χρησιμοποιούσαν μοχλούς τα οποία και έγιναν ευρέος διαδεδομένα μέχρι το 1930 με αποτέλεσμα να βρουν χρήση σε κάθε μεγάλη πόλη της Αμερικής. Τον Ιανουάριο του 1889 ο Herman Hollerith κατοχυρώνει την μέθοδο διάτρητων καρτών για την συλλογή στατιστικών δεδομένων η οποία θα αποτελέσει την βάση για τα συστήματα διάτρητων καρτών. Το 1962 έγινε η πρώτη χρήση συστημάτων οπτικών σαρωτών «mark-sense» στην California. Παράλληλα άρχισε να υιοθετείται η χρήση των συστημάτων διάτρητων καρτών όπως για παράδειγμα των Votomatic και Datavote (Evans and Paul 2004:24). Μετά το σκάνδαλο των προεδρικών εκλογών στην Florida το 2000 τα συστήματα διάτρητων καρτών άρχισαν να εγκαταλείπονται. Οι πρώτες μηχανές άμεσης ηλεκτρονικής ψηφοφορίας (DRE) χρησιμοποιήθηκαν σε πραγματικές εκλογές το 1975 στο Streamwood και Woodstock Illinois . Το 1990 εκδόθηκαν τα πρώτα προαιρετικά VSS (Voting System Standards) από το Federal Election Commission (FEC) Office of Elections Administration (OEA), που πλέον είναι ενσωματωμένη στην Election Assistance Commission (EAC), σύμφωνα με τα οποία στο εξής θα πιστοποιούνταν από τη National Association of State Election Directors (NASED) στα διάφορα από την ίδια εξουσιοδοτημένα ITAs (Independent Testing Authorities) τα εκάστοτε συστήματα ψηφοφορίας, τόσο ως προς το υλικό όσο και ως προς το λογισμικό. Τα συγκεκριμένα standards αναθεωρήθηκαν και έδωσαν τα 2002 VSS, που ισχύουν μέχρι σήμερα. Το Δεκέμβρη του 2005 η EAC ομόφωνα υιοθέτησε τα 2005 VSS, που αυξάνουν σημαντικά τις απαιτήσεις ασφάλειας και διευρύνουν την προσβασιμότητα, προσφέροντας περισσότερες δυνατότητες σε άτομα με ποικίλες «ανικανότητες». Θα τεθούν σε ισχύ το Δεκέμβρη του 2007, αντικαθιστώντας τα VSS 2002. Παράλληλα ξεκίνησε το 2001 και από την IEEE το project P1583, που αποτελεί το πρώτο standard για εξοπλισμό ψηφοφορίας που αναπτύχθηκε σε ανοιχτή συναινετική διαδικασία σύμφωνα με τους κανόνες ANSI (American National Standards Institute). Από τη στιγμή ωστόσο που δημιουργήθηκε η EAC από τη HAVA (Help America Vote Act) θα είναι αυτή υπεύθυνη για την έκδοση standards μέσω του National Institute of Standards and Technology (NIST) (Deutsch and Berger 2004:32-33). Η Βραζιλία το 2000 ήταν η πρώτη χώρα που χρησιμοποίησε σύστημα ψηφοφορίας πλήρως βασισμένο σε υπολογιστές. Η Georgia ήταν επίσης η πρώτη πολιτεία στις ΗΠΑ που υιοθέτησε εξολοκλήρου μια ενιαία DRE τεχνολογία για ψηφοφορία το 2002 (Neumann 2004:30). Σε σύγκριση με το 2000, οπότε χρησιμοποιήθηκαν διάφοροι τύποι ψηφοφορίας, το ποσοστό των undervotes (των ψηφοδελτίων όπου δεν είχαν συμπληρωθεί όλα τα απαιτούμενα στοιχεία εκλογής) μειώθηκε από 4,4% σε λιγότερο από 1%, κάτι που αποτελεί ποιοτικό δείκτη ακεραιότητας ενός εκλογικού συστήματος (Williams and

King 2004:39). Ηλεκτρονική ψηφοφορία εφαρμόστηκε επίσης στις δημοτικές εκλογές της Εσθονίας τον Οκτώβρη του 2005, όταν για πρώτη φορά το εκλογικό σώμα μιας ολόκληρης χώρας είχε τη δυνατότητα να ψηφίσει μέσω Internet. Εξαιτίας του γεγονότος ότι για πρώτη φορά εφαρμόστηκε κάτι τέτοιο και οι «ηλεκτρονικοί» ψηφοφόροι ήταν σχετικά λίγοι, δεν μπορούμε να εξάγουμε απολύτως ασφαλή συμπεράσματα. Ωστόσο η διαδικασία κρίθηκε σε γενικές γραμμές επιτυχής, γι' αυτό και χρησιμοποιήθηκε και στις βουλευτικές εκλογές του 2007 στην ίδια χώρα (Trechsel and Breuer 2006:40). Στις γενικές εκλογές της Αμερικής το 2006 χρησιμοποιήθηκαν DREs της εταιρίας Diebold σε 385 κομητείες οι οποίες αντιπροσώπευαν παραπάνω από το 10% των καταγεγραμμένων ψηφοφόρων. Τα συστήματα ήταν αρκετά ευπαθή, όπως δηλώνει ο Dr. Edward Felten, ο οποίος κατάφερε να φορτώσει κακόβουλο λογισμικό σε ένα δοκιμαστικό σύστημα σε λιγότερο από ένα λεπτό αποκτώντας την δυνατότητα να αλλάξει τα αποτελέσματα απαρατήρητος (Feldman et al 2006:2). Κατά καιρούς επιχειρήθηκε εφαρμογή τέτοιων συστημάτων, σε πιο περιορισμένη κλίμακα βέβαια, και σε άλλες χώρες όπως Αυστραλία, Βέλγιο, Βραζιλία, Καναδά, Γερμανία., Ιρλανδία, Ιταλία, Ολλανδία, Ινδία, Νορβηγία, Ρουμανία κι Ελβετία (Penha-Lopes 2005:412). Αξίζει σε αυτό το σημείο να αναφέρουμε, ότι οι ειδικοί προβλέπουν ότι τα συστήματα ηλεκτρονικής ψηφοφορίας μέσα στα επόμενα χρόνια θα εδραιωθούν σε μικρότερης δυναμικής εκλογές, όπως οι δημοτικές και οι σχολικές. Εκλογές τέτοιου βεληνεκούς παρουσιάζουν συνήθως τα μεγαλύτερα ποσοστά αποχής και είναι σχετικά «ελαστικές» από πλευράς ασφάλειας (συνήθως οι ίδιοι οι υποψήφιοι καταμετρούν τελικά τις ψήφους), ενώ επίσης στις περισσότερες περιπτώσεις γίνεται κατασπατάληση ανθρωπίνων και χρηματικών πόρων. Έτσι, τα συστήματα ηλεκτρονικής ψηφοφορίας εισάγονται σταδιακά, με στόχο τη μείωση της αποχής και του κόστους, με παράλληλη αύξηση της ασφάλειας.

ΥΠΟΚΕΦΑΛΑΙΟ 1.3: E-voting στην Ευρώπη

Η **Αλβανία** εργάστηκε σε δύο πιλοτικά έργα. Το πρώτο αφορούσε την εισαγωγή ηλεκτρονικών μέσων αναγνώρισης των ψηφοφόρων στα εκλογικά κέντρα (με την χρήση της εθνικής ταυτότητας) ενώ το δεύτερο σχετιζόταν με οπτικούς σαρωτές σε δύο περιφερειακά κέντρα καταμέτρησης κατά την διάρκεια των εκλογών του Ιουνίου το 2013. Και τα δύο έργα απέτυχαν.

Στην **Αρμενία**, η Κεντρική Εκλογική Επιτροπή επινόησε ένα (αρκετά απλό) σύστημα, επιτρέποντας στους Αρμένιους που εργάζονταν σε διπλωματικές αποστολές στο εξωτερικό και σε Αρμένιους επαγγελματίες που δούλευαν στο εξωτερικό για Αρμένικες εταιρίες να ψηφίσουν μέσω διαδικτύου. Το νομικό πλαίσιο ψηφίστηκε πριν από τις κοινοβουλευτικές εκλογές του 2012 αλλά το ποσοστό συμμετοχής ήταν μικρό.

Στην **Αυστρία**, μόνο η ψηφοφορία μέσω του διαδικτύου έχει συζητηθεί σοβαρά. Το αυστριακό ομοσπονδιακό Υπουργείο Εσωτερικών πραγματοποίησε μια διακυβερνητική μελέτη σκοπιμότητας που παρουσιάστηκε στα τέλη του 2004. Προκειμένου να εφαρμοστεί η ψηφοφορία μέσω του διαδικτύου, μια τροποποίηση στο ομοσπονδιακό σύνταγμα (πλειοψηφία δύο τρίτων στο κοινοβούλιο) ήταν απαραίτητη. Ορισμένες μη δεσμευτικές ακαδημαϊκές δοκιμές το 2003, 2004 και 2006 και μια νομικά δεσμευτική χρήση κατά την διάρκεια των εκλογών της αυστριακής ομοσπονδίας φοιτητών το 2009 ήταν οι μόνες αξιοσημείωτες εμπειρίες. Το 2011 το αυστριακό συνταγματικό δικαστήριο ανέστειλε ορισμένες διατάξεις του κανονισμού για τις φοιτητικές εκλογές του 2009. Την ίδια στιγμή, το συνταγματικό δικαστήριο τόνισε ότι σε όλες τις μελλοντικές εξελίξεις των ηλεκτρονικών ψηφοφοριών θα πρέπει να καθορίζεται ξεκάθαρα η νομοθεσία προκειμένου να εξασφαλίζεται η διαφάνεια τόσο για τις εκλογικές επιτροπές όσο και για τους ψηφοφόρους.

Το **Αζερμπαϊτζάν** στο παρελθόν, έτρεξε μερικούς μη δεσμευτικούς πιλότους του διαδικτυακών εκλογών (shadow elections) αλλά από τότε δεν έγιναν περαιτέρω βήματα για το e-voting.

Το **Βέλγιο** έπαψε να χρησιμοποιεί μηχανές ψηφοφορίας μετά τις συζητήσεις περί του θέματος στην Ολλανδία, αλλά πρόσφατα εξέτασε ένα νέο μηχανικό σύστημα εκλογών βασισμένο στο χαρτί (paper-based) το οποίο δοκιμάστηκε στις περιφερειακές εκλογές τον Οκτώβριο του 2012 και κατέδειξαν την ανάγκη για διάφορες τροποποιήσεις. Το βελτιωμένο σύστημα θα χρησιμοποιούνταν στις εκλογές του 2014 στην μισή χώρα. Η ψηφοφορία μέσω του διαδικτύου θα ληφθεί υπόψιν μόνο για Βέλγους ψηφοφόρους στο εξωτερικό.

Η **Βουλγαρία** άρχισε να συζητά λύσεις ηλεκτρονικής ψηφοφορίας τόσο για εκλογικά κέντρα όσο και μέσω του διαδικτύου το 2004. Ένα νομικό προσχέδιο επέτρεψε τις δοκιμές διαδικτυακών ψηφοφοριών. Το 2009, διεξήχθη μια δοκιμασία σε εννέα εκλογικές περιφέρειες. Μια νομική τροποποίηση σχετικά με την άδεια των ηλεκτρονικών ψηφοφοριών ψηφίστηκε το 2012, αλλά στη συνέχεια ανατράπηκε από το συνταγματικό δικαστήριο. Ο τότε τρέχων εκλογικός κώδικας όριζε την εισαγωγή μηχανικής ψηφοφορίας το 2015.

Η **Εσθονία** ήταν η πρώτη χώρα που εισήγαγε την ψηφοφορία μέσω του διαδικτύου ως νομικά δεσμευτικό κανάλι κατά την διάρκεια των δημοτικών εκλογών του 2005 και των κοινοβουλευτικών εκλογών του 2007. Οι ψήφοι μέσω του διαδικτύου πρέπει να εκτελούνται πριν από την ημέρα των εκλογών. Κατά τις δημοτικές εκλογές του 2013, το 24,3% των ψήφων ήρθε μέσω του διαδικτύου. Το σύστημα και η διαδικασία διαρκώς βελτιώνονται, για παράδειγμα με την εγκατάσταση μιας επιτροπής ηλεκτρονικής ψηφοφορίας αποτελούμενης από επαγγελματίες πληροφορικής που είναι υπεύθυνοι για τη διεξαγωγή της διεργασίας. Θα υπάρξει μεγαλύτερη διαφάνεια με την εισαγωγή ενός νέου

συστήματος επαλήθευσης το οποίο δοκιμάστηκε το 2013 και θα αποτελέσει αναπόσπαστο μέρος του νόμου το 2015.

Η **Φινλανδία** δοκίμασε μηχανές ψηφοφορίας με σύνδεση στο διαδίκτυο σε εκλογικά κέντρα σε τρείς δήμους το 2008. Μετά από κάποιες αδυναμίες και δικαστικές αποφάσεις το έργο σταμάτησε. Μια ομάδα εργασίας εξέτασε τις δυνατότητες της ψηφοφορίας στο διαδίκτυο και παρουσίασε μια έκθεση τον Ιούνιο του 2014. Προτάθηκε περαιτέρω έρευνα σχετικά με τη χρήση του διαδικτύου για συμμετοχικά μέσα.

Η **Γαλλία** χρησιμοποιεί μηχανές ηλεκτρονικής ψηφοφορίας σε ορισμένους δήμους, παρόλα αυτά ο αριθμός δεν θα αυξηθεί μετά τις συζητήσεις στην Ολλανδία και τη Γερμανία. Από τις αρχές του 2000, η διαδικτυακή ψηφοφορία για τους πολίτες εξωτερικού έχει συζητηθεί και πραγματοποιήθηκαν ορισμένες δοκιμασίες. Το 2012, οι εκπρόσωποι των Γάλλων που διαμένουν στον εξωτερικό εκλέχθηκαν μέσω του διαδικτύου για πρώτη φορά.

Η **Γερμανία** χρησιμοποιούσε μηχανές ψηφοφορίας σε ορισμένες εκλογικές περιφέρειες (για πάσης φύσης εκλογές) από τη δεκαετία του 1960. Λόγω καταγγελιών σχετικά με τις κοινοβουλευτικές εκλογές του 2005, το ομοσπονδιακό συνταγματικό δικαστήριο της Γερμανίας αποφάσισε στις 3 Μαρτίου του 2009 ότι η χρήση μηχανών υπονόμευσε την αρχή της δημοσιότητας. Ενώ οι ηλεκτρονικές μηχανές ψηφοφορίας με σύστημα αποκόμματος ελέγχου θα αρκούσαν για τις απαιτήσεις της απόφασης, η Γερμανία σταμάτησε να χρησιμοποιεί όλα τα είδη μηχανών. Η ψηφοφορία μέσω του διαδικτύου ασκείται σε πολύ μικρή κλίμακα σε ακαδημαϊκό και ημι-ιδιωτικό περιβάλλον, αλλά όχι σε πολιτικές εκλογές.

Η **Ιρλανδία** εισήγαγε μηχανές ηλεκτρονικής ψηφοφορίας το 2004, αλλά ποτέ δεν τις χρησιμοποίησε λόγω ανησυχιών του κοινού σχετικά με την αξιοπιστία τους. Οι μηχανές αποθηκεύτηκαν για χρόνια και τελικά κατεδαφίστηκαν το 2012.

Η **Λετονία** επικεντρώνεται στη χρήση του ΙΤC στη σάρωση και στην καταμέτρηση των ψηφοδελτίων. Εκτός από τους οπτικούς σαρωτές, οι ιδέες για την ψηφοφορία στο διαδίκτυο συζητιούνται έχοντας πρότυπο την γειτονική χώρα (Εσθονία).

Το **Λιχτενστάιν** έχει τη νομική βάση για την ηλεκτρονική ψηφοφορία στις δημοτικές εκλογές που είναι επηρεασμένη από τις εξελίξεις στην Ελβετία και έχει ακολουθήσει συζητήσεις περί του e-voting επί σειρά ετών χωρίς όμως περαιτέρω βήματα.

Η **Λιθουανία** προσπάθησε επανειλημμένα να ακολουθήσει το παράδειγμα της Εσθονίας, αλλά οι προτάσεις της κεντρικής εκλογικής επιτροπής για την εισαγωγή ηλεκτρονικής ψηφοφορίας δεν έχουν ακόμη λάβει επαρκή υποστήριξη στο κοινοβούλιο. Η Ολλανδία διέθετε μηχανολογικά και ηλεκτρονικά μηχανήματα ψηφοφορίας που χρονολογούνταν από τη δεκαετία του 1960 και χρησιμοποίησαν επίσης ψηφοφορία μέσω διαδικτύου για ορισμένα τμήματα. Μετά από αμφιβολίες

σχετικά με την ασφάλεια των μηχανών ψηφοφορίας που εκδηλώθηκαν δημοσίως από ένα μη κυβερνητικό οργανισμό, οι μηχανές ψηφοφορίας και η ψηφοφορία μέσω του διαδικτύου σταμάτησαν το 2008 με υπουργικό διάταγμα. Στα τέλη του 2013, μια επιτροπή μελέτης συνέστησε την εισαγωγή ηλεκτρονικής ψηφοφορίας και μέτρησης προκειμένου να καταστεί η διαδικασία ψηφοφορίας και καταμέτρησης πιο προσιτή και ταχύτερη. Οι εκλογικοί σταθμοί πρέπει να χρησιμοποιούν νέα μηχανήματα με εκτυπωτές ψηφοφορίας. Μια εθνική εξέλιξη θα μπορούσε να πραγματοποιηθεί μετά από μια φάση πειραματισμού κοντά στο 2018-2019.

Η **Νορβηγία** διενήργησε μελέτη σκοπιμότητας σχετικά με την ψηφοφορία μέσω του διαδικτύου το 2006 και πραγματοποίησε το πρώτο πιλοτικό πρόγραμμα σε τοπικό επίπεδο (10 δήμους και 4.5% του πληθυσμού) το 2011. Τα διδάγματα που αντλήθηκαν από άλλες περιπτώσεις όπως για παράδειγμα η ανάγκη καθολικής επαληθεύσεως, λήφθηκαν υπόψιν. Μια άλλη χρήση της ψηφοφορίας μέσω του διαδικτύου πραγματοποιήθηκε κατά τις κοινοβουλευτικές εκλογές του 2013 (12 δήμοι και 7% πληθυσμού). Τον Ιούνιο του 2014, η κυβέρνηση ανακοίνωσε να διακόψει τη χρήση των δοκιμασίων για το e-voting.

Στη **Ρωσία**, η κεντρική εκλογική επιτροπή εισήγαγε ηλεκτρονικές μηχανές ψηφοφορίας με απόκομμα ελέγχου το 2005. Τον Φεβρουάριο του 2013, η συνταγματική επιτροπή πρότεινε να εξετάσει και την ψηφοφορία μέσω του διαδικτύου.

Στη **Σλοβενία**, τα ηλεκτρονικά μηχανήματα ψηφοφορίας έχουν χρησιμοποιηθεί στα εκλογικά κέντρα προκειμένου να βοηθηθούν οι ψηφοφόροι με αναπηρία, αν και δεν φαίνεται να εξετάζεται περαιτέρω επέκταση.

Στην **Ισπανία**, οι δοκιμασίες σχετικά με τις μηχανές ηλεκτρονικής ψηφοφορίας διεξάγονται από το 1995. Επιπροσθέτως, ορισμένες δοκιμές μέσω διαδικτύου πραγματοποιήθηκαν σε περιφερειακό (2003, Καταλονία) και σε εθνικό επίπεδο (2005). Η βάση για την ψηφοφορία μέσω του διαδικτύου καθορίστηκε στον εκλογικό κώδικα της χώρας των Βάσκων το 1998. Τον τελευταίο καιρό δεν έχουν πραγματοποιηθεί άλλες σοβαρές συζητήσεις.

Η **Ελβετία** είχε τις πρώτες συζητήσεις της σχετικά με την ψηφοφορία μέσω του διαδικτύου το 1998 και άρχισε σε τρία καντόνια το 2002 ένα πιλοτικό πρόγραμμα ηλεκτρονικής ψηφοφορίας. Στην αρχή χρησιμοποιήθηκε μόνο στις δημοτικές εκλογές και τα δημοψηφίσματα. Το 2011, πραγματοποιήθηκε η πρώτη εθνική χρήση (για εθνικές κοινοβουλευτικές εκλογές). Η κυβέρνηση εξακολουθεί να βρίσκεται σε διαδικασία σταδιακής επέκτασης της χρήσης της ηλεκτρονικής ψηφοφορίας. Νέα νομικά χαρακτηριστικά για το ομοσπονδιακό επίπεδο εγκρίθηκαν τον Δεκέμβριο του 2013. Προκειμένου να επεκταθεί περαιτέρω η ψηφοφορία μέσω του διαδικτύου, θα απαιτηθεί ένα νέο μοντέλο επαλήθευσης και νέες ρουτίνες ελέγχου. Μέχρι το τέλος του 2013, 12 καντόνια χρησιμοποίησαν την ηλεκτρονική ψηφοφορία με τον ένα ή τον άλλον τρόπο.

Το **Ηνωμένο Βασίλειο** ασχολήθηκε πολύ με τη δοκιμή όλων των ειδών μεθόδων ηλεκτρονικής ψηφοφορίας στις αρχές της δεκαετίας του 2000. Οι δοκιμές σε διάφορες εκλογικές περιφέρειες μεταξύ 2002 και 2007 αφορούσαν την ψηφοφορία ψηφοδελτίων, την ψηφοφορία σε κιόσκι και στην ψηφοφορία στο διαδίκτυο. Μετά από αρνητικές εμπειρίες σε άλλες χώρες και κρίσιμες φωνές από το την εκλογική επιτροπή του Ηνωμένου Βασιλείου, η κυβέρνηση δεν έχει εξετάσει περαιτέρω τις ευκαιρίες της ηλεκτρονικής ψηφοφορίας. Τον Μάρτιο του 2014, ο πρόεδρος της βρετανικής εκλογικής επιτροπής ζήτησε τον εκσυγχρονισμό των εκλογών και την μετακίνηση στην ηλεκτρονική ψηφοφορία.

ΥΠΟΚΕΦΑΛΑΙΟ 1.4: Βασικές αρχές

Για να θεωρηθεί ένα σύστημα ηλεκτρονικής ψηφοφορίας ικανοποιητικό θα πρέπει να καλύπτει κάποιες αρχές και προϋποθέσεις. Οι περισσότερες από αυτές τις αρχές δεν ισχύουν μόνο για τα ηλεκτρονικά συστήματα αλλά εφαρμόζονται για κάθε μέθοδο διεκπεραίωσης εκλογών. Παρακάτω αναφέρονται οι πιο σημαντικές αρχές-ιδιότητες:

1. **Ακεραιότητα:** Η ιδιότητα που εγγυάται ότι τα δεδομένα που καταγράφονται από το σύστημα είναι σωστά και δεν πρόκειται να υποστούν οποιαδήποτε μεταβολή ή καταστροφή από την εκκίνηση των εκλογών. Η ακεραιότητα συνδέεται άμεσα με τις αρχές της εμπιστευτικότητας, της ευρωστίας και της αξιοπιστίας.

2. **Εμπιστευτικότητα:** Η αρχή της εμπιστευτικότητας έχει ως βάση την μυστικότητα και την ανωνυμία. Ένα σύστημα το οποίο εγγυάται τις δύο αυτές ιδιότητες μπορεί να εγγυηθεί και την εμπιστευτικότητα του.

3. **Μυστικότητα:** Η μυστικότητα απαιτεί για κάθε ψηφοφόρο που συμμετέχει στην διαδικασία της ψηφοφορίας να μην μπορεί να γίνει γνωστή η ψήφος του. Η μυστικότητα εξετάζεται από την σκοπιά του ψηφοφόρου. (Ryan et al 2009:664).

4. **Ανωνυμία:** Η ανωνυμία απαιτεί για κάθε ψήφο να μην μπορεί να γίνει γνωστός ο ψηφοφόρος της. Σε αντίθεση με την μυστικότητα, η ανωνυμία όπως φαίνεται, εξετάζεται από την σκοπιά της ψήφου. Και οι δυο αρχές έχουν ως προϋπόθεση να μην υπάρχει κάποιος εξωτερικός σύνδεσμος ο οποίος να ενώνει την ταυτότητα του ψηφοφόρου με την ψήφο του (Ryan et al 2009:664).

5. **Αξιοπιστία:** Ένα ηλεκτρονικό σύστημα ψηφοφορίας θεωρείται αξιόπιστο όταν λειτουργεί με βάση των προβλεπόμενων προϋποθέσεων.

Προβλέπεται ότι το σύστημα θα παρέχει ακρίβεια των αποτελεσμάτων του, διαφάνεια, ευρωστία, δημοκρατία και ισότητα κατά την διάρκεια των εκλογών (Osho et al 2015:204).

6. **Ατομική Επαληθευσιμότητα:** Κάθε ψηφοφόρος μπορεί να ελέγξει ότι η ψήφος του έχει καταγραφεί και μετρηθεί σωστά (Clarkson et al 2008:355).

7. **Οικουμενική Επαληθευσιμότητα:** Οποιοσδήποτε μπορεί να ελέγξει ότι οι ο συμψηφισμός των αποτελεσμάτων έχει γίνει σωστά και εμπεριέχει ακέραιους ψήφους από άτομα τα οποία έχουν δικαίωμα ψήφου (Clarkson et al 2008:355).

8. **Αντίσταση εξαναγκασμού:** Η αντίσταση εξαναγκασμού εγγυάται ότι ένας ψηφοφόρος δεν μπορεί να αποδείξει σε ένα τρίτο πρόσωπο την ψήφο του ακόμα και στην περίπτωση που αυτός συνεργάζεται λόγω εξαναγκασμού ή πώληση της ψήφου (Delaune et al 2006:12). Η ικανοποίηση της συγκεκριμένης ιδιότητας αποτελεί μια από τις δυσκολότερες προκλήσεις στον χώρο της ηλεκτρονικής ψηφοφορίας.

9. **Έλλειψη αποδεικτικότητας:** Ένας ψηφοφόρος δεν πρέπει να είναι σε θέση να αποδείξει σε ένα τρίτο πρόσωπο για το τι ψήφισε, δηλαδή δεν μπορεί να έχει ή να παράγει κάποια απόδειξη π.χ. (απόκομμα εκτυπωτή, φωτογραφικό υλικό κ.λπ.) των επιλογών της ψήφου του. Ένα κρυπτογραφικό απόκομμα συνιστά αποδεικτικό ψήφου αλλά όχι του περιεχομένου της. (Ryan et al 2009:664).

10. **Ευρωστία-Ανθεκτικότητα-Διαθεσιμότητα:** Ένα σύστημα πρέπει να εξυπηρετεί τον σκοπό του και να είναι διαθέσιμο όποτε το χρειαστεί ο χρήστης. Αυτό σημαίνει ότι το σύστημα θα πρέπει να είναι ανθεκτικό ενάντια σε επιθέσεις δικτύου όπως για παράδειγμα επιθέσεις άρνησης εξυπηρέτησης (denial of service), ανακατεύθυνση δικτυακής κίνησης (traffic redirection), πλημμύρα συνδέσεων (connection flooding), επιθέσεις βασισμένες στο υλικό του συστήματος και επιθέσεις εμπλοκής σήματος (jamming attack) (Osho et al 2015:204). Το σύστημα πρέπει να έχει μηχανισμούς ασφαλείας και να μπορεί να ανακάμπτει από σφάλματα και επιθέσεις.

11. **Ευχρηστία:** Το σύστημα θα πρέπει να είναι εύκολο στην χρήση και να μην προϋποθέτει γνώσεις υπολογιστή έτσι ώστε να συμμετέχουν άνθρωποι από όλες τις κοινωνικές ομάδες (Osho et al 2015:204).

12. **Προσιτότητα:** Είναι απαραίτητο, ένα σύστημα να δίνει με κάποιον τρόπο την δυνατότητα ψήφου σε άτομα τα οποία έχουν ειδικές ικανότητες. Σύμφωνα με το νομοθετικό πλαίσιο της εκάστοτε χώρας. Στις περισσότερες,

αν όχι όλες, δημοκρατικές χώρες υπάρχουν νόμοι που προστατεύουν τις ευπαθείς κοινωνικές ομάδες και υποχρεώνουν την εξασφάλιση τρόπων συμμετοχής σε εκλογές.

13. **Δημοκρατία:** Κάθε ψηφοφόρος έχει ισάξια δικαιώματα ως προς την διαδικασία της ψηφοφορίας και η ψήφος του καθενός έχει ισάξιο βάρος.

14. **Ευελιξία:** Ένα ευέλικτο σύστημα είναι ικανό να δουλέψει με διάφορους τρόπους εκλογών όπως για παράδειγμα μέθοδοι πλειοψηφίας, αναλογικής αντιπροσώπευσης κ.α. (Ryan et al 2009:665).

ΥΠΟΚΕΦΑΛΑΙΟ 1.5: Συστήματα ηλεκτρονικής ψηφοφορίας

Παρακάτω παρουσιάζουμε με λίγα λόγια κάποια συστήματα που έχουν χρησιμοποιηθεί ως τώρα για ηλεκτρονικές εκλογές:

- **Συστήματα διάτρητης κάρτας (punch-card systems):** Στο μοντέλο διάτρητης κάρτας το δελτίο αποτελείται από μια καρτέλα όπου ο ψηφοφόρος κάνει διάτρηση με την βοήθεια κάποιου εργαλείου δίπλα στον υποψήφιο της επιλογής του. Το δελτίο αυτό περνάει από μηχάνημα το οποίο σαρώνει την ψήφο και αναλαμβάνει την καταμέτρηση των ψήφων. Γνωστά συστήματα αυτής της κατηγορίας είναι το Votomatic βασισμένη στην τεχνολογία Port-A-Punch της IBM και το Datavote.

- **Συστήματα οπτικού σαρωτή (optical scan systems):** Τα συστήματα αυτά χρησιμοποιούν οπτικό σαρωτή για να διαβάσουν και να μετρήσουν τα σημειωμένα ψηφοδέλτια. Ψηφιακά στυλό, ηλεκτρονικοί μαρκαδόροι ψηφοφορίας και ηλεκτρο-γραφικά συστήματα MarkSense είναι μερικά από τα συστήματα που ανήκουν σε αυτή την κατηγορία και μπορούν όχι μόνο να διαβάσουν αλλά και να σημειώσουν ψηφοδέλτια. Αυτά τα συστήματα κρατούν ένα απτό ψηφοδέλτιο ως απόδειξη αλλά η καταμέτρηση γίνεται πολύ γρήγορα. Το πλεονέκτημα αυτών είναι ότι σε περίπτωση υποψίας λάθους ή επίθεσης, το αποτέλεσμα των εκλογών μπορεί να διασταυρωθεί με την καταμέτρηση των ψηφοδελτίων.

- **Ηλεκτρονικά συστήματα απευθείας εγγραφής (DRE):** Τα συστήματα DRE (Direct-recording electronic voting machines) διευκολύνουν αρκετά την διαδικασία της ψηφοφορίας καθώς είναι εύκολα στην κατανόηση και γρήγορα στην καταμέτρηση. Συνήθως αποτελούνται από μηχανές που λειτουργούν με οθόνη αφής ή απλές οθόνες και καταγράφουν την ψήφο με το πάτημα κουμπιών. Οι ψήφοι αποθηκεύονται σε μνήμες και καταμετρούνται είτε με την μεταφορά της μνήμης και την εισαγωγή της σε κάποιο κεντρικό υπολογιστικό σύστημα είτε με την μετάδοση σε αυτό μέσω

τηλεφωνικών γραμμών και του διαδικτύου. Παρόλο που διευκολύνουν την διαδικασία της ψηφοφορίας ακόμα και για τα άτομα με αναπηρία στην όραση, υπάρχουν πολλά κενά ασφαλείας και κίνδυνοι στην χρήση τους. Συστήματα αυτής της κατηγορίας παρήγαγαν οι εταιρίας Diebold και Nedap οι οποίες κρατούσαν ένα πολύ μεγάλο μερίδιο της αγοράς των εκλογικών συστημάτων.

- **Σύστημα αποκόμματος VVPAT:** Το VVPAT (Voter-verified paper audit trail) ή αλλιώς VPR (verifiable paper record), δεν αποτελεί αυτόνομο σύστημα ηλεκτρονικής ψηφοφορίας αλλά συνδυάζεται με συστήματα τα οποία δεν συγκρατούν αποδεικτικά σε απτή μορφή όπως για παράδειγμα τα DRE. Το συγκεκριμένο υποσύστημα χρησιμοποιείται ως ένα ανεξάρτητο κομμάτι του συνολικού συστήματος ψηφοφορίας με σκοπό να παρέχει ένα απόκομμα στον ψηφοφόρο ο οποίος στην συνέχεια μπορεί να σιγουρέψει την συμμετοχή του στις εκλογές.

- **Συστήματα i-voting:** Είναι τα συστήματα απομακρυσμένης ψηφοφορίας που βασίζονται στο διαδίκτυο για την μετάδοση των ψήφων. Τα συστήματα αυτά μπορούν να λειτουργούν τελείως απομακρυσμένα είτε σε κιόσκι εκλογικών κέντρων. Και στις δυο περιπτώσεις τα συστατικά του συστήματος (λογισμικό και υλικό) είναι υπό τον έλεγχο των εκλογικών υπαλλήλων. Τα συστήματα που είναι βασισμένα στο διαδίκτυο είναι και τα πιο επικίνδυνα διότι υπάρχουν περισσότερα περιθώρια για επιθέσεις και σφάλματα που μπορούν να παραλύσουν την εκλογική διαδικασία μέχρι να μεταβάλλουν τα αληθινά αποτελέσματα. Παραδείγματα αυτών είναι τα Civitas, Helios και Pret A Voter.

ΥΠΟΚΕΦΑΛΑΙΟ 1.6: Προκλήσεις

Ο χώρος των ηλεκτρονικών εκλογών αντιμετωπίζει κάποιες δύσκολες προκλήσεις οι οποίες τελικά αποτελούν και την αφορμή για την δυσανασχέτηση της επιστημονικής κοινότητας (Springall et al 2014:713) αλλά ακόμη και για την διακοπή χρήσης τους. Δεν είναι τυχαίο άλλωστε που πάρα πολλές χώρες συνεχίζουν να χρησιμοποιούν τον παραδοσιακό τρόπο εκλογών (χάρτινα ψηφοδέλτια). Κοιτώντας το θέμα επιφανειακά μπορεί πολύ σωστά κανείς να χαρακτηρίσει περίεργο το γεγονός του ότι ενώ έχουμε δει αλματώδη ανάπτυξη στην τεχνολογία, το ζήτημα της ηλεκτρονικής ψηφοφορίας δεν έχει λυθεί. Κι όμως τα ηλεκτρονικά συστήματα ψηφοφοριών βασίζονται κυρίως στην επιστήμη των υπολογιστών και της κρυπτογραφίας, επιστήμες στις οποίες υπάρχει πολύς δρόμος ακόμα να διανυθεί από την επιστημονική κοινότητα. Αυτό συμβαίνει γιατί η τεχνολογία των υπολογιστών είναι συνεχώς μεταβαλλόμενη και οι τεχνικές κρυπτογράφησης θέλουν χρόνο για να δοκιμαστούν και να θεωρηθούν ασφαλείς.

Η πρώτη και βασικότερη πρόκληση ενός συστήματος είναι η ταυτόχρονη ικανοποίηση των ιδιοτήτων της ακεραιότητας και της εμπιστευτικότητας. Το πρόβλημα βρίσκεται στα αντίφαση μεταξύ των συμφερόντων τους. Η ακεραιότητα διασφαλίζεται εύκολα με την απλή ανύψωση του χεριού αλλά ταυτόχρονα καταστρέφεται η εμπιστευτικότητα. Η εμπιστευτικότητα ικανοποιείται ψηφίζοντας μυστικά αλλά αυτό δυσκολεύει πάρα πολύ την απόδειξη της ακεραιότητας, (Clarkson et al 2008:354) ειδικά όταν πρόκειται για ηλεκτρονικά συστήματα των οποίων η διασφάλιση της ασφάλειας αποτελεί ξεχωριστό τομέα της επιστήμης των υπολογιστών.

Η αποφυγή εξαναγκασμού παραμένει ένα από τα θεμελιώδη ζητήματα που απασχολούν την έρευνα στο χώρο του i-voting. Όπως και με την αντίφαση μεταξύ της ακεραιότητας και της μυστικότητας έτσι και εδώ είναι δύσκολη η ικανοποίηση της αποφυγής εξαναγκασμού συνάμα με την επαληθευσιμότητα. Για παράδειγμα το σύστημα Helios δεν έχει δυνατό μηχανισμό στο να αντιστέκεται στον εξαναγκασμό των ψηφοφόρων καθώς ο ψηφοφόρος μπορεί να αλλάξει την ψήφο του όσες φορές επιθυμεί μέχρι το τέλος των εκλογών και μόνο η τελευταία μετράει. Αυτό σημαίνει ότι αν τα στοιχεία ταυτοποίησης του χρήστη διαρρεύσουν—μια από τις συνηθέστερες περιπτώσεις παραβιάσεων—ο επιτιθέμενος θα μπορέσει να «ρίξει» την τελευταία ψήφο. Επιπλέον ο χρήστης μπορεί να είναι υπό την επήρεια σιωπηλού εξαναγκασμού, όπου ο ίδιος δεν γνωρίζει ότι ο επιτιθέμενος έχει εισβάλει με κάποιον τρόπο στην εφαρμογή ή την συσκευή πελάτη (Grewal et al 2013:367).

Στην συνέχεια υπάρχει το δίλημμα μεταξύ λογισμικού ανοιχτού και κλειστού κώδικα. Κρατώντας το λογισμικό ανοιχτό ο καθένας μπορεί να ελέγξει τον κώδικα και να σιγουρευτεί ότι δεν υπάρχουν λάθη ή κακοήθη μέρη τα οποία μπορούν να απειλήσουν την διαδικασία των εκλογών. Ακόμα, οι πολίτες μπορούν να προσφέρουν στην συντήρηση, επισκευή και εξέλιξη του λογισμικού. Αν και αυτά προϋποθέτουν ένα μεγάλο σύνολο γνώσεων, η διαδικασία φαίνεται να είναι πιο δημοκρατική. Η πλευρά αυτή έχει βέβαια και τα μειονεκτήματα της καθώς τυχόν ευπάθειες μπορούν να εντοπιστούν από τρίτους με αποτέλεσμα να γίνεται πιο εύκολο το έργο αυτών που θέλουν να αλλοιώσουν τα αποτελέσματα και την διαδικασία της ψηφοφορίας. Το συγκεκριμένο πρόβλημα λύνεται κρατώντας μέρη ή και ολόκληρο το λογισμικό κρυφό από το κοινό, αλλά χάνεται έτσι η εμπιστοσύνη του κόσμου εφόσον δεν μπορεί πλέον ο απλός πολίτης να εξετάσει την λειτουργία των συστημάτων. Σε αυτή την περίπτωση ο έλεγχος βρίσκεται σε περιορισμένο αριθμό ατόμων, κάτι που διευκολύνει την εξαγορά αυτών με στόχο την αλλοίωση του αποτελέσματος. Για αυτόν τον λόγο, σε πολλές χώρες έχουν δημιουργηθεί ανεξάρτητες αρχές και εταιρίες που αναλαμβάνουν την πιστοποίηση των εκλογικών συστημάτων με τους προβλεπόμενους νόμους της εκάστοτε χώρας.

Ένα σύστημα το οποίο χρησιμοποιεί ως μέσω μετάδοσης το διαδίκτυο προσθέτει αρκετή πολυπλοκότητα στην διασφάλιση της ασφάλειας του. Με την απευθείας σύνδεση στο διαδίκτυο δίνεται η δυνατότητα επίθεσης από απόσταση με

αποτέλεσμα να είναι δύσκολο έως αδύνατο να εντοπιστεί ο επιτιθέμενος. Ο κυβερνοπόλεμος, μια κάποτε υποθετική απειλή, έχει μετατραπεί σε μια τεκμηριωμένη μελέτη. Επιθέσεις από ξένα κράτη αποτελούν μια πιθανή απειλή για ένα εθνικό εκλογικό σύστημα (Springall et al 2014:703). Το σύστημα ηλεκτρονικής ψηφοφορίας θα πρέπει να ακολουθεί πολύ αυστηρή πολιτική ασφαλείας για να αποτρέψει πιθανές επιθέσεις. Κάποιος θα θεωρούσε αυτονόητη την συμμόρφωση του προσωπικού με τους προβλεπόμενους κανόνες και την πολιτική ασφαλείας όταν επρόκειτο για αληθινές εκλογές, κάτι το οποίο δεν έγινε όπως αναφέρεται στα ευρήματα της έρευνας (Springall et al 2014:706-7). Το υλικό και το λογισμικό που χρησιμοποιείται από τις οντότητες του συστήματος θα πρέπει να είναι δοκιμασμένο και εξετασμένο από ειδικούς και επιτροπές. Όλα αυτά όμως θέλουν χρόνο και εκτενής μελέτη από άτομα του χώρου. Στην περίπτωση που το σύστημα λειτουργεί με εφαρμογή πελάτη η εγγύηση της ασφάλειας φαντάζει αδύνατη. Αυτό συμβαίνει διότι δεν υπάρχει έλεγχος στις συσκευές πελάτη π.χ. (προσωπικός υπολογιστής, smartphone κτλ.). Η συσκευή πελάτη ενός χρήστη θα μπορούσε να είναι μολυσμένη με κακόβουλο λογισμικό ή ακόμα να στοχοποιηθεί και να παραβιαστεί κατά την διάρκεια των εκλογών. Δεν είναι λίγα τα κρούσματα ransomware που έχουν παρατηρηθεί τα τελευταία χρόνια σε υπολογιστές τόσο προσωπικούς όσο και εταιρικούς. Μια ευπάθεια στο λειτουργικό σύστημα της συσκευής ή στα κανάλια μετάδοσης μπορεί να αποβεί μοιραία για την εκλογική διαδικασία. Μη προβλεπόμενες ευπάθειες όπως τα λεγόμενα zero-day exploits (βλ. OpenSSL Heartbleed bug) μπορούν να αχρηστέψουν τα περισσότερα μέτρα ασφαλείας που έχουν εφαρμοστεί.

Τα νομοθετικά πλαίσια που καθορίζουν την διαδικασία της ψηφοφορίας απαιτούν μεγάλα χρονικά διαστήματα για να εξελιχθούν. Για παράδειγμα στην περίπτωση της Ελβετίας όπου η εισαγωγή της επιστολικής ψηφοφορίας διήρκησε τρεις δεκαετίες και συνέχισε να εξελίσσεται τουλάχιστον ως το 2015 όπως αναφέρεται στην μελέτη (Serdult et al 2015:127). Μέχρι τις αρχές του 21^{ου} αιώνα πολλές χώρες δεν είχαν προβλέψει για την νομοθεσία που αφορά την ηλεκτρονική ψηφοφορία ή ήταν αρκετά απαρχαιωμένη όπως στην περίπτωση της Ολλανδίας (Krimmer et al 2008:3).

Η εμπιστοσύνη του κόσμου χάνεται εύκολα όταν υπάρχουν υποψίες ότι ένα σύστημα ψηφοφορίας υπολειτουργεί ή ότι είναι ευπαθές σε επιθέσεις. Όταν χάνεται η εμπιστοσύνη του κόσμου στα εκλογικά συστήματα, δύσκολα κερδίζεται πίσω (Krimmer et al 2008:10). Η εμπειρία του μέσου πολίτη που αφορά ιούς, καταστροφή ψηφιακών αρχείων, σφάλματα προγραμμάτων μαζί με την αυξημένη ευκολία λήψης προγραμμάτων μέσω του διαδικτύου έχουν ενισχύσει την γνώμη ότι τα συστήματα ηλεκτρονικής ψηφοφορίας μπορεί να πάσχουν από τις ίδιες ευπάθειες (Deutsch 2005:94). Υπάρχουν προειδοποιήσεις ότι αντικαθιστώντας την παραδοσιακή μέθοδο ψηφοφορίας (paper-based) με την ηλεκτρονική, εξαιρείται ένα ευμέγεθες σύνολο ψηφοφόρων εκ των οποίων το μεγαλύτερο ποσοστό

αποτελείται από άτομα μεγάλης ηλικίας και άτομα με χαμηλό γνωστικό επίπεδο στην επιστήμη των υπολογιστών (Osho et al 2015:208).

ΥΠΟΚΕΦΑΛΑΙΟ 1.7: Κόστος

Το κόστος της ηλεκτρονικής ψηφοφορίας είναι αμφιλεγόμενο όταν πρόκειται για τα συστήματα DRE ή οποιαδήποτε μορφή i-voting. Οι κατακριτές των ηλεκτρονικών εκλογών υποστηρίζουν ότι το κόστος αγοράς και συντήρησης των συστημάτων είναι υψηλότερο από το κόστος των παραδοσιακών εκλογών ειδικά όταν δεν αποτελεί την κυρία μέθοδο αλλά προσφέρεται ως εναλλακτική λύση ψηφοφορίας. Από την άλλη πλευρά οι υποστηρικτές δηλώνουν ότι με την πάροδο του χρόνου ένα ηλεκτρονικό σύστημα μειώνει το συνολικό κόστος της διαδικασίας εκλογών όταν τα έξοδα εκτύπωσης και αποστολής μπορούν να εξοικονομηθούν. Υπάρχει και μια τρίτη άποψη η οποία θεωρεί το κόστος μη σχετικό παράγοντα από την ώρα που αυξάνεται η προσιτότητα και η συμμετοχή του κοινού ειδικά όταν πρόκειται για ομογενείς μετανάστες και άτομα με αναπηρία. Όταν λαμβάνονται υπόψιν τα κόστη ενός ηλεκτρονικού συστήματος πρέπει να εξεταστούν τα παρακάτω:

Μη επαναλαμβανόμενα κόστη:

Λογισμικό:

- Ανάπτυξη του συστήματος (ανάπτυξη και έλεγχος)
- Κόστος προμήθειας και εγκατάστασης επιπλέον λογισμικού (λογισμικού ασφαλείας, βάσεων δεδομένων κ.α.)
- Τέλη αδειών χρήσης (διάφορες άδεις λογισμικού και μετάδοσης)

Υποδομές:

- Κόστος προμήθειας και εγκατάστασης κεντρικών υποδομών για το σύστημα (κέντρο υπολογιστών, εξυπηρετητές, τοίχος προστασίας firewall, δικτύωση, κόστος χτισίματος κτιρίου)
- Κόστος προμήθειας και εγκατάστασης αποκεντρωμένων υποδομών για το σύστημα (επιπλέον υλικό, λογισμικό όπως για παράδειγμα scanners κλπ.)

Εκλογικός κατάλογος:

- Κόστος ανάπτυξης και προμήθειας βάσης δεδομένων εκλογικού καταλόγου
- Ανάπτυξη διεπαφής εκλογικού καταλόγου-υποψηφίων

Ρυθμίσεις συστήματος:

- Περαιτέρω ανάπτυξη του συστήματος (ανάπτυξη και έλεγχος του λογισμικού)

Πιστοποίηση:

- Κόστος πιστοποίησης του συστήματος από εξωτερικό οργανισμό

Ετήσια επαναλαμβανόμενα κόστη/επενδύσεις:

Ηλεκτρονικό σύστημα:

- Λειτουργικά κόστη (κόστος συντήρησης και αδειοδότησης του λογισμικού του συστήματος)
- Λειτουργικά κόστη των υποδομών του συστήματος (κόστη συντήρησης για το κέντρο υπολογιστών, τους εξυπηρετητές, την δικτύωση κ.τ.λ.)
- Εξωτερικοί έλεγχοι ασφαλείας (κόστη ελέγχων εισβολής penetration testing, καταγραφής, συνοδευτικές ελεγκτικές ομάδες)

Ανάπτυξη του ηλεκτρονικού συστήματος:

- Κόστος έρευνας και περαιτέρω ανάπτυξης (συμβόλαια μελέτης, συνεδριάσεις ομάδας εργασίας κ.λπ.)

Συγκεκριμένα κόστη ανά εκλογή/δημοψήφισμα:

- Πληροφόρηση ψηφοφόρων (οδηγίες για τους ψηφοφόρους)
- Χαρτί και κόστη εκτυπώσεων
- Ταχυδρομικά τέλη (επιπλέον κόστη συγκρινόμενα με μη ηλεκτρονικές εκλογές)
- Κόστος για την παροχή δεδομένων από τα μητρώα ψηφοφόρων
- Κόστος καταμέτρησης (επιπλέον κόστη συγκρινόμενα με μη ηλεκτρονικές εκλογές)

Προσωπικό:

- Διαχείριση έργου
- Εξωτερικές εντολές

Δημόσιες σχέσεις:

- Κόστος διαδικτυακών δημοσιεύσεων και εκτύπωσης αυτών

Απόσβεση μεμονωμένων επενδύσεων:

- Υλικό (ανά 5 χρόνια)
- Λογισμικό (ανά 3 χρόνια)
- Άδειες (μέχρι την λήξη τους)
- Πιστοποιήσεις (μέχρι την λήξη τους)

ΥΠΟΚΕΦΑΛΑΙΟ 1.8: Πλεονεκτήματα

Το αρχικό κόστος της εισαγωγής συστημάτων ηλεκτρονικής ψηφοφορίας στην περίπτωση του i-voting μπορεί είναι υψηλό όταν πρόκειται για εθνικές εκλογές άλλα όχι υψηλότερο όταν χρησιμοποιείται ήδη κάποιο άλλο ηλεκτρονικό σύστημα (Clarkson et al 2008:365). Παρόλα αυτά στην πάροδο του χρόνου το κόστος μειώνεται σε σχέση με τις παραδοσιακές μεθόδους καθώς εξαλείφονται τα έξοδα εκτυπώσεων, μεταφορών και κυρίως του ανθρώπινου δυναμικού. Το κόστος ανά ψηφοφόρο μειώνεται όσο αυξάνεται ο αριθμός των συμμετοχών. Τα πλεονεκτήματα που θα δούμε παρακάτω δικαιολογούν το κόστος τέτοιων συστημάτων.

Από τα πρώτα πράματα που έρχονται στο μυαλό όταν σκέφτεται κανείς τις ηλεκτρονικές εκλογές, είναι η ταχύτητα της διεξαγωγής της διαδικασίας αλλά και της καταμέτρησης των αποτελεσμάτων. Στις εθνικές εκλογές θα έλεγε κανείς ότι η ταχύτητα δεν μετράει τόσο εφόσον η συχνότητα αυτών είναι μικρή (ανά 2-4 χρόνια). Παρόλα αυτά η ταχύτητα είναι ευνοϊκή στα δημοψηφίσματα τα οποία συμβαίνουν πολύ συχνότερα ειδικότερα στις ανεπτυγμένες ευρωπαϊκές χώρες. Η ταχύτητα επηρεάζει άμεσα τους ψηφοφόρους και συγκεκριμένα την συμμετοχή τους στην διαδικασία. Ένα αργό σύστημα εκλογών θα δημιουργήσει ουρές στα κέντρα εκλογών και αργή εξυπηρέτηση (ακόμα και με i-voting συστήματα) τα οποία μεταφράζονται σε μεγαλύτερη αναμονή και μεγαλύτερη πιθανότητα απουσίας ψηφοφόρων.

Στο σενάριο που το σύστημα εν χρήσει είναι ικανό και καλύπτει τα βασικά κριτήρια μπορεί να εγγυηθεί μεγαλύτερη ασφάλεια από την παραδοσιακή μέθοδο ψηφοφορίας του χαρτιού. Αυτό εξηγείται με την δραστική μείωση του ανθρώπινου παράγοντα από τα κρίσιμα μέρη της εκλογικής διαδικασίας. Ένα σύστημα, για το οποίο έχει γίνει εκτενής έρευνα και οι διαδικασίες ανάπτυξης και ελέγχου υλικού και λογισμικού έχουν εκτελεστεί σωστά, τα περιθώρια για λάθη μειώνονται σημαντικά σε σχέση με την χειρωνακτική μέθοδο.

Πέρα από την ταχύτητα, το e-voting διευκολύνει την συμμετοχή των ανθρώπων με αναπηρία μέσω της διάθεσης φωνητικής συνοδού και διεπαφής braille. Ψηφοφόροι που δεν μπορούν να παρευρεθούν λόγω απόστασης ή υποχρεώσεων έχουν την δυνατότητα να ψηφίσουν με την χρήση του διαδικτύου αντικαθιστώντας την επιστολική ψήφο η οποία δεν εμπνέει μεγάλη εμπιστοσύνη. Οι εκλογές μέσω του διαδικτύου λύνουν τα περισσότερα αν όχι όλα τα προβλήματα της προσιτότητας.

Με σωστές μεθοδολογίες μπορεί να διασφαλιστεί η ευστοχία των αποτελεσμάτων μέχρι και την τελευταία ψήφο κάτι που απαιτεί πολλούς πόρους στην χειρωνακτική μέθοδο. Η χρήση των υπολογιστών κάνουν ακόμα πιο εύκολη την μετέπειτα παρουσίαση των αποτελεσμάτων και των στατιστικών της ψηφοφορίας. Η

πληροφορία στην σημερινή εποχή είναι πολύ χρήσιμη διότι μπορεί να χρησιμοποιηθεί για την εξαγωγή διάφορων συμπερασμάτων (data mining).

Τα ηλεκτρονικά συστήματα αυξάνουν δραματικά την ευχρηστία όσον αφορά την εκλογική διαδικασία. Η διάδοση της πληροφορίας γίνεται εύκολο έργο όταν αυτή υπάρχει σε ηλεκτρονική μορφή. Οδηγίες μπορούν να παρέχονται στο διαδίκτυο ή ακόμα και στο ίδιο το σύστημα την ώρα της ψηφοφορίας στο εκλογικό κέντρο. Με την βοήθεια ειδικών οι διεπαφές μπορούν να γίνουν φιλικές στον άνθρωπο κάνοντας την διαδικασία κατανοητή και λιγότερο επίπονη και σε ανθρώπους με λιγότερες γνώσεις υπολογιστών.

Το κυριότερο και βασικότερο πλεονέκτημα είναι η αύξηση της συμμετοχής του κοινού. Όπως φαίνεται η συμμετοχή επηρεάζεται από πολλούς παράγοντες εκ των οποίων η ασφάλεια, η προσιτότητα και η ταχύτητα αποτελούν τους πιο σημαντικούς. Ένα ηλεκτρονικό σύστημα είναι ικανό να καλύψει όλες αυτές τις ανάγκες σε μεγάλο ποσοστό και σε αποδεκτό κόστος.

Ο συνδυασμός της ηλεκτρονικής ψηφοφορίας με την ηλεκτρονική διακυβέρνηση μπορεί να ενισχύσει την δημοκρατία και να κάνει και το έργο της ψηφοφορίας ακόμα ευκολότερο ειδικά στο κομμάτι της ταυτοποίησης όπως για παράδειγμα στην Εσθονία. Η χρήση ταυτοτήτων smartcards δίνουν την δυνατότητα της ταυτοποίησης και της κρυπτογράφησης της ψήφου (Springall et al 2014:704).

ΥΠΟΚΕΦΑΛΑΙΟ 1.9: Μειονεκτήματα

Συστήματα που δεν παρέχουν εναλλακτικό τρόπο επιβεβαίωσης (ψηφοδέλτια, αποδείξεις) των αποτελεσμάτων όπως π.χ. τα D.R.E προσδίδουν στο ρίσκο επανάληψης των εκλογών. Γι' αυτό φταίει η ολική εξάρτηση στο υλικό π.χ. (μνήμες) οι οποίες είναι εύκολο να κλαπούν ή να καταστραφούν. Χωρίς τρόπο να γίνει η επαναμέτρηση των ψήφων, δεν υπάρχει άλλη λύση πέρα από την επανάληψη της διαδικασίας.

Η ανάγκη για πιστοποίηση από τρίτους αυξάνει το κόστος και την διάρκεια της ανάπτυξης του συστήματος. Για να διατηρηθεί η εμπιστοσύνη του κόσμου στο θέμα αυτό, το κράτος και ο ελεγκτικός οργανισμός θα πρέπει να συνεργάζονται με διαφάνεια. Αυτό προϋποθέτει σωστές μεταρρυθμίσεις από την μεριά της κυβέρνησης, τον καθορισμό και την εφαρμογή κάποιων standards. Όλα αυτά αυξάνουν την πολυπλοκότητα, αποθαρρύνοντας έτσι το εκλογικό σώμα να δεχτούν την υιοθέτηση ηλεκτρονικών συστημάτων ψηφοφορίας.

Θέτοντας σε εφαρμογή ένα σχέδιο για την χρήση ηλεκτρονικών συστημάτων εκλογών, αυτόματα το κράτος εξαρτάται από τους πωλητές για την προμήθεια και την συντήρηση τους. Σε περίπτωση ολιγοπωλίου (Krimmer et al 2008:4) το κράτος χάνει ήδη αρκετό έδαφος όσο αφορά τις συμφωνίες για το κόστος προμήθειας των

και συστημάτων και των αδειών. Λίγες είναι οι χώρες που θα καταφέρουν να ανεξαρτοποιηθούν από τρίτους. Έτσι μόλις η κυβέρνηση εισάγει την ηλεκτρονική ψηφοφορία, πρέπει να συνεχίσει να ασκεί πίεση και να μην κάνει πίσω αφήνοντας την αγορά να κυριαρχήσει (Krimmer et al 2008:10).

Άλλο ένα μειονέκτημα είναι η μικρότερη συμμετοχή από ανθρώπους μεγάλης ηλικίας και ανθρώπους που έχουν καθόλου έως λίγες γνώσεις υπολογιστών. Σε μια χώρα όπως η Ελλάδα για παράδειγμα, η οποία αντιμετωπίζει το δημογραφικό πρόβλημα, το ποσοστό αυτών των ανθρώπων θα είναι εμφανές. Ευτυχώς με την ραγδαία ανάπτυξη και χρήση της τεχνολογίας το πρόβλημα αυτό θα εξαλειφθεί εντελώς στις επόμενες δεκαετίες. Η χρήση υπολογιστών πλέον διδάσκεται από τις πρώτες τάξεις στα περισσότερα σχολεία. Η διείσδυση των φορητών συσκευών (smartphones, tablets) αυξάνει τις πιθανότητες εκμάθησης από νεαρή ηλικία.

Με την μεταβολή από τον παραδοσιακό στον ηλεκτρονικό τρόπο ψηφοφορίας έρχεται και το βάρος της εκμάθησης των ψηφοφόρων. Είναι κάτι που πρέπει να γίνει σωστά και έχει κόστος. Η συμμετοχή μπορεί να μειωθεί εύκολα αν δεν δοθεί προσοχή κατά την εισαγωγή του συστήματος. Ιδανικά θα ήταν σωστό να δίνεται η ηλεκτρονική ψηφοφορία σαν εναλλακτική λύση μέχρι να εξοικειωθεί το κοινό στην χρήση του.

Σε ένα σύστημα στο οποίο δεν ακολουθούνται σωστές πρακτικές κατά την ανάπτυξη του (Kohno et al 2004:46), δυσχεραίνεται ο εντοπισμός σφαλμάτων αφήνοντας την διαδικασία ανάκαμψης στο έλεος της τύχης. Πέρα από τις σωστές πρακτικές χρειάζονται και μηχανισμοί καταγραφής (auditing) οι οποίοι μειώνουν την μυστικότητα.

ΚΕΦΑΛΑΙΟ 2: Πλατφόρμα ηλεκτρονικής ψηφοφορίας

ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο αναλύεται η υλοποίηση της πλατφόρμας και οι τεχνολογίες που χρησιμοποιήθηκαν. Γίνεται μια περιγραφή των βασικών κομματιών που απαρτίζουν το σύστημα (εξυπηρετητής, σχήμα βάσης) και του συνολικού σχεδιασμού.

ΥΠΟΚΕΦΑΛΑΙΟ 2.1: Αρχιτεκτονική Rest

Η αρχιτεκτονική REST (Representational state transfer) ή αλλιώς Restful web services είναι ένας τρόπος για την παροχή διαλειτουργικότητας μεταξύ υπολογιστικών συστημάτων στον παγκόσμιο ιστό. Οι υπηρεσίες που είναι συμβατές με το REST επιτρέπουν στα αιτούμενα συστήματα να έχουν πρόσβαση και να χειρίζονται τις παραστάσεις κειμένου των πόρων του διαδικτύου χρησιμοποιώντας ένα ομοιόμορφο και προκαθορισμένο σύνολο λειτουργιών που δεν συντηρούν κατάσταση. Άλλες μορφές υπηρεσιών web που εκθέτουν τα δικά τους αυθαίρετα σύνολα λειτουργιών είναι τα WSDL και το SOAP. Παρόλα αυτά η αρχιτεκτονική REST έχει δει μεγάλη δημοτικότητα καθώς είναι πλέον ο standard τρόπος για να χτίσει κανείς εμπορικές και όχι μόνο υπηρεσίες.

Οι «δικτυακοί πόροι» ορίστηκαν για πρώτη φορά στον παγκόσμιο ιστό ως έγγραφα ή αρχεία που προσδιορίζονται από τις διευθύνσεις URL τους. Σήμερα όμως έχουν έναν πολύ πιο γενικό και αφηρημένο ορισμό που περιλαμβάνει κάθε

πράγμα ή οντότητα που μπορεί να αναγνωριστεί, να ονομαστεί, να αντιμετωπιστεί με οποιονδήποτε τρόπο στο διαδίκτυο. Σε μια Restful υπηρεσία, τα αιτήματα που υποβάλλονται στο URI ενός πόρου θα προκαλέσουν μια απάντηση που μπορεί να είναι XML, HTML, JSON ή κάποια άλλη καθορισμένη μορφή. Η απόκριση μπορεί να επιβεβαιώσει ότι έχουν γίνει ορισμένες αλλαγές στον αποθηκευμένο πόρο και μπορεί να παρέχει συνδέσεις υπερκειμένου με άλλους σχετικούς πόρους ή συλλογές πόρων. Χρησιμοποιώντας το HTTP, όπως είναι πιο συνηθισμένο, το είδος των διαθέσιμων λειτουργιών περιλαμβάνει εκείνες που είναι προκαθορισμένες από τα ρήματα (HTTP), δηλαδή GET, PUT, POST, DELETE και ούτω καθεξής.

Χρησιμοποιώντας ένα stateless πρωτόκολλο και τις τυποποιημένες λειτουργίες, τα συστήματα REST επιδιώκουν την γρήγορη απόδοση, την αξιοπιστία και την ικανότητα ανάπτυξης, επαναχρησιμοποιώντας συστατικά που μπορούν να διαχειρίζονται και να ενημερώνονται χωρίς να επηρεάζουν το σύστημα στο σύνολο του, ακόμα και όταν εκτελείται. Ο όρος REST εισήχθη και καθορίστηκε το 2000 από τον Roy Fielding στην διδακτορική διατριβή του. Ο Fielding χρησιμοποίησε το REST για να σχεδιάσει το HTTP 1.1 και το Uniform Resource Identifiers (URI). Ο όρος αποσκοπεί στην δημιουργία μιας εικόνας για το πώς συμπεριφέρεται μια καλά σχεδιασμένη εφαρμογή Web: είναι ένα δίκτυο πόρων όπου ο χρήστης προχωρά μέσα από την εφαρμογή επιλέγοντας συνδέσμους, όπως (/user/tom), και λειτουργίες όπως GET ή DELETE, με αποτέλεσμα η ανανεωμένη κατάσταση του πόρου να μεταφέρεται στον χρήστη.

ΥΠΟΚΕΦΑΛΑΙΟ 2.2: Play Framework

ΥΠΟΚΕΦΑΛΑΙΟ 2.2.1: Βασική περιγραφή

Το Play είναι ένα πλαίσιο εφαρμογών ιστού ανοιχτού κώδικα, γραμμένο στην γλώσσα Scala. Παρόλα αυτά, στις πιο πρόσφατες εκδόσεις, το πλαίσιο είναι και χρησιμοποιήσιμο από την γλώσσα γενικής χρήσης Java μέσω της διεπαφής API. Το Play ακολουθεί την αρχιτεκτονική μοντέλου-προβολής-ελεγκτή (MVC). Στόχος του πλαισίου είναι να βελτιστοποιήσει την παραγωγικότητα των προγραμματιστών χρησιμοποιώντας συμβατικές ρυθμίσεις (convention over configuration), επί τόπου φόρτωση κώδικα και προβολή σφαλμάτων στον φυλλομετρητή. Το Play είναι βαριά εμπνευσμένο από τα ASP.NET MVC, Ruby on Rails, και Django και μοιάζει αρκετά με όλη αυτή την οικογένεια των πλαισίων. Το περιβάλλον του Play θυμίζει λιγότερο αυτό του Java Enterprise Edition. Αυτό κάνει το πλαίσιο πιο απλό στην ανάπτυξη εφαρμογών αντίθετα με το Enterprise το οποίο προσθέτει μεγαλύτερη πολυπλοκότητα και έχει τους δικούς του περιορισμούς.

Η υποστήριξη της Scala από το πλαίσιο υπάρχει από την έκδοση 1.1. Στην έκδοση 2.0 του πλαισίου, ο πυρήνας του ξαναγράφτηκε στην Scala κάνοντας την γλώσσα first class citizen. Η κατασκευή (build) και η εγκατάσταση (deployment) μεταφέρθηκαν στο SBT project και τα πρότυπα (templates) χρησιμοποιούν Scala αντί για Groovy. Εκδόσεις μεγαλύτερες της 2.5 έχουν απαραίτητη προϋπόθεση το Java 8 JVM.

Το Play 2.5.x κάνει χρήση των παρακάτω Java βιβλιοθηκών:

- Netty ή Akka-HTTP για τον εξυπηρετητή δικτύου.
- Akka Streams για ασύγχρονες λειτουργίες και λειτουργίες ροής.
- Δεν είναι απαραίτητο κάποιο ORM αλλά υποστηρίζονται τα Anorm(Scala), Slick(Scala), και Ebean (Java) για την επικοινωνία με την βάση δεδομένων.
- Twirl (Scala) σαν μηχανή template.
- Ενσωματωμένο επί τόπου φόρτωμα (hot reloading) κώδικα.
- SBT ως εργαλεία για building και dependency management.

Οι παρακάτω λειτουργίες βρίσκονται στον πυρήνα του πλαισίου:

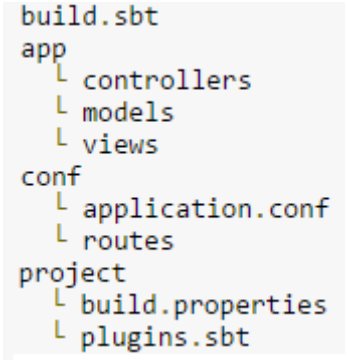
- Ένα καθαρό RESTful πλαίσιο.
- CRUD: μια ενότητα για την απλοποίηση της επεξεργασίας αντικειμένων μοντέλων.
- Secure: μια ενότητα για την ενεργοποίηση απλής ταυτοποίησης χρήστη.
- Ένα πλαίσιο επικύρωσης (validation) βασισμένο σε annotations.
- JSON και XML parsers και marshallers.
- Ένα persistence layer βασισμένο στο JPA.
- Μια ενσωματωμένη βάση δεδομένων για σκοπούς γρήγορης εγκατάστασης (deployment) και ελέγχου (testing).
- Ένα πλήρες ενσωματωμένο πλαίσιο ελέγχων (testing).
- Μια αυτόματη λειτουργία upload για αρχεία.
- Μια αρθρωτή αρχιτεκτονική, η οποία επιτρέπει την εύκολη προσθήκη νέων χαρακτηριστικών στον πυρήνα.

Γνωστές ιστοσελίδες που χρησιμοποιούν το Play είναι οι: LinkedIn, Coursera, Lightbend, SyncVideo, Samsung Artik και πολλές άλλες.

ΥΠΟΚΕΦΑΛΑΙΟ 2.2.2: Δομή ενός project

Η δομή ενός Play Java project είναι αρκετά απλή (βλ. εικόνα 1):

- **Αρχείο build.sbt:** Εδώ βρίσκονται οι δηλώσεις εξωτερικών βιβλιοθηκών.



Εικόνα 1 Δομή Play Java Project

- **Κατάλογος app:** Ο κατάλογος περιέχει τρεις βασικούς υποκαταλόγους. Στον υποκατάλογο controllers βρίσκονται τα .java αρχεία «ελεγκτές» στα οποία βρίσκεται η βασική λογική για την επεξεργασία των εισερχόμενων αιτημάτων. Ο κατάλογος models περιέχει τα αρχεία μοντέλα τα οποία «βλέπει» και το ORM. Ο κατάλογος views κρατάει τα αρχεία .scala.html τα οποία χρησιμοποιούνται στο σερβίρισμα ιστοσελίδων.
- **Κατάλογος conf:** Εδώ βρίσκεται το κυριότερο αρχείο του πλαισίου, το αρχείο routes. Στο αρχείο αυτό γίνεται η δρομολόγηση των αιτημάτων με βάση το HTTP ρήμα (GET, POST, PATCH, DELETE) και το URI. Μπορεί κανείς να δηλώσει ποια μέθοδος (αρχείου controller) θα είναι υπεύθυνη για την εξυπηρέτηση αιτημάτων που έρχονται σε συγκεκριμένο μονοπάτι και με συγκεκριμένη λειτουργία. Στο αρχείο application.conf μπορεί κανείς να ρυθμίσει διάφορες λειτουργίες του play, όπως σύνδεση βάσης δεδομένων, λειτουργία http και πολλών άλλων.
- **Κατάλογος project:** Στο αρχείο plugins.sbt δηλώνονται οι επιπρόσθετες λειτουργίες όπως play enhancer, ebean κλπ.

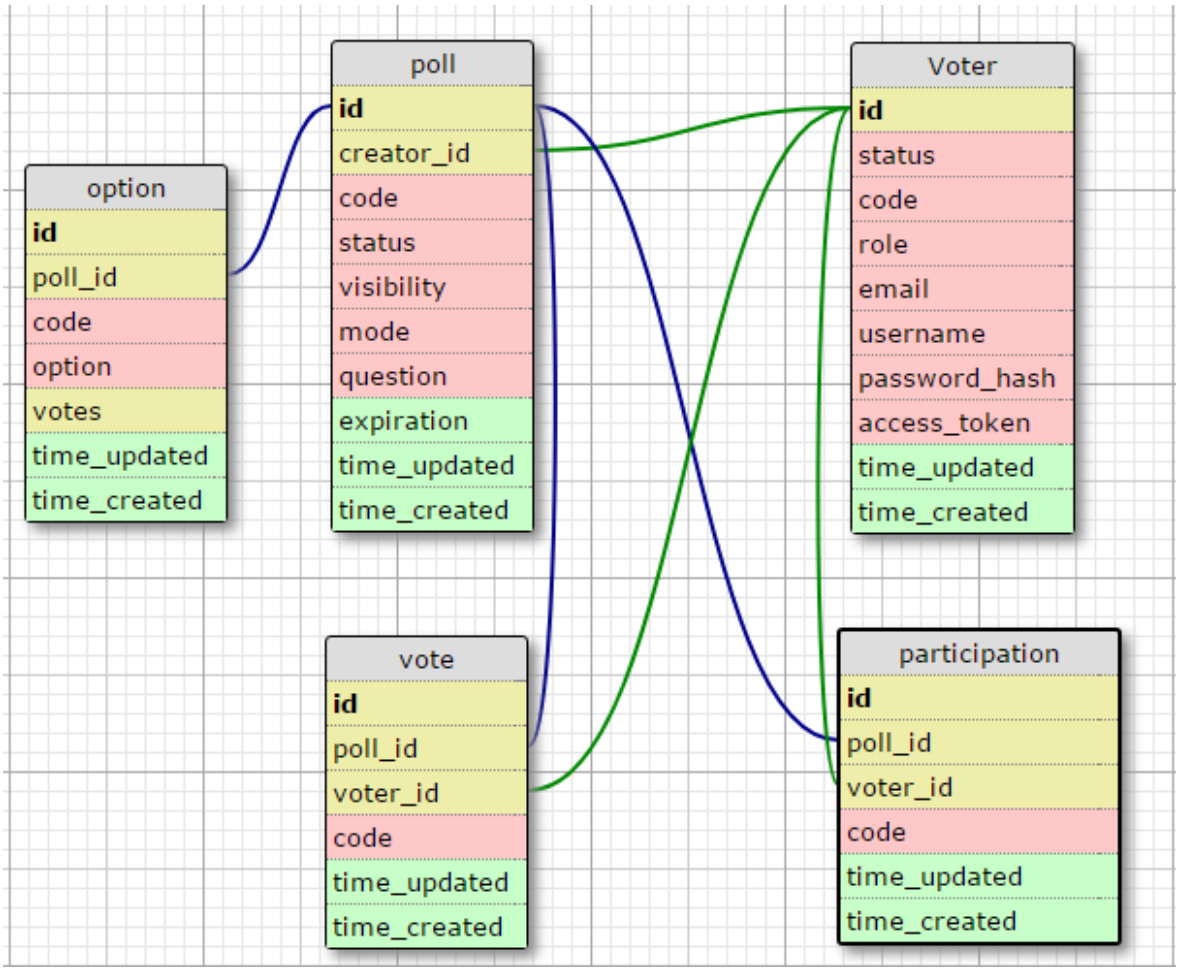
ΥΠΟΚΕΦΑΛΑΙΟ 2.3: Υλοποίηση

ΥΠΟΚΕΦΑΛΑΙΟ 2.3.1: Βάση δεδομένων

Για το κομμάτι της βάσης δεδομένων γίνεται η χρήση μιας σχεσιακής βάσης και πιο συγκεκριμένα της PostgreSQL. Η επιλογή έγινε τυχαία και δεν υπήρχε κάποιος πολύ σοβαρός λόγος για την προτίμηση της Postgres. Παράλληλα γίνεται και η χρήση του προγράμματος pgAdmin 3 για την διευκόλυνση κάποιων λειτουργιών και για την καλύτερη οπτική των δεδομένων. Αυτό βέβαια δεν είναι απαραίτητο για

όποιον θα ήθελε να χρησιμοποιήσει την βάση, γι' αυτό αρκεί και η απλή εγκατάσταση της Postgres.

Το σχήμα της βάσης (βλ. εικόνα 2) είναι σχετικά απλό. Οι κύριες οντότητες είναι ο ψηφοφόρος (voter) και η ψηφοφορία poll. Πιο συγκεκριμένα, μια ψηφοφορία μπορεί να έχει μια ή περισσότερες επιλογές option και έναν δημιουργό μόνο για κάθε εγγραφή. Ο ψηφοφόρος voter μπορεί να συμμετέχει σε μια ψηφοφορία μέσω της σχέσης participation και να ψηφίζει μέσω της σχέσης vote. Το σχήμα αυτό δίνει την δυνατότητα πρόσκλησης χρηστών σε ψηφοφορίες χωρίς να έχουν ψηφίσει. Επίσης όπως φαίνεται δεν κρατείται πουθενά σχέση χρήστη-ψήφου-ψηφοφορίας. Αυτό επιλέχθηκε σαν προστατευτικό μέτρο για την ανωνυμία του χρήστη. Το αρνητικό με αυτή τη κίνηση είναι ότι σε περίπτωση σφάλματος η εκλογή θα πρέπει να επαναληφθεί καθώς δεν υπάρχει τρόπος για την ανάκτηση των επιλογών του κάθε χρήστη.



Εικόνα 2 Σχήμα της βάσης δεδομένων

ΥΠΟΚΕΦΑΛΑΙΟ 2.3.2: Μοντέλα και ORM

Για την ευκολότερη μεταχείριση των δεδομένων γίνεται η χρήση του Ebean ORM πλαισίου. Το Ebean είναι ένα πλαίσιο γραμμένο σε Java, το οποίο αναλαμβάνει την μετατροπή των δεδομένων της βάσης σε αντικείμενα java και το αντίστροφό κατή την προσπέλαση και την αποθήκευση στην βάση. Είναι σχεδιασμένο για να είναι απλούστερο στη χρήση και την κατανόηση από τις υπόλοιπες υλοποιήσεις των JPA (Java Persistence API) και JDO (Java Data Objects). Αυτό το καταφέρνει μέσα από την αρχιτεκτονική της η οποία δεν κρατάει συνόδους (session-less). Το Ebean δεν απαιτεί την ύπαρξη διαχειριστή οντολογιών (JPA Entity Manager, JDO Persistence Manager) και αυτό αφαιρεί την έννοια των detached/attached beans και τα προβλήματα με τον καθαρισμό και την διαχείριση των συνόδων. Όλα αυτά καθιστούν το Ebean API πολύ ευκολότερο στην κατανόηση και στην χρήση του. Παρόλο που το Ebean έχει πλήρη χαρακτηριστικά ORM (ισοδύναμο με το JPA), έχει ενσωματώσει SQL/σχεσιακές δυνατότητες και στοιχεία. Ο λόγος είναι πως πολλές περιπτώσεις χρήσης ανάπτυξης απαιτούν τον έλεγχο της ακριβούς sql, των αποθηκευμένων διαδικασιών, ή απλώς επιλύονται πιο απλά με τον «σχεσιακό» τρόπο. Ο απώτερος σκοπός του Ebean είναι ο συνδυασμός των καλύτερων χαρακτηριστικών του JPA και των σχεσιακών χαρακτηριστικών όπως αυτών του MyBatis σε ένα persistence πλαίσιο. Το Ebean χρησιμοποιεί τον ίδιο τρόπο αντιστοίχισης-μαρκαρίσματος με το JPA (@Entity, @Column, @OneToMany κλπ.). Η χαρτογράφηση των οντοτήτων μπορεί να είναι συμβατή μεταξύ του Ebean και του JPA. Πέρα από το JPA, το Ebean υποστηρίζει τα Java Generics και την προσκόμιση partial objects με το Query αντικείμενο.

Έτσι λοιπόν, ο προγραμματιστής δεν χρειάζεται να γράφει παραπάνω κώδικα για την διαχείριση των δεδομένων, αρκεί να έχει φτιάξει σωστά το μοντέλο του πόρου και να το έχει μαρκάρει καταλλήλως με τα ειδικά annotations.

```java
@Entity
public class Voter extends Model {

    @Id @GeneratedValue @NotNull
    public Long id;
    @Column(nullable = false)
    public String status;
    @Column(nullable = false)
    public String code;
    @Column(nullable = false)
    public String role;
    @Column(nullable = false)
    public String email;
    @Column(nullable = false) @Size(min = 5, max = 20)
    public String username;
    @Column
    public String passwordHash;
    @Column
    public String accessToken;
    @Column(nullable = false)
    public Instant timeUpdated;
    @Column(nullable = false)
    public Instant timeCreated;

    @OneToMany(mappedBy = "creator")
    public List<Poll> createdPolls;
    @OneToMany(mappedBy = "voter")
    public List<Vote> votedPolls;
    @OneToMany(mappedBy = "voter")
    public List<Participation> participatedPolls;
```

Εικόνα 3 Java Μοντέλο ψηφοφόρου (Voter.java) 1/2

Επίσης το ORM κάνει πιο ευδιάκριτο των κώδικα γιατί αντικαθιστά τον κώδικα SQL με μεθόδους της αντίστοιχης γλώσσας. Όπως φαίνεται και στην παραπάνω (βλ. εικόνα 3) για να δηλωθεί ο ψηφοφόρος σαν πίνακας στην βάση δεδομένων, πρέπει να μαρκαριστεί η κλάση με το “Entity” annotation. Για κάθε στήλη που αντιστοιχεί στον πίνακα, γίνεται η αντιστοίχιση της μεταβλητής με την χρήση του annotation “Column”. Εντός παρενθέσεων γίνεται να καθοριστούν κάποια στοιχεία για την στήλη όπως για παράδειγμα το αν παίρνουν null τιμές (nullable) ή το μέγεθος του στοιχείου (length) κ.α. Το μέγεθος μπορεί να καθοριστεί πιο συγκεκριμένα με την χρήση του annotation “Size” με το οποίο θέτονται τα άκρα (min/max) των τιμών της στήλης. Το κύριο κλειδί δηλώνεται με το annotation “Id” και με το “GeneratedValue” θεωρείται πως η βάση αναλαμβάνει την ανάθεση του αναγνωριστικού. Για την δήλωση των σχέσεων μεταξύ των υπόλοιπων πινάκων υπάρχουν τα annotations (“OneToOne”, ”OneToMany”, “ManyToMany”).

```
public static Finder<Long, Voter> find = new Finder<Long, Voter>(Voter.class);

public static Voter findVoterByCode(String code) {
    return find
            .where()
            .eq("code", code)
            .findUnique();
}

public static Voter findVoterByUsername(String username) {
    return find
            .where()
            .eq("username", username)
            .findUnique();
}

public static Voter findVoterByEmail(String email) {
    return find
            .where()
            .eq("email", email)
            .findUnique();
}
```

Εικόνα 4 Java Μοντέλο ψηφοφόρου (Voter.java) 2/2

Με αυτά επιτυγχάνεται η αντιστοίχιση οποιασδήποτε σχέσης, αλλά και ο καθορισμός της συμπεριφοράς κατά της επεξεργασία και την προσπέλαση με την ρύθμιση της ιδιότητας “cascading” εντός των παρενθέσεων. Με την χρήση της κλάσης Finder δημιουργεί κανείς λύματα σε java τα οποία μετατρέπονται σε sql από το πλαίσιο (βλ. εικόνα 4). Επομένως με την ανάκτηση δεδομένων από την βάση, συμπληρώνονται αντικείμενα java και οι επιμέρους μεταβλητές τους με τα στοιχεία από τις αντίστοιχες στήλες τα οποία είναι έτοιμα για να τα διαχειριστεί ο προγραμματιστής. Η δημιουργία και η επεξεργασία γραμμών σε κάποιο πίνακα γίνεται με την αντίστροφη διαδικασία, δηλαδή την δημιουργία ή επεξεργασία ενός java αντικειμένου και την χρήση της μεθόδου αποθήκευσης (save).

ΥΠΟΚΕΦΑΛΑΙΟ 2.3.3: Restful API

Το API της πλατφόρμας περιβάλλεται από τέσσερεις βασικές συλλογές πόρων:

- Users
- Polls

- Participations
- Votes

Για κάθε πόρο υποστηρίζονται κάποιες λειτουργίες οι οποίες φαίνονται στην εικόνα 5. Το σύστημα υποστηρίζει μόνο json αναπαράσταση αυτών των πόρων οπότε οι εφαρμογές πελάτες θα πρέπει να βάζουν τις ανάλογες κεφαλίδες (Content-Type, Accept). Κάποιοι πόροι όπως για παράδειγμα οι ψηφοφορίες, υποστηρίζουν pagination, δηλαδή την προσπέλαση ψηφοφοριών σε σελίδες συγκεκριμένου μεγέθους.

```
# Routes
# This file defines all application routes (Higher priority routes first)
# ~~~~

# Index
GET     /                           controllers.HomeController.index
# API documentation
GET     /documentation              controllers.Assets.at(path="/public", file = "html/documentation.html")
# User resource
POST    /users                      controllers.VotersController.createUser()
POST    /users/token                controllers.VotersController.authenticate()
DELETE  /users/token                controllers.VotersController.logout()
GET     /users/me                   controllers.VotersController.fetchUserSelf()
PATCH   /users/me                   controllers.VotersController.updateUserSelf()
GET     /users/:id                  controllers.VotersController.fetchUser(id:String)
PATCH   /users/:id                  controllers.VotersController.updateUser(id:String)
# Poll resource
POST    /polls                      controllers.PollsController.createPoll()
GET     /polls/public               controllers.PollsController.fetchPollsPublic(page:Int, size:Int)
GET     /polls/private              controllers.PollsController.fetchPollsPrivate(page:Int, size:Int)
GET     /polls/created              controllers.PollsController.fetchPollsCreated(page:Int, size:Int)
GET     /polls/:id                  controllers.PollsController.fetchPoll(id:String)
# Participation resource
POST    /participations             controllers.ParticipationsController.createParticipation()
GET     /participations/:id         controllers.ParticipationsController.fetchParticipation(id:String)
DELETE  /participations/:id         controllers.ParticipationsController.deleteParticipation(id:String)
# Voter resource
POST    /votes                      controllers.VotesController.createVote()
GET     /votes/:id                  controllers.VotesController.fetchVote(id:String)
```

Εικόνα 5 Δρομολόγηση αιτημάτων (routes)

Για την μορφή των απαντήσεων του εξυπηρετητή επιλέχθηκε το JSON-API specification, για μια πιο δομημένη ιεραρχία. Το JSNO-API δημιουργήθηκε με σκοπό να υπάρξει ένα γενικευμένο πρότυπο που να δουλεύει με την αρχιτεκτονική REST. Με αυτόν τον τρόπο δίνεται η ευκαιρία στους προγραμματιστές που είναι εξοικειωμένοι με το specification, να αναπτύσσουν τις εφαρμογές πελάτη πιο γρήγορα και εύκολα χωρίς να μαντεύουν την μορφή των απαντήσεων του εξυπηρετητή. Η απάντηση του εξυπηρετητή θα περιέχει ένα μήνυμα με την μορφή της εικόνας 6. Στην περίπτωση σφάλματος το μήνυμα θα περιέχει κάποιες βασικές πληροφορίες μέσα στο αντικείμενο “errors”.

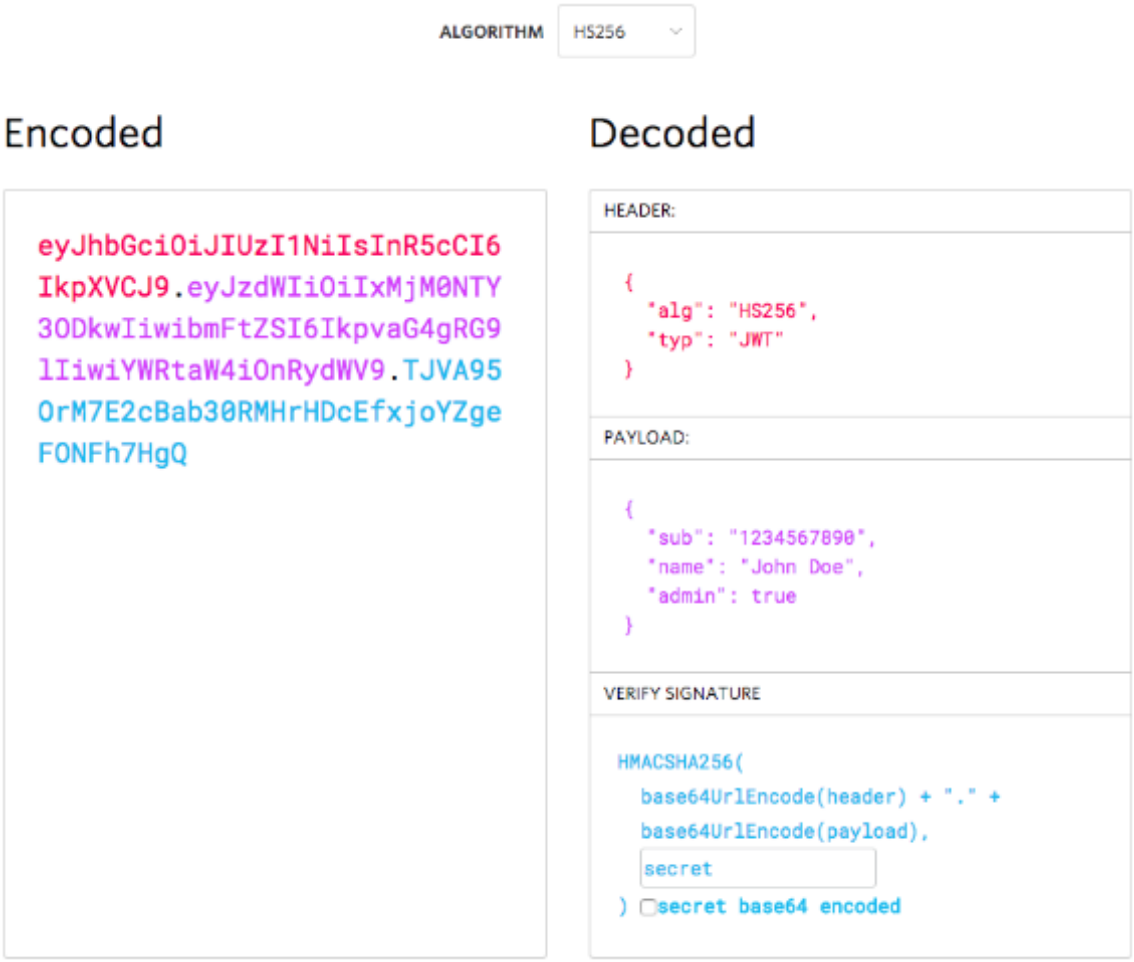
```
{
  "data": {
    "type": "articles",
    "id": "1",
    "attributes": {
      // ... this article's attributes
    },
    "relationships": {
      // ... this article's relationships
    }
  }
}
```

Εικόνα 6 Δομή σώματος JSON-API Specification

ΥΠΟΚΕΦΑΛΑΙΟ 2.3.4: Ασφάλεια

Για την ταυτοποίηση των χρηστών χρησιμοποιείται η υλοποίηση του OAuth2 password με την διανομή Json web tokens (JWT). Η διαδικασία κυλάει ως εξής:

- Ο χρήστης αρχικά κάνει εγγραφή στο σύστημα βάζοντας στο σώμα του αιτήματος email, username και έναν κωδικό.
- Ο εξυπηρετητής αποθηκεύει τα στοιχεία του χρήστη στην βάση δεδομένων. Εδώ πρέπει να τονίσουμε ότι ο κωδικός δεν αποθηκεύεται ως plain text, αλλά περνάει από την μέθοδο haspw της βιβλιοθήκης Bcrypt και μετατρέπεται σε ένα αλφαριθμητικό String το οποίο είναι ασφαλές ακόμα και αν κλαπεί η βάση δεδομένων.
- Ο χρήστης στέλνει ένα FormUrlEncoded POST αίτημα στο /users/token βάζοντας τρείς τιμές. Η πρώτη τιμή είναι ονομάζεται grant_type και πρέπει να είναι οπωσδήποτε “password” (οδηγίες του OAuth2) για να το αποδεχτεί ο εξυπηρετητής. Οι επόμενες δύο είναι το username και το password του χρήστη.
- Ο εξυπηρετητής ψάχνει τον χρήστη με βάση το ψευδώνυμο και περνάει από την hashpw τον κωδικό που έστειλε στο αίτημα για να συγκρίνει το αποτέλεσμα με αυτό που βρίσκεται στην βάση.
- Αν ο χρήστης ταυτοποιηθεί τότε ο εξυπηρετητής φτιάχνει ένα JWT με τα ανάλογα claims και το στέλνει στην απόκριση του.
- Για κάθε αίτημα το οποίο είναι προστατευμένο από guests, ο χρήστης πρέπει να βάζει στην κεφαλίδα Authorization του HTTP την τιμή “bearer <token string>”. Ο εξυπηρετητής για κάθε κλήση σε προστατευμένο πόρο, ελέγχει το JWT και συνεχίζει μόνο αν επικυρωθεί.



Εικόνα 7 Δομή JWT

Το JWT έχει την μορφή ενός μεγάλου αλφαριθμητικού χωρισμένου σε τρία μέρη από 2 τελείες (βλ. εικόνα 7). Και τα τρία μέρη είναι κωδικοποιημένα με base64Url. Το πρώτο μέρος αποτελεί την κεφαλίδα που περιέχει τις τιμές alg και typ οι οποίες καθορίζουν τον αλγόριθμο κατακερματισμού και τον τύπο του token π.χ. (“alg”: HS256, “typ”: “JWT”). Το δεύτερο κομμάτι αποτελεί την καρδιά του token και περιέχει τα claims δηλαδή κάποιες πληροφορίες όπως για παράδειγμα (issuer, name, role κτλ.). Υπάρχουν δημόσια claims τα οποία είναι ορισμένα στο IANA Web Token Registry, αλλά μπορούν να χρησιμοποιηθούν και custom, αρκεί να μην υπάρχουν συγκρούσεις με τα δημόσια. Το τρίτο κομμάτι είναι η υπογραφή signature των προηγούμενων 2 μερών κωδικοποιημένων με base64Url και ενός μυστικού κλειδιού που ξέρει μόνο ο εξυπηρετητής. Τίθεται το ερώτημα: Δεν θα μπορούσε κάποιος να αλλοιώσει το token αποκωδικοποιώντας το; Η απάντηση είναι πως αυτό δεν θα δουλέψει. Ο εξυπηρετητής όταν λάβει το token θα δημιουργήσει μια υπογραφή των δύο πρώτων μερών και θα την συγκρίνει με αυτή στο τρίτο μέρος. Ο επιτιθέμενος θα έπρεπε να γνωρίζει το κλειδί του εξυπηρετητή για να μπορεί να αλλάξει και την υπογραφή.

```
@With(SecuredAction.class)
public Result createPoll() {...}

@With(SecuredAction.class)
public Result fetchPoll(String id) {...}

@With(SecuredAction.class)
public Result fetchPollsPublic(int page, int size) {...}
```

Εικόνα 8 Ασφάλιση κλήσεων με την χρήση annotation

Στο σύστημα ο έλεγχος του token γίνεται στην κλάση SecuredAction.java. Το Play framework επιτρέπει την προσθήκη του annotation “With” πάνω από μεθόδους με αποτέλεσμα να εκτελείται πρώτα το δηλωθέν action πριν τρέξει η μαρκαρισμένη μέθοδος (βλ. εικόνα 8). Αυτό είναι ιδιαίτερα βολικό γιατί επιτρέπει στην ροή των γεγονότων να έχουν μεγάλη ευελιξία.

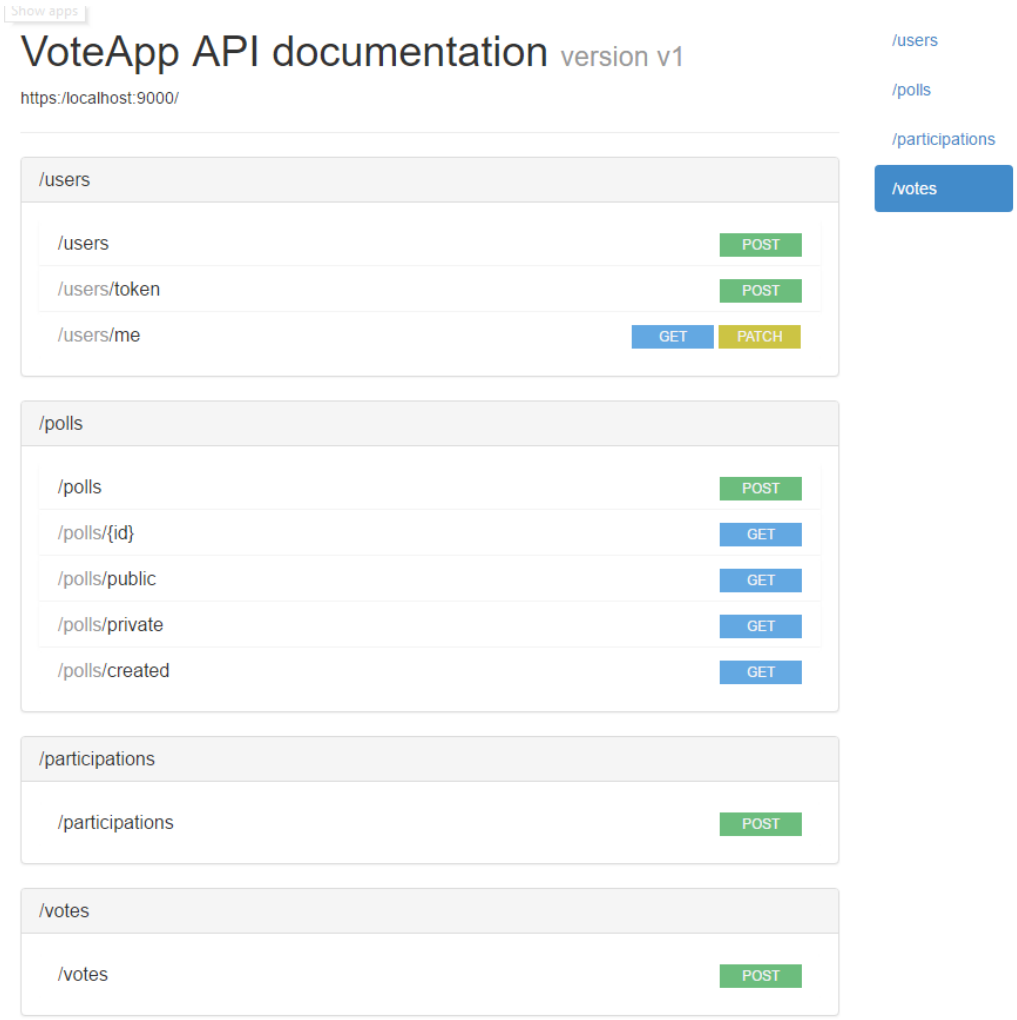
Ακόμα πρέπει να αναφερθεί η χρήση αλφαριθμητικών αναγνωριστικών για κάθε πόρο (βλ. στήλη code στο σχεσιακό διάγραμμα). Ο λόγος για τον οποίο γίνεται αυτό, είναι η απόκρυψη των διαδοχικών (sequential) αναγνωριστικών που αναθέτει η βάση. Γενικά δεν πρέπει ένας πελάτης (client) να βλέπει το διαδοχικό αναγνωριστικό καθώς γίνεται να εξάγει συμπεράσματα για τον αριθμό των οντοτήτων όπως για παράδειγμα ο αριθμός των χρηστών στην πλατφόρμα. Για το σκοπό αυτό, διαδεδομένη είναι η χρήση των UUIDs λόγω της μοναδικότητας τους. Εδώ όμως επιλέγεται η παραγωγή τυχαίων αλφαριθμητικών μεγέθους πέντε χαρακτήρων εκ των οποίων ο κάθε χαρακτήρας επιλέγεται τυχαία από το σύνολο του αγγλικού αλφάβητου και των δέκα ψηφίων. Αυτό επιτρέπει ένα σύνολο 5.4210109e+44 αναγνωριστικών. Παρόλα αυτά, πολλά από τα αναγνωριστικά συνήθως αφαιρούνται καθώς μπορεί να αποτελούν ύβρης ή να προκαλούν άλλα προβλήματα στην λειτουργία του συστήματος.

ΥΠΟΚΕΦΑΛΑΙΟ 2.4: Τεκμηρίωση

Η σωστή τεκμηρίωση ενός RESTful API αποτελεί τον κυριότερο σκοπό κατά την ανάπτυξη του. Χωρίς σωστή τεκμηρίωση, το έργο των προγραμματιστών εφαρμογών πελατών θα γίνει σαφέστατα δυσκολότερο αν όχι αδύνατο. Μια απλή καταγραφή για το κάθε endpoint του API δεν αρκεί. Χρειάζεται μια πλήρης λύση στην οποία να φαίνονται οι δυνατότητες του API και τα απαραίτητα στοιχεία κάθε αιτήματος. Για κάθε πόρο πρέπει να φαίνονται όλα τα διαθέσιμα μονοπάτια URI με όλες τις διαθέσιμες ενέργειες που μπορεί κανείς διαπράξει. Είναι απαραίτητο να φαίνονται για κάθε κλήση όλες οι πιθανές αποκρίσεις του εξυπηρετητή με παραδείγματα αυτών. Όλα αυτά πρέπει να είναι διαθέσιμα από ένα σημείο π.χ. μια σελίδα του συστήματος.

Για την επίτευξη αυτού του στόχου έγινε η χρήση του εργαλείου RAML. Το RAML (RESTful API Modeling Language) είναι ένας τρόπος περιγραφής RESTful APIs ο οποίος είναι αρκετά ευδιάκριτος και από ανθρώπους και από μηχανές. Το RAML δεν είναι αυστηρό. Προς το παρόν εστιάζει στην καθαρή περιγραφή πόρων, μεθόδων, παραμέτρων, απαντήσεων, τύπων μέσων και άλλων δομών HTTP που αποτελούν την βάση για σύγχρονα API που υπακούουν σε πολλούς, ίσως όχι όλους, τους περιορισμούς της αρχιτεκτονικής REST. Το RAML είναι ένα μη-αποκλειστικό, ουδέτερο από πωλητές, ανοικτό πρότυπο. Ο σκοπός του είναι να βοηθήσει το σημερινό οικοσύστημα API, να επιλύσει άμεσα προβλήματα και εν συνέχεια να ενθαρρύνει τα ολοένα και καλύτερα πρότυπα API.

Με την συγγραφή των αρχείων RAML και την χρήση μια δευτερεύουσας βιβλιοθήκης, παράγεται μια ιστοσελίδα σε μορφή html η οποία σερβίρεται από τον εξυπηρετητή στο URI (/documentation) και δίνει έναν ωραίο και καθαρό τρόπο για την περιγραφή του API (βλ. εικόνα 9).



Εικόνα 9 RAML API documentation

ΚΕΦΑΛΑΙΟ 3: Εφαρμογή ηλεκτρονικής ψηφοφορίας Android

ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο αυτό περιγράφεται η ανάπτυξη της εφαρμογής πελάτη και ο τρόπος λειτουργίας αυτής με την πλατφόρμα εξυπηρετητή που αναλύθηκε στο προηγούμενο κεφάλαιο.

ΥΠΟΚΕΦΑΛΑΙΟ 3.1: Android

ΥΠΟΚΕΦΑΛΑΙΟ 3.1.1: Βασική περιγραφή

Το Android είναι ένα λειτουργικό σύστημα κινητών συσκευών το οποίο αναπτύσσεται από την Google και είναι βασισμένο στον πυρήνα Linux. Έχει σχεδιαστεί κυρίως για συσκευές με οθόνη αφής όπως για παράδειγμα smartphones και tablets. Η διεπαφή του (user interface) βασίζεται στην άμεση μεταχείριση, με την χρήση χειρονομιών αφής που αντιστοιχούν σε πράξεις του πραγματικού κόσμου όπως swiping, tapping και pinching, αντικειμένων που εμφανίζονται στην οθόνη παράλληλα με το εικονικό πληκτρολόγιο για εισαγωγή κειμένου. Εκτός από τις συσκευές αφής, η Google έχει αναπτύξει κάποιες παραλλαγές του ίδιου λειτουργικού για την χρήση τους και σε άλλες περιπτώσεις χρήσης όπως τηλεοράσεις, αυτοκίνητα, ρολόγια χειρός με ειδικά διαμορφωμένη διεπαφή. Η χρήση του Android δεν αποκλείεται από φορητούς υπολογιστές, κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές και άλλα ηλεκτρονικά.

Αρχικά το λειτουργικό αναπτύχθηκε από την Android Inc., την οποία αγόρασε το 2005 η Google και στην συνέχεια κυκλοφόρησε το 2007 μαζί με την ίδρυση του Open Handset Alliance, μιας κοινοπραξίας εταιριών υλικού, λογισμικού και τηλεπικοινωνιών που αφιερώνεται στην προώθηση ανοικτών προτύπων για κινητές συσκευές. Ξεκινώντας με την πρώτη εμπορική συσκευή Android τον Σεπτέμβριο του 2008, το λειτουργικό σύστημα έχει περάσει από πολλές σημαντικές εκδόσεις, με την τρέχουσα έκδοση να είναι 7.0 "Nougat", η οποία κυκλοφόρησε τον Αύγουστο του 2016. Οι εφαρμογές Android μπορούν να αποκτηθούν από το Google Play το οποίο διαθέτει πάνω από 2,7 εκατομμύρια εφαρμογές από τον Φεβρουάριο του 2017. Το Android είναι το OS με τις καλύτερες πωλήσεις σε tablets από το 2013 και τρέχει στη συντριπτική πλειοψηφία των smartphones. Από τον Μάιο του 2017, το Android έχει ενεργούς χρήστες δύο δισεκατομμυρίων μηνών και έχει τη μεγαλύτερη εγκατεστημένη βάση οποιουδήποτε λειτουργικού συστήματος.

Ο πηγαίος κώδικας του Android κυκλοφορεί από την Google με άδεια ανοικτού κώδικα, αν και οι περισσότερες συσκευές Android κυκλοφορούν τελικά με ένα συνδυασμό ελεύθερου και ανοικτού κώδικα λογισμικού και ιδιόκτητου λογισμικού, συμπεριλαμβανομένου του ιδιόκτητου λογισμικού που απαιτείται για την πρόσβαση στις υπηρεσίες Google. Το Android είναι δημοφιλές σε εταιρείες τεχνολογίας που απαιτούν ένα έτοιμο, χαμηλού κόστους και προσαρμόσιμο λειτουργικό σύστημα για συσκευές υψηλής τεχνολογίας. Ο ανοιχτός χαρακτήρας της ενθάρρυνε μια μεγάλη κοινότητα προγραμματιστών να χρησιμοποιήσουν τον κώδικα ανοιχτού κώδικα ως θεμέλιο για κοινοτικά προγράμματα, τα οποία παρέχουν ενημερώσεις σε παλαιότερες συσκευές, προσθέτουν νέες δυνατότητες για προχωρημένους χρήστες ή φέρνουν το Android σε συσκευές που αποστέλλονται αρχικά με άλλα λειτουργικά συστήματα. Η εκτεταμένη ποικιλία του υλικού σε συσκευές Android προκαλεί σημαντικές καθυστερήσεις στις αναβαθμίσεις λογισμικού, ενώ οι νέες εκδόσεις του λειτουργικού συστήματος και τα μπαλώματα ασφαλείας συνήθως χρειάζονται μήνες πριν φτάσουν στους καταναλωτές. Η επιτυχία του Android έχει καταστήσει στόχο την επίλυση των διπλωμάτων ευρεσιτεχνίας και των δικαιωμάτων πνευματικής ιδιοκτησίας στο πλαίσιο των αποκαλούμενων «πολέμων smartphone» μεταξύ των εταιρειών τεχνολογίας.

ΥΠΟΚΕΦΑΛΑΙΟ 3.1.2: Αρχιτεκτονική

Ο πυρήνας του Linux: Η βάση της πλατφόρμας Android είναι ο πυρήνας του Linux. Για παράδειγμα, το Runtime του Android (ART) βασίζεται στον πυρήνα του Linux για τις υποκείμενες λειτουργίες, όπως η διαχείριση της μνήμης και η διαχείριση χαμηλής μνήμης. Η χρήση του πυρήνα Linux επιτρέπει στο Android να εκμεταλλευτεί τα βασικά χαρακτηριστικά ασφάλειας και επιτρέπει στους κατασκευαστές συσκευών να αναπτύξουν προγράμματα οδήγησης υλικού για έναν γνωστό πυρήνα.

Hardware Abstraction Layer (HAL): Το επίπεδο αφαίρεσης υλικού (HAL) παρέχει τυποποιημένες διασυνδέσεις που εκθέτουν τις δυνατότητες υλικού συσκευών στο πλαίσιο API ανώτερου επιπέδου Java. Το HAL αποτελείται από πολλαπλές ενότητες βιβλιοθήκης, καθένα από τα οποία υλοποιεί μια διεπαφή για έναν συγκεκριμένο τύπο υλικού, όπως η κάμερα ή η μονάδα bluetooth. Όταν ένα API πλαισίου κάνει μια κλήση για πρόσβαση στο υλικό της συσκευής, το σύστημα Android φορτώνει τη λειτουργική μονάδα της βιβλιοθήκης για το εν λόγω στοιχείο υλικού.

Runtime Android: Για συσκευές με έκδοση Android 5.0 (επίπεδο 21 API) ή υψηλότερη, κάθε εφαρμογή εκτελείται με τη δική της διεργασία και με τη δική της παρουσία του Android Runtime (ART). Το ART γράφτηκε για την εκτέλεση πολλαπλών εικονικών μηχανών σε συσκευές χαμηλής μνήμης, εκτελώντας αρχεία

DEX, μια μορφή bytecode ειδικά σχεδιασμένη για το Android που έχει βελτιστοποιηθεί για ελάχιστη κατανάλωση μνήμης. Μερικά από τα κύρια χαρακτηριστικά της ART περιλαμβάνουν τα εξής:

- Προηγούμενη (AOT) και just-in-time (JIT) μεταγλώττιση
- Βελτιστοποιημένη συλλογή απορριμμάτων (GC)
- Καλύτερη υποστήριξη εντοπισμού σφαλμάτων, συμπεριλαμβανομένου ενός ειδικού προφίλ δειγματοληψίας, λεπτομερείς διαγνωστικές εξαιρέσεις και αναφορές σφαλμάτων και η δυνατότητα καθορισμού σημείων παρακολούθησης για την παρακολούθηση συγκεκριμένων πεδίων(debugging)

Πριν από την έκδοση Android 5.0 (επίπεδο 21 API), το Dalvik ήταν το runtime του Android. Εάν μια εφαρμογή τρέχει καλά στην ART, τότε θα πρέπει να δουλέψει και στον Dalvik, αλλά το αντίστροφο μπορεί να μην είναι αλήθεια. Το Android περιλαμβάνει επίσης ένα σύνολο βασικών βιβλιοθηκών χρόνου εκτέλεσης που παρέχουν τις περισσότερες λειτουργίες της γλώσσας προγραμματισμού Java, συμπεριλαμβανομένων ορισμένων λειτουργιών γλώσσας Java 8, τις οποίες χρησιμοποιεί το πλαίσιο API Java.

Native βιβλιοθήκες C / C ++: Πολλά βασικά στοιχεία και υπηρεσίες του συστήματος Android, όπως το ART και το HAL, είναι χτισμένα από εγγενή κώδικα που απαιτούν εγγενείς βιβλιοθήκες γραμμένες σε C και C ++. Η πλατφόρμα Android παρέχει API πλαισίου Java για να αποκαλύψει τις λειτουργίες ορισμένων από αυτές τις μητρικές βιβλιοθήκες σε εφαρμογές. Για παράδειγμα, παρέχεται πρόσβαση στο OpenGL ES μέσω του Java OpenGL API του πλαισίου Android για την υποστήριξη για την σχεδίαση και τον χειρισμό γραφικών 2D και 3D στην εφαρμογή. Σε περίπτωση ανάπτυξης μιας εφαρμογής που απαιτεί κώδικα C ή C ++, μπορεί να χρησιμοποιηθεί το Android NDK για την απόκτηση πρόσβασης σε ορισμένες από αυτές τις βιβλιοθήκες απευθείας από τον εγγενή κώδικα.

Java API Framework: Το σύνολο των χαρακτηριστικών του Android OS είναι διαθέσιμο μέσω των API που είναι γραμμένα στη γλώσσα Java. Αυτά τα API αποτελούν τις δομικές μονάδες για να δημιουργήσει κανείς εφαρμογές Android απλοποιώντας την επαναχρησιμοποίηση βασικών, αρθρωτών εξαρτημάτων και υπηρεσιών του συστήματος, τα οποία περιλαμβάνουν τα εξής:

- Ένα πλούσιο και επεκτάσιμο σύστημα προβολής με το οποίο μπορεί κανείς να χρησιμοποιήσει για να δημιουργήσει ένα περιβάλλον χρήστη της εφαρμογής, συμπεριλαμβανομένων λιστών, πλεγμάτων, πλαισίων κειμένου, κουμπιών και ακόμη και ενός ενσωματωμένου προγράμματος περιήγησης ιστού
- Έναν διαχειριστή πόρων, που παρέχει πρόσβαση σε πόρους, όπως μεταφρασμένο κείμενο, γραφικά και αρχεία διαμόρφωσης

- Έναν διαχειριστή ειδοποιήσεων που επιτρέπει σε όλες τις εφαρμογές να εμφανίζουν προσαρμοσμένες ειδοποιήσεις στη γραμμή κατάστασης
- Έναν διαχειριστή δραστηριοτήτων που διαχειρίζεται τον κύκλο ζωής των εφαρμογών και παρέχει μια κοινή στοίβα πλοήγησης
- Παροχείς περιεχομένου που επιτρέπουν σε εφαρμογές να έχουν πρόσβαση σε δεδομένα από άλλες εφαρμογές, όπως η εφαρμογή "Επαφές", ή να μοιράζονται τα δικά τους δεδομένα

Οι προγραμματιστές έχουν πλήρη πρόσβαση στα ίδια API πλαισίου που χρησιμοποιούν οι εφαρμογές του συστήματος Android.

Εφαρμογές συστήματος: Το Android περιλαμβάνει ένα σύνολο βασικών εφαρμογών για μηνύματα ηλεκτρονικού ταχυδρομείου, μηνύματα SMS, ημερολόγια, περιήγηση στο Internet, επαφές και πολλά άλλα. Οι εφαρμογές που περιλαμβάνονται στην πλατφόρμα δεν έχουν ιδιαίτερη κατάσταση μεταξύ των εφαρμογών που επιλέγει ο χρήστης για εγκατάσταση. Έτσι, μια third-party εφαρμογή μπορεί να γίνει το προεπιλεγμένο πρόγραμμα περιήγησης ιστού, το SMS messenger ή ακόμα και το προεπιλεγμένο πληκτρολόγιο (ισχύουν ορισμένες εξαιρέσεις, όπως η εφαρμογή ρυθμίσεων του συστήματος). Οι εφαρμογές του συστήματος λειτουργούν ως εφαρμογές για χρήστες και παρέχουν βασικές δυνατότητες που μπορούν να έχουν πρόσβαση οι προγραμματιστές από τη δική τους εφαρμογή. Για παράδειγμα, εάν μια εφαρμογή θα ήθελε να παραδώσει ένα μήνυμα SMS, δεν χρειάζεται να κατασκευαστεί η συγκεκριμένη λειτουργικότητα. Μπορεί να επικαλεστεί όποια εφαρμογή SMS έχει ήδη εγκατασταθεί για να αποσταλθεί ένα μήνυμα στον παραλήπτη.

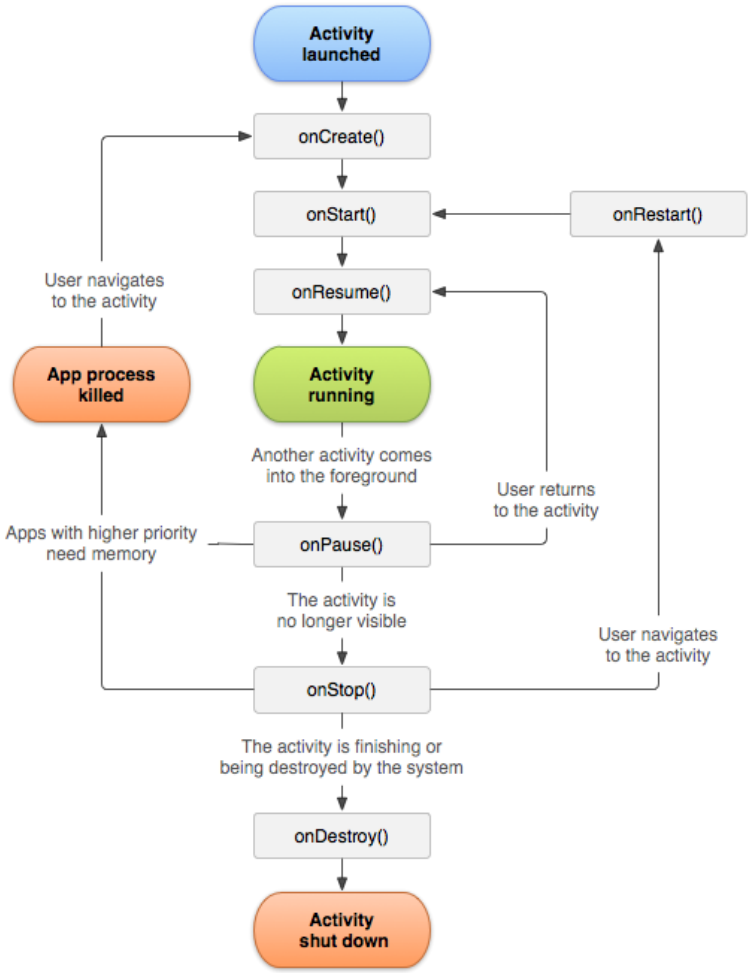
ΥΠΟΚΕΦΑΛΑΙΟ 3.1.3: Ο κύκλος ζωής ενός Activity

Κατά την περιήγηση του χρήστη σε μια εφαρμογή (είσοδος, έξοδος), οι δραστηριότητες (Activities) περνάνε από διάφορες καταστάσεις του κύκλου ζωής τους. Η κλάση Activity παρέχει μια σειρά callbacks που επιτρέπουν στη δραστηριότητα να γνωρίζει ότι έχει αλλάξει μια κατάσταση όπως για παράδειγμα ότι το σύστημα δημιουργεί. Σταματάει ή συνεχίζει μια δραστηριότητα ή ότι καταστρέφει την διεργασία στην οποία βρίσκεται η δραστηριότητα. Μέσα στις μεθόδους επανάκλησης (callbacks) του κύκλου ζωής μπορεί να καθοριστεί η συμπεριφορά της δραστηριότητας όταν ο χρήστης εγκαταλείπει και επανεισάγει τη δραστηριότητα. Για παράδειγμα, στην περίπτωση μια εφαρμογής αναπαραγωγής βίντεο συνεχούς ροής, ενδέχεται να διακοπεί το βίντεο και να τερματιστεί η σύνδεση δικτύου όταν ο χρήστης μεταβεί σε άλλη εφαρμογή. Όταν ο χρήστης επιστρέψει, μπορεί ο χρήστης να επανασυνδεθεί στο δίκτυο και να του επιτραπεί να συνεχίσει το βίντεο από το ίδιο σημείο. Με άλλα λόγια, κάθε επανάκληση επιτρέπει την εκτέλεση συγκεκριμένης εργασίας που είναι κατάλληλη για μια

δεδομένη αλλαγή στην κατάσταση της εφαρμογής. Κάνοντας τις κατάλληλες εργασίες στις κατάλληλες επανακλήσεις παράλληλα με την σωστή διαχείριση των μεταβάσεων, η εφαρμογή γίνεται πιο ισχυρή και αποτελεσματική. Για παράδειγμα, η καλή υλοποίηση των επανακλήσεων του κύκλου ζωής μιας δραστηριότητες μπορεί να βοηθήσει στην αποφυγή:

- Της κατάρρευσης της εφαρμογής στην περίπτωση που ο χρήστης λάβει μια τηλεφωνική κλήση ή αλλάξει σε άλλη εφαρμογή κατά την διάρκεια χρήσης.
- Της κατανάλωσης πολύτιμων πόρων συστήματος όταν ο χρήστης δεν χρησιμοποιεί την εφαρμογή ενεργά.
- Της απώλειας προόδου του χρήστη, εάν εγκαταλείψει την εφαρμογή και επιστρέψει σε αυτήν αργότερα.
- Της κατάρρευσης ή της απώλειας της προόδου του χρήστη, όταν η οθόνη περιστρέφεται μεταξύ οριζόντιου και κατακόρυφου προσανατολισμού.

Για την περιήγηση μεταξύ των μεταβάσεων κατάστασης του κύκλου ζωής, η κλάση Activity παρέχει ένα σύνολο έξι επανακλήσεων: onCreate, onStart, onResume, onPause, onStop, και onDestroy (βλ. εικόνα 10). Το σύστημα επικαλείται κάθε μια από αυτές τις επανακλήσεις κάθε φορά που μια δραστηριότητα εισέρχεται σε μια νέα κατάσταση. Καθώς ο χρήστης αρχίζει να αφήνει τη δραστηριότητα, το σύστημα καλεί μεθόδους για να αποσυναρμολογήσει τη δραστηριότητα. Σε ορισμένες περιπτώσεις, αυτή η αποσυναρμολόγηση είναι μόνο μερική. Η δραστηριότητα εξακολουθεί να βρίσκεται στη μνήμη (όπως όταν ο χρήστης μεταβαίνει σε άλλη εφαρμογή) και μπορεί να επανέλθει στο προσκήνιο.



Εικόνα 10 Κύκλος ζωής ενός Activity

Εάν ο χρήστης επιστρέψει σε αυτή τη δραστηριότητα, αυτή συνεχίζεται από εκεί που ο χρήστης έπαψε να λειτουργεί. Η πιθανότητα του συστήματος να σκοτώσει μια δεδομένη διαδικασία, μαζί με τις δραστηριότητες σε αυτήν, εξαρτάται από την κατάσταση της δραστηριότητας εκείνη τη στιγμή. Η κατάσταση δραστηριότητας και η έξοδος από τη μνήμη παρέχουν περισσότερες πληροφορίες σχετικά με τη σχέση μεταξύ της κατάστασης και της ευπάθειας στην έξοδο. Ανάλογα με την πολυπλοκότητα της δραστηριότητας που αναπτύσσεται, πιθανώς να μη χρειαστεί να υλοποιηθούν όλες οι μέθοδοι του κύκλου ζωής της. Ωστόσο, είναι σημαντικό να κατανοηθούν όλες οι επανακλήσεις και να εφαρμόζονται οι απαραίτητες ώστε να διασφαλίζεται η αναμενόμενη εμπειρία χρήσης. Παρακάτω παρουσιάζονται οι μέθοδοι επανακλήσεων πιο αναλυτικά:

1. **onCreate():** Είναι η βασική επανάκληση που ενεργοποιείται όταν το σύστημα δημιουργήσει για πρώτη φορά τη δραστηριότητα και πρέπει πάντα να υλοποιείται. Κατά την δημιουργία, η δραστηριότητα εισέρχεται στην κατάσταση κατασκευής. Σε αυτή τη μέθοδο εκτελούνται εργασίες εκκίνησης οι οποίες πραγματοποιούνται μόνο μια φορά κατά την διάρκεια ζωής μιας δραστηριότητας. Για παράδειγμα, η μέθοδος onCreate μπορεί να συνδέσει δεδομένα σε λίστες, να προετοιμάσει τα νήματα φόντου και να δημιουργήσει

παραστάσεις σε ορισμένες μεταβλητές κλάσης. Αυτή η μέθοδος λαμβάνει την παράμετρο savedInstanceState, η οποία είναι ένα αντικείμενο τύπου Bundle που περιέχει την προηγούμενη αποθηκευμένη κατάσταση της δραστηριότητας. Εάν η δραστηριότητα δεν υπήρχε ποτέ πριν, η τιμή του αντικειμένου Bundle είναι μηδενική. Αφού ολοκληρωθεί η εκτέλεση της μεθόδου onCreate, η δραστηριότητα εισέρχεται στην κατάσταση έναρξης και το σύστημα καλεί τις μεθόδους onStart και onResume γρήγορα.

2. **onStart():** Όταν η δραστηριότητα εισέλθει στην κατάσταση Έναρξης, το σύστημα ενεργοποιεί αυτή την επανάκληση. Η κλήση onStart καθιστά τη δραστηριότητα ορατή στον χρήστη, καθώς η εφαρμογή προετοιμάζει τη δραστηριότητα να εισέλθει στο προσκήνιο και να γίνει διαδραστική. Για παράδειγμα, εδώ η εφαρμογή αρχικοποιεί τον κώδικα που διατηρεί το περιβάλλον χρήστη. Μπορεί επίσης να καταχωρίσει ένα BroadcastReceiver που παρακολουθεί τις αλλαγές που αντικατοπτρίζονται στο UI. Η μέθοδος onStart ολοκληρώνεται πολύ γρήγορα και, όπως συμβαίνει με τη κατάσταση κατασκευής, η δραστηριότητα δεν παραμένει στην κατάσταση έναρξης. Μόλις τελειώσει αυτή η επανάκληση, η δραστηριότητα εισέρχεται στην κατάσταση Resume και το σύστημα ενεργοποιεί τη μέθοδο onResume.

3. **onResume():** Όταν η δραστηριότητα εισέλθει στην κατάσταση συνέχειας, έρχεται στο προσκήνιο και τότε το σύστημα ενεργοποιεί την onResume επανάκληση. Εδώ η εφαρμογή αλληλεπιδρά με τον χρήστη. Η εφαρμογή παραμένει σε αυτήν την κατάσταση μέχρι να συμβεί κάτι που θα «κλέψει» την εστίαση από την εφαρμογή. Ένα τέτοιο συμβάν μπορεί να είναι, για παράδειγμα, μια εισερχόμενη τηλεφωνική κλήση, η πλοήγηση του χρήστη σε μια άλλη δραστηριότητα ή η απενεργοποίηση της οθόνης της συσκευής. Όταν συμβαίνει ένα τέτοιο συμβάν, η δραστηριότητα εισέρχεται στην κατάσταση παύσης και το σύστημα ενεργοποιεί την επανάκληση onPause. Αν η δραστηριότητα επιστρέψει στην κατάσταση Resume από την κατάσταση παύσης, το σύστημα εκ νέου ενεργοποιεί τη μέθοδο Resume. Για το λόγο αυτό, θα πρέπει να εφαρμόζεται η onResume για την προετοιμασία των στοιχείων που απελευθερώνονται κατά τη διάρκεια της onPause. Το σύστημα καλεί αυτή τη μέθοδο κάθε φορά που η δραστηριότητά έρχεται στο προσκήνιο, ακόμη και όταν δημιουργείται για πρώτη φορά. Ως εκ τούτου, θα πρέπει να εφαρμόζεται η onResume για την προετοιμασία των στοιχείων που απελευθερώνονται κατά τη διάρκεια της onPause και να εκτελούνται οποιεσδήποτε άλλες αρχικοποιήσεις που πρέπει να γίνουν κάθε φορά που η δραστηριότητα εισέρχεται στην κατάσταση Resumed.

4. **onPause():** Το σύστημα καλεί αυτή τη μέθοδο ως την πρώτη ένδειξη ότι ο χρήστης εγκαταλείπει τη δραστηριότητά (αν και δεν σημαίνει πάντοτε ότι η

δραστηριότητα καταστρέφεται). Χρησιμοποιείται η μέθοδος onPause για να διακοπούν λειτουργίες όπως κινούμενες εικόνες και αναπαραγωγή μουσικής που δεν θα πρέπει να συνεχιστούν ενώ η δραστηριότητα βρίσκεται σε κατάσταση παύσης και αναμένονται να ξαναρχίσουν σύντομα. Υπάρχουν διάφοροι λόγοι για τους οποίους μια δραστηριότητα μπορεί να εισέλθει σε αυτήν την κατάσταση. Για παράδειγμα:

- Ορισμένο συμβάν διακόπτει την εκτέλεση της εφαρμογής, όπως περιγράφεται στην ενότητα onResume. Αυτή είναι η πιο συνηθισμένη περίπτωση.
- Στο Android 7.0 (επίπεδο API 24) ή υψηλότερο, πολλαπλές εφαρμογές εκτελούνται σε λειτουργία πολλαπλών παραθύρων. Επειδή μόνο μία από τις εφαρμογές (παράθυρα) έχει εστίαση ανά πάσα στιγμή, το σύστημα θέτει σε παύση όλες τις άλλες εφαρμογές.
- Μια νέα, ημι-διαφανής δραστηριότητα (όπως ένας dialog) ανοίγει. Όσο η δραστηριότητα εξακολουθεί να είναι μερικώς ορατή αλλά όχι εστιασμένη, παραμένει σε παύση.

Η μέθοδος onPause μπορεί να χρησιμοποιηθεί για την απελευθέρωση πόρων συστήματος, όπως δέκτες ραδιοφώνου, χειριστήρια σε αισθητήρες (όπως το GPS) ή πόρων που μπορεί να επηρεάσουν τη διάρκεια ζωής της μπαταρίας ενώ η δραστηριότητά είναι σε παύση και ο χρήστης δεν την χρειάζεται. Η εκτέλεση της onPause είναι πολύ σύντομη και δεν παρέχει απαραίτητα αρκετό χρόνο για την εκτέλεση εργασιών εξοικονόμησης. Για το λόγο αυτό, δεν πρέπει να χρησιμοποιείται η onPause για την αποθήκευση δεδομένων εφαρμογών ή χρηστών, την πραγματοποίηση κλήσεων δικτύου ή την εκτέλεση συναλλαγών βάσεων δεδομένων. Τέτοιες εργασίες μπορεί να μην προλάβουν να ολοκληρωθούν πριν ολοκληρωθεί η μέθοδος. Αντίθετα, αυτές θα πρέπει να εκτελεστούν κατά τη διάρκεια της λειτουργίας της onStop. Η ολοκλήρωση της μεθόδου onPause δεν σημαίνει ότι η δραστηριότητα εγκαταλείπει την κατάσταση παύσης. Αντίθετα, η δραστηριότητα παραμένει σε αυτήν την κατάσταση έως ότου η δραστηριότητα επαναληφθεί ή γίνει εντελώς αόρατη για τον χρήστη. Αν η δραστηριότητα ξαναρχίσει, το σύστημα επαναλαμβάνει ξανά την επανάκληση onResume. Εάν η δραστηριότητα επιστρέφει από την κατάσταση παύσης στην κατάσταση συνέχειας, το σύστημα διατηρεί την εμφάνιση της δραστηριότητας στη μνήμη, υπενθυμίζοντας το στιγμιότυπο όταν το σύστημα ενεργοποιεί την Resume. Σε αυτό το σενάριο, δεν χρειάζεται η εκ νέου προετοιμασία στοιχείων που δημιουργήθηκαν κατά τη διάρκεια οποιασδήποτε από τις μεθόδους επανάκλησης που οδήγησαν στην κατάσταση Resume. Εάν η δραστηριότητα καθίσταται εντελώς αόρατη, το σύστημα καλεί την onStop.

5. **onStop():** Όταν η δραστηριότητά δεν είναι πλέον ορατή από το χρήστη, έχει τεθεί σε κατάσταση διακοπής και το σύστημα ενεργοποιεί την επανάκληση onStop. Αυτό μπορεί να συμβεί, για παράδειγμα, όταν μια νέα

δραστηριότητα που ξεκίνησε καλύπτει ολόκληρη την οθόνη. Το σύστημα μπορεί επίσης να καλέσει την onStop όταν η δραστηριότητα έχει τελειώσει να τρέχει και πρόκειται να τερματιστεί. Στη μέθοδο onStop, η εφαρμογή θα πρέπει να απελευθερώσει σχεδόν όλους τους πόρους που δεν χρειάζονται, όσο ο χρήστης δεν τους χρησιμοποιεί. Για παράδειγμα, εάν έχει καταχωρηθεί ένας BroadcastReceiver στην onStart για να ακούγονται οι αλλαγές που ενδέχεται να επηρεάσουν το περιβάλλον χρήστη, μπορεί να καταργηθεί η εγγραφή του δέκτη εκπομπής στην onStop, καθώς ο χρήστης δεν βλέπει το περιβάλλον χρήστη. Είναι επίσης σημαντικό να χρησιμοποιείται η onStop για να απελευθερώνονται πόροι που ενδέχεται να διαρρεύσουν τη μνήμη, επειδή είναι δυνατό για το σύστημα να σκοτώσει τη διαδικασία που φιλοξενεί τη δραστηριότητά χωρίς να καλέσει την τελική μέθοδο τερματισμού onDestroy της δραστηριότητας. Επίσης η onStop πρέπει να χρησιμοποιείται για την εκτέλεση λειτουργιών έντονης χρήσης της CPU. Για παράδειγμα, εάν δεν βρίσκεται καταλληλότερη περίοδος για την αποθήκευση πληροφοριών σε μια βάση δεδομένων, αυτή μπορεί να γίνει κατά τη διάρκεια της λειτουργίας onStop. Όταν η δραστηριότητά, μεταβαίνει στην κατάσταση διακοπής, το αντικείμενο της διατηρείται μόνιμα στη μνήμη: Διατηρεί όλες τις πληροφορίες κατάστασης και μελών, αλλά δεν συνδέεται με τον διαχειριστή παραθύρων. Όταν η δραστηριότητα ξαναρχίσει, η δραστηριότητα ανακτά αυτές τις πληροφορίες. Δεν χρειάζεται να επαναληφθεί η προετοιμασία των στοιχείων που δημιουργήθηκαν κατά τη διάρκεια οποιασδήποτε από τις μεθόδους επανάκλησης που οδήγησαν στην κατάσταση Resume. Το σύστημα παρακολουθεί επίσης την τρέχουσα κατάσταση για κάθε αντικείμενο View στη διάταξη, οπότε αν ο χρήστης εισάγει κείμενο σε ένα γραφικό στοιχείο EditText, αυτό το περιεχόμενο διατηρείται, ώστε να μην χρειαστεί η αποθήκευση και η επαναφορά του. Από την κατάσταση διακοπής, η δραστηριότητα είτε επανέρχεται για να αλληλοεπιδράσει με το χρήστη, είτε η δραστηριότητα τελείωσε και εκτυλίσσεται. Αν η δραστηριότητα επανέλθει, το σύστημα ενεργοποιεί την onRestart. Εάν η δραστηριότητα ολοκληρώσει την εκτέλεση της, το σύστημα καλεί την onDestroy.

6. **onDestroy():** Καλείται πριν καταστραφεί η δραστηριότητα. Αυτή είναι η τελευταία κλήση που λαμβάνει η δραστηριότητα. Το σύστημα είτε καλεί αυτή την επανάκληση επειδή η δραστηριότητα τελειώνει εξαιτίας του τερματισμού κλήσης της μεθόδου finish, είτε επειδή το σύστημα καταστρέφει προσωρινά τη διαδικασία που περιέχει τη δραστηριότητα για εξοικονόμηση χώρου. Μπορεί να ξεχωρίσει κανείς τις δύο αυτές περιπτώσεις με τη χρήση της μεθόδου isFinishing. Το σύστημα μπορεί επίσης να καλέσει αυτήν τη μέθοδο όταν εμφανιστεί μια αλλαγή προσανατολισμού και στη συνέχεια να καλέσει αμέσως την onCreate για να αναδημιουργήσει τη διαδικασία (και τα στοιχεία που περιέχει) με τον νέο προσανατολισμό. Η κλήση της onDestroy απελευθερώνει όλους τους

πόρους που δεν έχουν ακόμη κυκλοφορήσει από προηγούμενες επανακλήσεις, όπως η onStop.

ΥΠΟΚΕΦΑΛΑΙΟ 3.2: Απαιτήσεις

Η εφαρμογή πελάτη θα εκμεταλλευτεί την διεπαφή της ηλεκτρονικής πλατφόρμας που αναπτύχθηκε και αναλύθηκε στο προηγούμενο κεφάλαιο και θα πρέπει να καλύπτει τις παρακάτω βασικές απαιτήσεις για να δίνει στον χρήστη την δυνατότητα μιας απλής και ασφαλής διαδικασίας ψηφοφοριών:

1) Εγγραφή χρήστη στο σύστημα ψηφοφορίας.
2) Ταυτοποίηση χρήστη με την πλατφόρμα-εξυπηρετητή ηλεκτρονικής ψηφοφορίας.
3) Δυνατότητα δημιουργίας δημόσιων και ιδιωτικών ψηφοφοριών και καθορισμός τύπου και διάρκειας αυτών.
4) Δυνατότητα πρόσκλησης άλλων χρηστών από τον δημιουργό μιας ψηφοφορίας.
5) Δυνατότητα δημιουργίας λίστας φίλων (friend list) για την διευκόλυνση της απαίτησης υπ' αριθμό 3.
6) Δυνατότητα ψηφοφορίας single και multiple choice (ανάλογα με το τι έχει καθορίσει ο δημιουργός).
7) Εμφάνιση αποτελεσμάτων ψηφοφορίας με μορφή διαγράμματος πίτας.

Για να γίνει η εφαρμογή συμβατή με όσο το δυνατό περισσότερες συσκευές android επιλέγεται minSdkVersion 17 και targetSdkVersion 23. Έτσι καλύπτουμε τουλάχιστον το 70% των συσκευών σύμφωνα με τα στατιστικά της Google.

ΥΠΟΚΕΦΑΛΑΙΟ 3.3: Υλοποίηση

ΥΠΟΚΕΦΑΛΑΙΟ 3.3.1: Δικτύωση

Για την κάλυψη των αναγκών δικτύωσης της εφαρμογής γίνεται η χρήση της βιβλιοθήκης Retrofit σε συνδυασμό με την βιβλιοθήκη jsonapi-converter για την επικοινωνία με τον εξυπηρετητή. Η βιβλιοθήκη Retrofit αναλαμβάνει τον περισσότερο φόρτο και δίνει την ευκαιρία στον προγραμματιστή να ορίσει το είδος των αιτημάτων που θέλει να κάνει στον εξυπηρετητή με την δήλωση μιας διεπαφής interface. Με την χρήση annotations μπορεί ο προγραμματιστής να δηλώσει τον τύπο του αιτήματος (GET, POST, PUT, DELETE) μαζί με το μονοπάτι url, τις κεφαλίδες του π.χ. (Content-type, Accept-Charset), και το σώμα, δηλαδή το περιεχόμενο το οποίο μπορεί να είναι κάποιου είδους αντικείμενο το οποίο

μετατρέπεται αυτόματα από τον αντάπτορα που έχουμε δηλώσει στην κλάση (βλ. εικόνα 11). Ακόμα διαλέγει και τον τύπο αντικειμένου στον οποίο θα αποσυναρμολογηθεί η απάντηση του εξυπηρετητή. Όλα αυτά δίνουν τεράστια ευελιξία κατά την ανάπτυξη καθώς χρειάζεται απλώς:

- Να δημιουργηθεί η διεπαφή και να δηλωθούν οι διαφορετικές κλήσεις προς τον εξυπηρετητή.
- Να μαρκαριστούν με ειδικά annotations τα στοιχεία που θέλουμε να μεταφέρονται και να συμπληρώνονται κατά την επικοινωνία στις κλάσεις των μοντέλων.
- Να αρχικοποιηθεί κάπου η διεπαφή ώστε να μπορεί να χρησιμοποιηθεί (στην περίπτωση μας η κλάση NetworkService)

```
@Headers({"Content-Type: application/json"})
@POST("users")
Observable<Response<JSONAPIDocument<User>>> submitSignupObservable(
        @Body JSONAPIDocument<User> requestBody);

@Headers({"Content-Type: application/json"})
@GET("users/{id}")
Observable<Response<JSONAPIDocument<User>>> fetchMyUserProfile(
        @Path("id") String id,
        @Body JSONAPIDocument<User> requestBody);
```

Εικόνα 11 Διεπαφή δικτύωσης (NetworkApi)

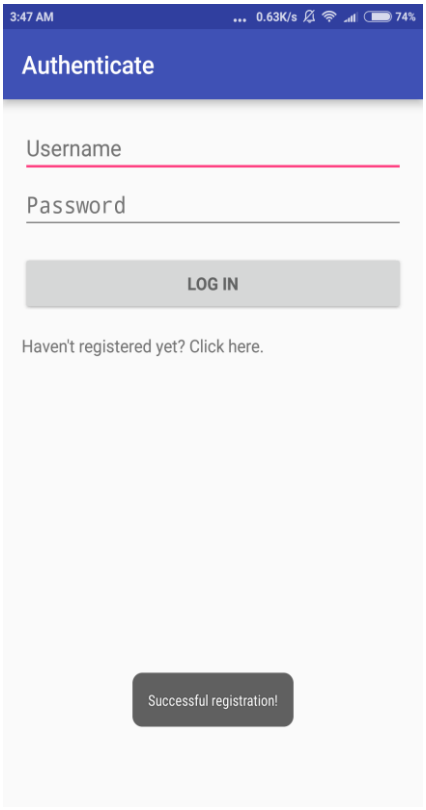
Για τα μοντέλα πρέπει να γίνει η δήλωση της ιεραρχίας των σχέσεων έτσι ώστε να μπορούν να συναρμολογούνται (serialize) και να αποσυναρμολογούνται (deserialize) τα δεδομένα που περιέχονται στο σώμα των http πακέτων στην μορφή που ορίζουν οι οδηγίες του json-api.

ΥΠΟΚΕΦΑΛΑΙΟ 3.3.2: Εγγραφή χρήστη

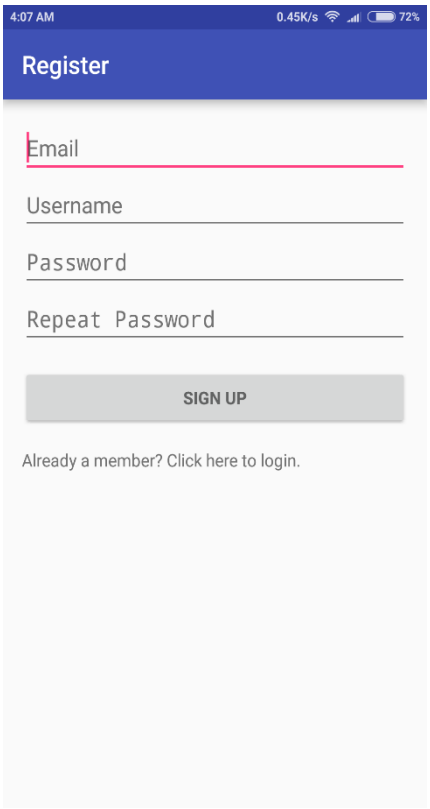
Προϋπόθεση για την χρήση της πλατφόρμας ηλεκτρονικής ψηφοφορίας είναι η εγγραφή του χρήστη στο σύστημα. Την πρώτη φορά που θα ανοίξει ο χρήστης την εφαρμογή θα αντικρύσει μια οθόνη για την ταυτοποίηση του (βλ. εικόνα 12). Οι χρήστες που δεν έχουν εγγραφεί ακόμα μπορούν να ακολουθήσουν τον σύνδεσμο που προτρέπει στην οθόνη εγγραφής (βλ. εικόνα 13).

Η διεπαφή εγγραφής αποτελείται από τέσσερα πεδία, τα οποία αντιστοιχούν στην διεύθυνση email, το ψευδώνυμο του χρήστη, τον κωδικό και την επανάληψη αυτού, δύο κουμπιά εκ των οποίων το ένα πυροδοτεί την διαδικασία εγγραφής και το άλλο μεταφέρει τον χρήστη στην οθόνη ταυτοποίησης και έναν κρυμμένο περιστρεφόμενο δείκτη (loading spinner) ο οποίος εμφανίζεται και εξαφανίζεται ανάλογα με την κατάσταση του αιτήματος για να καταλαβαίνει ο χρήστης καλύτερα

την κατάσταση της εφαρμογής. Ο αλγόριθμος κατά την υποβολή της εγγραφής προχωράει ως εξής:



Εικόνα 12 Οθόνη ταυτοποίησης



Εικόνα 13 Οθόνη εγγραφής

- Γίνεται ο έλεγχος των πεδίων για τυχόν μη αποδεκτές τιμές (validation). Στην περίπτωση παράβασης κάποιου κανόνα, τα αντίστοιχα πεδία εμφανίζουν ανάλογο μήνυμα και δίνεται έμφαση στο πεδίο με την χρήση του focus.
- Αν τα πεδία είναι αποδεκτά τότε ο αλγόριθμος αποστέλλει το αίτημα στον εξυπηρετητή. Στην περίπτωση σφάλματος εμφανίζεται ανάλογο μήνυμα στον χρήστη όπως για παράδειγμα:
  - Η διεύθυνση email ή το ψευδώνυμο χρησιμοποιείται ήδη.
  - Κάποιο πεδίο δεν είναι αποδεκτό (server validation).
  - Η εφαρμογή ή ο εξυπηρετητής αντιμετωπίζουν κάποιο πρόβλημα.
- Αν το αίτημα γίνει αποδεκτό τότε εμφανίζεται ανάλογο μήνυμα και ο χρήστης μεταφέρεται στην οθόνη ταυτοποίησης.

ΥΠΟΚΕΦΑΛΑΙΟ 3.3.3: Ταυτοποίηση χρήστη

Η οθόνη ταυτοποίησης είναι πανομοιότυπη με την οθόνη εγγραφής. Ο αλγόριθμός εκτελεί τα ίδια βήματα με την διαφορά στην μεταχείριση του JWT (Json Web Token). Στην περίπτωση λοιπόν που ο χρήστης ταυτοποιηθεί, ο αλγόριθμος θα αποθηκεύσει το token σε μορφή String (βλ. εικόνα 14) στην ειδική τοποθεσία που προβλέπει το λειτουργικό σύστημα SharedPreferences για την μελλοντική χρήση του μαζί με το ψευδώνυμο του.

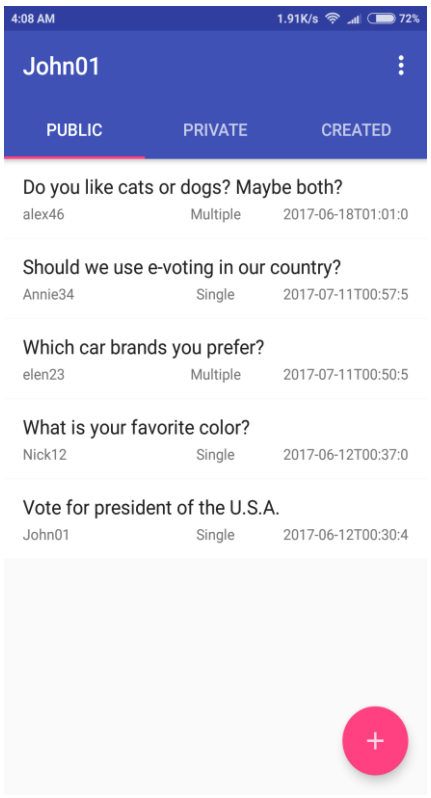
```
switch(tokenResponse.code()) {
    case 200:
        prefs.edit().putString("AUTHENTICATION_TOKEN", tokenResponse.body().getAccessToken()).apply();
        prefs.edit().putString("USERNAME", formData[0]).apply();
        makeToastShort("Successful login!");
        Intent intent = new Intent(getApplicationContext(), MainActivity.class);
        startActivity(intent);
        break;
```

Εικόνα 14 Αποθήκευση JWT (Login Activity)

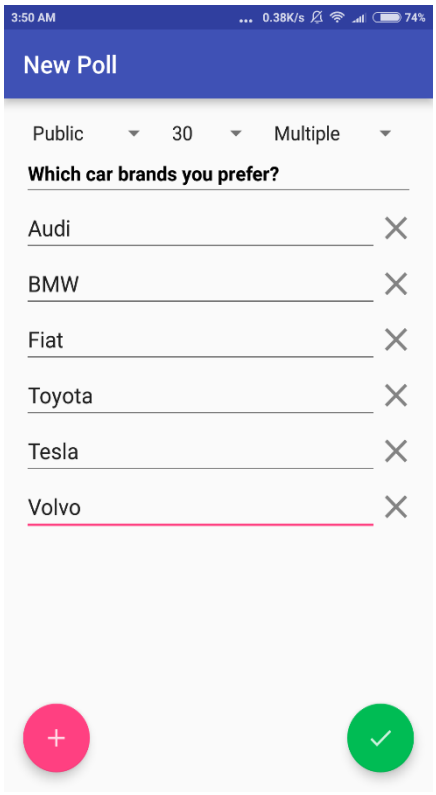
Πρόκειται για ένα μέρος της μνήμης το οποίο φιλοξενεί κάποια στοιχεία της εφαρμογής τα οποία μπορούν να είναι αναγνώσιμα και από τις υπόλοιπες εφαρμογές ή να παραμείνουν ιδιωτικά προς την εφαρμογή όπως και γίνεται στην παρούσα περίπτωση. Κάθε είδους token π.χ. (web cookies) είναι καλό να προφυλάσσεται και να μην είναι αναγνώσιμο ή εγγράψιμο από τον χρήστη της συσκευής ή από κάποια άλλη εφαρμογή (εσωτερική ή εξωτερική). Τα SharedPreferences παρόλο που αποτελούν έναν ωραίο τρόπο αποθήκευσης δεδομένων μικρού μεγέθους, δεν μπορεί να εγγυηθεί η ακεραιότητα και η μυστικότητα τους. Αυτό όμως δεν αποτελεί κίνδυνο για την περίσταση αυτή καθώς το JWT παραμένει ασφαλές όπως είδαμε στο προηγούμενο κεφάλαιο ακόμα και στην περίπτωση αλλοίωσης του. Μετά την αποθήκευση του token, ο χρήστης μεταφέρεται στην βασική οθόνη της εφαρμογής.

ΥΠΟΚΕΦΑΛΑΙΟ 3.3.4: Εμφάνιση ψηφοφοριών

Η οθόνη ψηφοφοριών είναι και η κύρια οθόνη της εφαρμογής. Κάθε φόρα που ανοίγει ο χρήστης την εφαρμογή, ο αλγόριθμος τσεκάρει αν το αναγνωριστικό token του χρήστη υπάρχει στην μνήμη. Αν δεν υπάρχει ή έχει λήξει η διάρκεια του, ο χρήστης μεταφέρεται πάλι στην οθόνη ταυτοποίησης. Με αυτό τον τρόπο ο χρήστης δεν χρειάζεται να περνάει συνεχώς από την διαδικασία της ταυτοποίησης κρατώντας τον έτσι σε logged-in-state, εκτός αν ο ίδιος επιλέξει μέσα από το μενού να αποσυνδεθεί (log out). Εκτός από την εγγραφή και την ταυτοποίηση, όπως είδαμε και στο προηγούμενο κεφάλαιο, η εφαρμογή πρέπει να προσθέτει στην κεφαλίδα «Authorization» του (HTTP) πακέτου το αναγνωριστικό token για να εγκριθεί οποιοδήποτε αίτημα έχει ασφαλίσει ο εξυπηρετητής.

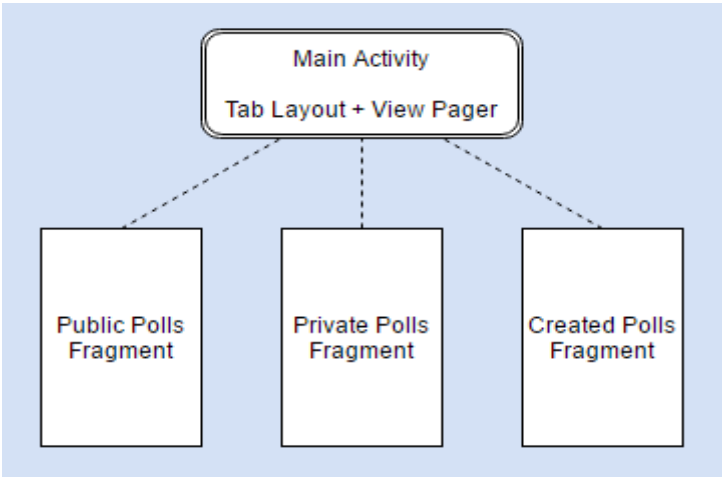


Εικόνα 15 Οθόνη ψηφοφοριών



Εικόνα 16 Οθόνη δημιουργίας ψηφοφοριών

Η κύρια οθόνη αποτελείται από τρεις καρτέλες (βλ. εικόνα 15). Στην πρώτη καρτέλα εμφανίζονται όλες οι δημόσιες ψηφοφορίες, στην δεύτερη εμφανίζονται οι ιδιωτικές ψηφοφορίες στις οποίες έχει καλεστεί ο χρήστης και στην τελευταία καρτέλα φαίνονται όλες οι ψηφοφορίες που έχει δημιουργήσει. Για να εμφανιστούν οι ψηφοφορίες σε κάθε καρτέλα, ο χρήστης πρέπει να κάνει μια κίνηση swipe-to-refresh από πάνω προς τα κάτω.



Εικόνα 17 Δομή της κύριας οθόνης

Η κύρια οθόνη ενσωματώνει τις τρεις καρτέλες ως ξεχωριστά κομμάτια ή αλλιώς fragments. Για να πετύχει αυτό γίνεται η χρήση των κλάσεων TabLayout,

ViewPager και FragmentPagerAdapter (βλ. εικόνα 17). Οι κλάσεις αυτές συνεργάζονται εύκολα μεταξύ τους, δίνοντας την δυνατότητα στον χρήστη να πλοηγείτε ανάμεσα στις καρτέλες είτε με το ακούμπημα tap μιας καρτέλας είτε με οριζόντιες κινήσεις swipe.

```
private void initializeViews() {
    Toolbar toolbar = (Toolbar) findViewById(R.id.main_toolbar);
    setSupportActionBar(toolbar);
    getSupportActionBar().setTitle(username);
    viewPager = (ViewPager) findViewById(R.id.main_viewpager);
    setupViewPager();
    TabLayout tabLayout = (TabLayout) findViewById(R.id.main_layout_tab);
    tabLayout.setupWithViewPager(viewPager);
    FloatingActionButton addButton = (FloatingActionButton) findViewById(R.id.main_add_poll_fab);
    addButton.setOnClickListener((v) → {
            Intent intent = new Intent(MainActivity.this, PollCreationActivity.class);
            startActivity(intent);
    });
}
private void setupViewPager() {
    ViewPagerAdapter adapter = new ViewPagerAdapter(getSupportFragmentManager());
    adapter.addFragment(new PublicPollsFragment(), "Public");
    adapter.addFragment(new PrivatePollsFragment(), "Private");
    adapter.addFragment(new CreatedPollsFragment(), "Created");
    viewPager.setAdapter(adapter);
    viewPager.setOffscreenPageLimit(3);
}
```

Εικόνα 18 Δημιουργία του Tab Layout (Main Activity)

Πάνω δεξιά βρίσκεται το χαρακτηριστικό κουμπί για το μενού στο οποίο ο χρήστης θα βρει δύο επιλογές «Manage friends» και «Log out». Η πρώτη τον μεταφέρει στην οθόνη διαχείρισης φίλων (βλ. εικόνα 27) που αναλύεται στο υποκεφάλαιο 3.3.8. Η δεύτερη επιλογή μεταφέρει τον χρήστη στη οθόνη ταυτοποίησης αφού σβήσει το αναγνωριστικό token από την μνήμη. Τέλος το κουμπί κάτω δεξιά μεταφέρει τον χρήστη στην οθόνη δημιουργίας ψηφοφορίας (βλ. εικόνα 16). Κάθε fragment αποτελείται από μια δομή λίστας RecyclerView (βλ. εικόνα 19) η οποία μέσω ενός αντάπτορα χειρίζεται τα αντικείμενα (items) της λίστας. Κάθε φορά που ο χρήστης κάνει κάθετο swipe:

- Γίνεται ένα αίτημα στον εξυπηρετητή για την μετάδοση των επόμενων δέκα ψηφοφοριών (page size) οι οποίες και προστίθενται στην λίστα.
- Ένας μετρητής κρατάει το νούμερο της επόμενης σελίδας (page) και όταν ο εξυπηρετητής δεν έχει άλλα αποτελέσματα τότε αυτός μηδενίζεται και η λίστα των ψηφοφοριών αδειάζει.
- Σε κάθε αντικείμενο αναγράφεται ο τίτλος, ο δημιουργός, ο τύπος και η ημερομηνία λήξης της ψηφοφορίας.

```
public class PublicPollsFragment extends Fragment {

    private SwipeRefreshLayout swipeRefreshLayout;
    private PollRecyclerAdapter adapter;
    private int currentPollsPage = 0;
    private int currentPollsPageSize = 10;

    public PublicPollsFragment() {}

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_public_polls, container, false);
        swipeRefreshLayout = (SwipeRefreshLayout) view.findViewById(R.id.public_layout_swiperefresh);
        swipeRefreshLayout.setOnRefreshListener(swipeRefreshListener);
        RecyclerView recyclerView = (RecyclerView) view.findViewById(R.id.public_recyclerview);
        recyclerView.setHasFixedSize(true);
        RecyclerView.ItemDecoration itemDecoration = new DividerItemDecoration(getActivity().getBaseContext(), Div
        recyclerView.addItemDecoration(itemDecoration);
        RecyclerView.LayoutManager layoutManager = new LinearLayoutManager(getActivity().getBaseContext());
        recyclerView.setLayoutManager(layoutManager);
        adapter = new PollRecyclerAdapter();
        recyclerView.setAdapter(adapter);
        return view;
    }
```

Εικόνα 19 Δημιουργία του Recycler View (Public Polls Fragment)

Πατώντας σε κάποιο αντικείμενο της λίστας ο χρήστης μεταφέρεται στην οθόνη λεπτομερειών της ψηφοφορίας (βλ. εικόνα 21).

ΥΠΟΚΕΦΑΛΑΙΟ 3.3.5: Δημιουργία ψηφοφορίας

Στην οθόνη δημιουργίας ο χρήστης μπορεί να συμπληρώσει τον τίτλο και να καθορίσει τον τύπο την διάρκεια και την σκοπιά της ψηφοφορίας. Υπάρχουν και τα αντίστοιχα κουμπιά για την διαγραφή ή την προσθήκη επιλογών για τις οποίες δεν υπάρχει όριο. Όταν ο χρήστης υποβάλλει την δημιουργία της ψηφοφορίας τότε:

- Γίνεται ο έλεγχος των πεδίων για τυχόν μη αποδεκτές τιμές (validation). Στην περίπτωση παράβασης κάποιου κανόνα, τα αντίστοιχα πεδία εμφανίζουν ανάλογο μήνυμα και δίνεται έμφαση στο πεδίο με την χρήση του focus.
- Αν τα πεδία είναι αποδεκτά τότε ο αλγόριθμος αποστέλλει αίτημα δημιουργίας ψηφοφορίας στον εξυπηρετητή με τα στοιχεία που συμπλήρωσε ο χρήστης. Στην περίπτωση σφάλματος εμφανίζεται ανάλογο μήνυμα στον χρήστη όπως για παράδειγμα:
  - Η δημιουργία ψηφοφορίας απέτυχε (server validation).
  - Η εφαρμογή ή ο εξυπηρετητής αντιμετωπίζουν κάποιο πρόβλημα (βλ. εικόνα 20).
- Αν το αίτημα γίνει αποδεκτό τότε εμφανίζεται ανάλογο μήνυμα και ο χρήστης μεταφέρεται πίσω στην οθόνη ψηφοφοριών.

```
@Override
public void onNext(Response<JSONAPIDocument<Poll>> pollCreationResponse) {
    switch(pollCreationResponse.code()) {
        case 201:
            makeToastShort("Poll created!");
            finish();
            break;
        case 400:
            //No need to analyze errors since we do validate before hitting the endpoint.
            makeToastShort("Failed to create poll!");
            break;
    }
}
```

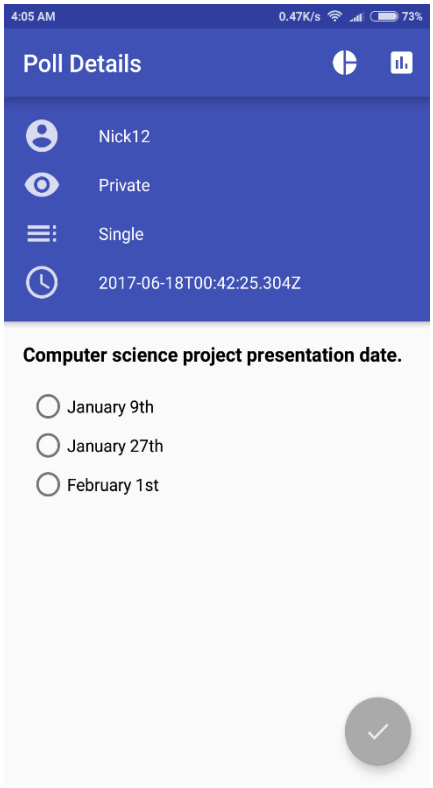
Εικόνα 20 Διαχείριση απάντησης αιτήματος (Poll Creation Activity)

ΥΠΟΚΕΦΑΛΑΙΟ 3.3.6: Λεπτομέρειες ψηφοφορίας

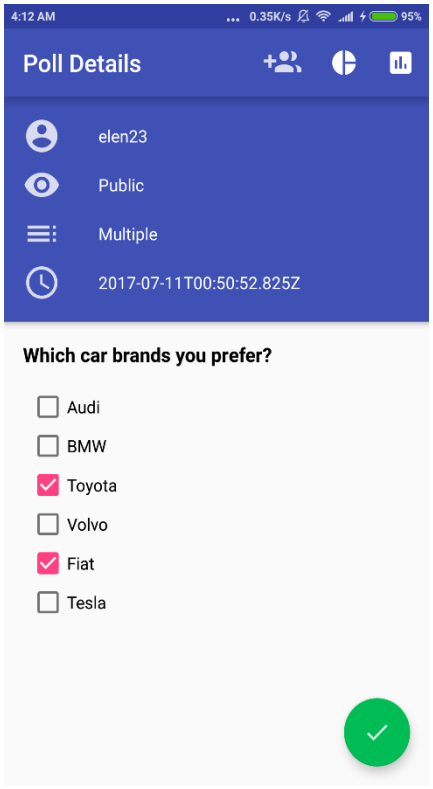
Στην οθόνη λεπτομερειών της ψηφοφορίας παρουσιάζονται τα στοιχεία της με τις επιλογές να αποτελούνται από κουμπιά radio buttons ή checkboxes στην περίπτωση που η ψηφοφορία είναι πολλαπλού τύπου (βλ. εικόνα 22).

Ο χρήστης μπορεί να επιλέξει ανάλογα και να υποβάλλει την ψήφο του. Η εφαρμογή αναλαμβάνει κάνοντας τα παρακάτω:

- Αποστέλλει αίτημα δημιουργίας ψήφου για την συγκεκριμένη ψηφοφορία. Αν υπάρχει σφάλμα τότε εμφανίζεται ανάλογο μήνυμα όπως:
  - Η ψηφοφορία έχει λήξει.
  - Αποτυχία δημιουργίας ψήφου.
  - Έχετε ψηφίσει ήδη.
  - Η εφαρμογή ή ο εξυπηρετητής αντιμετωπίζουν κάποιο πρόβλημα.
- Αν το αίτημα είναι επιτυχές τότε εμφανίζεται ανάλογο μήνυμα και αυξάνεται ο μετρητής των ψήφων.



Εικόνα 21 Οθόνη λεπτομερειών (Single)



Εικόνα 22 Οθόνη λεπτομερειών (Multiple)

Στην συνέχεια ο χρήστης μπορεί να πατήσει το κουμπί γραφήματος (πίτας) πάνω δεξιά στην μπάρα για να μεταφερθεί στην οθόνη γραφήματος πίτας (βλ. εικόνα 24). Στην οθόνη γραφήματος φαίνονται τα αποτελέσματα σε ωραία μορφή με διαφορετικούς χρωματισμούς, κάτω αριστερά αναγράφονται οι ετικέτες χρωματισμού για κάθε επιλογή και κάτω δεξιά αναγράφεται το σύνολο των ψήφων. Για το γράφημα γίνεται η χρήση των κλάσεων της βιβλιοθήκης MPAndroidChart. Πρώτα φτιάχνουμε το dataset και μετά ανανεώνουμε το γράφημα (βλ. εικόνα 23). Αν ο χρήστης είναι ο δημιουργός της ψηφοφορίας, πάνω στην μπάρα εμφανίζεται το κουμπί για την πρόσκληση άλλων χρηστών που έχει νόημα αν η εκλογή είναι ιδιωτική. Με το πάτημα του κουμπιού ο χρήστης μεταφέρεται στην οθόνη πρόσκλησης χρηστών (βλ. εικόνα 25).

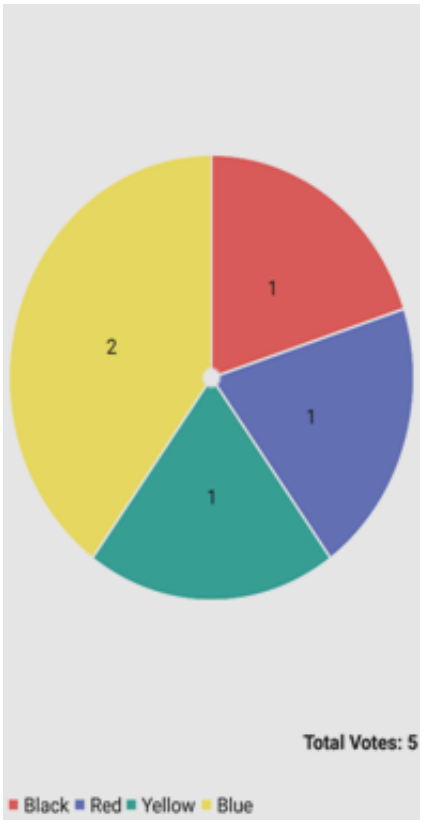
```
//Create the data set.
PieDataSet pieDataset = new PieDataSet(pieEntries, "");
pieDataset.setSliceSpace(2f);

//Add multiple colors.
pieDataset.setColors(PIE_CHART_COLOR_PALETTE);

PieData pieData = new PieData(pieDataset);
pieData.setValueTextSize(16f);
pieData.setValueFormatter(new DefaultValueFormatter(0));

//Configure pie chart view.
pieChart.setData(pieData);
pieChart.setHoleRadius(4f);
pieChart.setDrawEntryLabels(false);
pieChart.setTransparentCircleRadius(5f);
pieChart.getDescription().setText("Total Votes: " + totalVotes);
pieChart.getDescription().setTextSize(16f);
pieChart.getDescription().setTextAlign(Paint.Align.RIGHT);
pieChart.getDescription().setTypeface(Typeface.DEFAULT_BOLD);
pieChart.getLegend().setTextSize(16f);
pieChart.getLegend().setWordWrapEnabled(true);
pieChart.getLegend().setDirection(Legend.LegendDirection.LEFT_TO_RIGHT);
pieChart.getLegend().setHorizontalAlignment(Legend.LegendHorizontalAlignment.LEFT);
pieChart.getLegend().setVerticalAlignment(Legend.LegendVerticalAlignment.BOTTOM);

//Refresh pie chart and animate.
pieChart.invalidate();
pieChart.animateY(3500 , Easing.EasingOption.EaseInOutExpo);
```

Εικόνα 23 Προετοιμασία του γραφήματος (Pie Chart Activity)

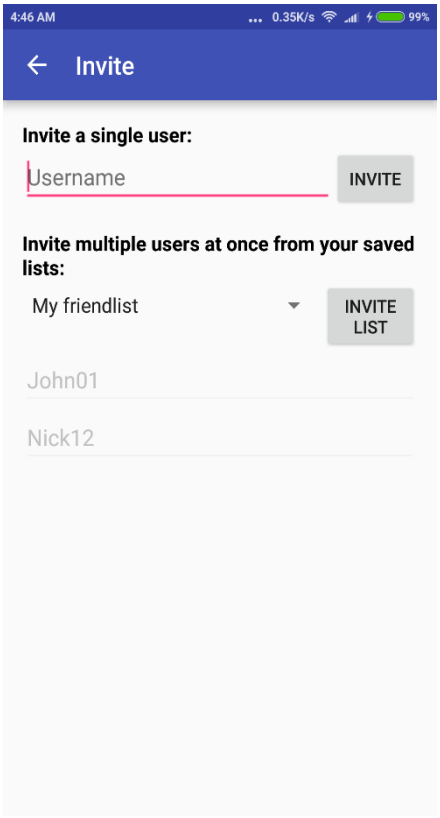
ΥΠΟΚΕΦΑΛΑΙΟ 3.3.7: Πρόσκληση φίλων

Στην οθόνη πρόσκλησης χρηστών ο χρήστης μπορεί να εκμεταλλευτεί τις λίστες φίλων για να στείλει μαζικές προσκλήσεις. Κατά την υποβολή ο αλγόριθμος για κάθε φίλο:

- Στέλνει ένα αίτημα δημιουργίας πρόσκλησης για την συγκεκριμένη ψηφοφορία και το συγκεκριμένο ψευδώνυμο. Αν η πρόσκληση είναι επιτυχής ή ο χρήστης έχει ήδη προσκληθεί, το όνομα του πρασινίζει.
- Αν υπάρχει σφάλμα τότε εμφανίζεται ανάλογο μήνυμα όπως:
  - Η ψηφοφορία έχει λήξει.
  - Η εφαρμογή ή ο εξυπηρετητής αντιμετωπίζουν κάποιο πρόβλημα.



Εικόνα 24 Οθόνη πρόσκλησης χρηστών



Εικόνα 25 Οθόνη γραφήματος πίτας

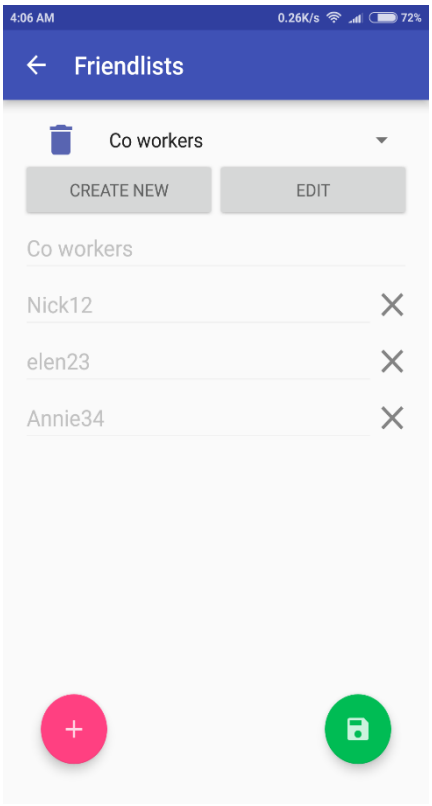
Παρόλα αυτά ο χρήστης μπορεί να στείλει τις προσκλήσεις μεμονωμένα η κατάσταση των οποίων επιβεβαιώνεται με ανάλογο μήνυμα. Ο αλγόριθμος είναι πανομοιότυπος με τον παραπάνω.

ΥΠΟΚΕΦΑΛΑΙΟ 3.3.8: Διαχείριση φίλων

Η οθόνη διαχείρισης φίλων είναι και η τελευταία αυτής της εφαρμογής. Στην οθόνη αυτή ο χρήστης μπορεί να δημιουργήσει λίστες ψευδώνυμων χρηστών για να γίνεται ευκολότερη η διαδικασία προσκλήσεων. Οι λίστες αποθηκεύονται χρησιμοποιώντας ξανά τα SharedPreferences. Σε κάθε περίπτωση χρήσης (δημιουργία ή διαγραφή) αποθηκεύουμε τα ονόματα των λιστών και κρατάμε ξεχωριστά Strings για κάθε λίστα με κλειδί το όνομα και τιμή τα ψευδώνυμα χωρισμένα με κόμμα (βλ. εικόνα 26).

```
prefs.edit().putString(selectedFriendlistKey, friendlistValue.toString()).apply();
friendlists.add(selectedFriendlistKey);
prefs.edit().putStringSet(USERNAME + "_" + "FRIENDLISTS", friendlists).apply();
spinnerAdapter.add(selectedFriendlistKey);
```

Εικόνα 26 Αποθήκευση λίστας φίλων (Friendlists Activity)



Εικόνα 27 Οθόνη διαχείρισης φίλων

ΥΠΟΚΕΦΑΛΑΙΟ 3.4: Προβλήματα - Προτάσεις

1) **Ασφάλεια αναγνωριστικού token:** Καλύτερα μέτρα ασφαλείας θα μπορούσαν να παρθούν για την διασφάλιση του JWT που αποθηκεύεται στα SharedPreferences. Ίσως η παράλληλη χρήση API keys θα μπορούσε να δώσει λύση στο πρόβλημα της ασφάλειας.

2) **Μέγεθος εφαρμογής:** Παρόλο που μια τέτοια εφαρμογή θα μπορούσε να αναπτυχθεί με πολύ μικρότερο μέγεθος, η παρούσα αγγίζει τα 40 Mega Byte. Το μέγεθος ευθύνεται στην βαριά χρήση των βιβλιοθηκών.

3) **Έλεγχος κώδικα:** Δεν έχουν συγγραφεί unit tests για κάθε ξεχωριστή λειτουργία της εφαρμογής. Αυτό σημαίνει ότι σε μελλοντικές αλλαγές θα πρέπει να ελεγχθούν ξανά όλες οι λειτουργίες ξεχωριστά, κάτι που είναι πολύ χρονοβόρο και αντιπαραγωγικό για τον προγραμματιστή.

4) **Μεταβολή προσανατολισμού οθόνης:** Ένα θέμα που προβληματίζει τους περισσότερους αρχάριους προγραμματιστές android εφαρμογών είναι η μεταβολή προσανατολισμού οθόνης κατά την διάρκεια χρήσης της εφαρμογής. Το λειτουργικό σύστημα σε περίπτωση διακοπής π.χ. εισερχόμενη κλήση, διαλύει την οθόνη και την ξαναδημιουργεί όταν ο

χρήστης διευθετήσει το θέμα. Στην παρούσα εφαρμογή, είναι λίγες περιπτώσεις χρήσης στις οποίες δεν αντιμετωπίζεται καθόλου το πρόβλημα αυτό.

5) **Απόδοση ανά συσκευή:** Δεν υπάρχει καμία εγγύηση για την απόδοση της εφαρμογής σε παλαιότερες συσκευές καθώς δεν έχει γίνει ο απαραίτητος έλεγχος.

6) **Οργάνωση κώδικα:** Ο κώδικας θα μπορούσε να ακολουθεί πολύ καλύτερη δομή με την χρήση κάποιου pattern όπως για παράδειγμα του MVP (model, view, presenter). Αυτό βέβαια προσθέτει μεγάλη πολυπλοκότητα για μια τέτοια εφαρμογή αλλά θα διευκολύνει την μελλοντική συντήρηση και την συγγραφή πιο απομονωμένων ελέγχων (unit tests).

ΣΥΜΠΕΡΑΣΜΑΤΑ

Η παρούσα πτυχιακή εργασία επιχείρησε να παρουσιάσει τα βασικά χαρακτηριστικά που καθορίζουν ένα σύστημα ηλεκτρονικής ψηφοφορίας. Με την παρουσίαση των προβλημάτων και των προκλήσεων του χώρου των ηλεκτρονικών εκλογών έγινε πιο εύκολη η αντιμετώπιση του βασικού σκοπού της εργασίας.

Η υλοποίηση της πλατφόρμας αναπτύχθηκε λαμβάνοντας υπόψιν τις σωστές πρακτικές προγραμματισμού και ανάπτυξης λογισμικού. Το σύστημα είναι όμοιο μιας εμπορική εφαρμογή καθώς χρησιμοποιεί μοντέρνες τεχνολογίες οι οποίες παρατηρούνται συχνά στην σημερινή αγορά. Παρόλα αυτά αποτελεί ένα λειτουργικό σύστημα για διεξαγωγή απλών μη-κρίσιμων ψηφοφοριών.

Με γνώμονα την έρευνα που έγινε στην σχετική βιβλιογραφία, συμπεραίνεται ότι οι ηλεκτρονικές εκλογές απαιτούν ιδιαίτερη προσοχή καθώς διακυβεύεται η δημοκρατία. Είναι προτιμότερο να συνεχίσει η χρήση των παραδοσιακών μέσων ψηφοφορίας από το να υιοθετηθεί βεβιασμένα κάποιο σύστημα απλά και μόνο στο όνομα της τεχνολογίας. Θεωρείται πως ένα καλοσχεδιασμένο σύστημα ηλεκτρονικής διακυβέρνησης όπως για παράδειγμα αυτό της Εσθονίας, αποτελεί τεράστιο πλεονέκτημα στην υποστήριξη ηλεκτρονικών εκλογών. Η ενθάρρυνση χρήσης των ηλεκτρονικών εκλογών σε εκλογές χαμηλότερου προφίλ όπως για παράδειγμα οι ακαδημαϊκές εκλογές, είναι ο σωστός δρόμος για την δοκιμή και την συλλογή δεδομένων της χρήσης των ηλεκτρονικών συστημάτων. Οι τελευταίες εξελίξεις στο χώρο της ασφάλειας των υπολογιστών αποτελούν άλλον έναν λόγο για τον οποίο η διεξαγωγή ηλεκτρονικών εκλογών θα πρέπει να επανεξεταστεί με σοβαρότητα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Clarkson, M. R., Chong, S., & Myers, A. C. (2008). Civitas: Toward a Secure Voting System (pp. 354–368). IEEE. https://doi.org/10.1109/SP.2008.32

Delaune, S., Kremer, S., & Ryan, M. (2006). Coercion-resistance and receipt-freeness in electronic voting. In Computer Security Foundations Workshop, 2006. 19th IEEE (p. 12–pp). IEEE. Retrieved from http://ieeexplore.ieee.org/abstract/document/1648706/

Deutsch, H. (2005). Public opinion's influence on voting system technology. Computer, 38(3), 93–95.

Deutsch, H., & Berger, S. (2004). STANDARDS AND CERTIFICATIONS. Communications of the ACM, 47(10), 31.

Evans, D., & Paul, N. (2004). Election security: Perception and reality. IEEE Security & Privacy, 2(1), 24–31.

Feldman, A. J., Halderman, J. A., & Felten, E. W. (2006). Security analysis of the Diebold AccuVote-TS voting machine. Retrieved from https://www.usenix.org/event/evt07/tech/full_papers/feldman/feldman_html/

Grewal, G. S., Ryan, M. D., Bursuc, S., & Ryan, P. Y. A. (2013). Caveat Coercitor: Coercion-Evidence in Electronic Voting (pp. 367–381). IEEE. https://doi.org/10.1109/SP.2013.32

Kohno, T., Stubblefield, A., Rubin, A. D., & Wallach, D. S. (2004). Analysis of an electronic voting system. In IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004 (pp. 27–40). https://doi.org/10.1109/SECPRI.2004.1301313

Krimmer, R., Grimm, R., EVOTE, & Europarat (Eds.). (2008). Electronic voting 2008: (EVOTE08) ; 3rd international conference ; August 6th - 9th in Castle Hofen, Bregenz, Austria. Bonn: Ges. für Informatik.

Neumann, P. G. (2004). VOTING SYSTEMS. Communications of the ACM, 47(10), 29.

Osho, O., Yisa, V. L., & Jebutu, O. J. (2015). E-voting in Nigeria: A survey of voters' perception of security and other trust factors. In Cyberspace (CYBER-Abuja), 2015 International Conference on (pp. 202–211). IEEE. Retrieved from http://ieeexplore.ieee.org/abstract/document/7360511/

Penha-Lopes, J. M. (2005). Why use an open source e-voting system? ACM SIGCSE Bulletin, 37(3), 412–412.

Ryan, P. Y. A., Bismark, D., Heather, J., Schneider, S., & Zhe Xia. (2009). PrÊt À Voter: a Voter-Verifiable Voting System. IEEE Transactions on Information Forensics and Security, 4(4), 662–673. https://doi.org/10.1109/TIFS.2009.2033233

Serdult, U., Germann, M., Mendez, F., Portenier, A., & Wellig, C. (2015). Fifteen years of internet voting in Switzerland [history, governance and use]. In eDemocracy & eGovernment (ICEDEG), 2015 Second International Conference on (pp. 126–132). IEEE. Retrieved from http://ieeexplore.ieee.org/abstract/document/7114482/

Springall, D., Finkenauer, T., Durumeric, Z., Kitcat, J., Hursti, H., MacAlpine, M., & Halderman, J. A. (2014). Security Analysis of the Estonian Internet Voting System (pp. 703–715). ACM Press. https://doi.org/10.1145/2660267.2660315

Trechsel, A. H., & Breuer, F. (2006). E-voting in the 2005 local elections in Estonia and the broader impact for future e-voting projects. In Proceedings of the 2006 international conference on Digital government research (pp. 40–41). Digital Government Society of North America. Retrieved from http://dl.acm.org/citation.cfm?id=1146614

Williams, B. J., & King, M. S. (2004). Implementing voting systems: the Georgia method. Communications of the ACM, 47(10), 39–42.

ΟΔΗΓΟΣ ΧΡΗΣΗΣ ΛΟΓΙΣΜΙΚΟΥ

1) Download and install PostgreSQL DB from the official website.

2) Create a database for the Play server and make sure you write down the credentials. You can use pgAdmin or pgAdmin4Web for a graphical interface and ease of use.

3) Download and install the latest JDK. Current version of the Play project requires version of at least 1.8. Make sure java is correctly installed by opening a cmd and typing 'java -version'. Edit the environment variables and set the Java path.

4) Download and install scala-sbt from the official website. Current version of the Play project requires version of at least 0.13.13. Make sure sbt is correctly installed by opening a cmd and typing 'sbt -version'.

5) Extract the Play project or clone it from the GitHub repository (if still available) in a folder. The Play Framework Version of this project is 2.5.4. In case of errors check the official Play Framework website or use their mailing list. For further development, it is recommended to use IntellJ's IDE.

6) Edit "application.conf" within the project conf directory and change the database credentials. The database configurations are located within the ".conf" file in the db object surrounded with '{}'.

7) Open a cmd and navigate to top level of the project folder "Voteapp". Type the command 'sbt' and press enter. Wait until sbt has loaded the project and downloaded its dependencies. You should be running a sbt shell. Now type 'run' and press enter to bring the server online. The server listens to localhost:9000. You can stop the process by pressing 'Ctrl+D'. The first time you visit the page apply the SQL script prompted for the tables' initialization. To activate https, you need to add the '-Dhttps = enabled' as a parameter to the 'run' command. Make sure to check Play's website on how to configure the key store and https. The API documentation resides in the "/documentation" URI.

8) The android app has the server's local IP hardcoded within the NetworkApi.java file. For it to communicate with the server, you must recompile the app with the new IP address. For the android project, you can extract or clone it from the GitHub repository (if still available) and edit it with Android Studio.