Αυτόνομο αμεσοδημοκρατικό κοινωνικό δίκτυο βασισμένο σε τεχνολογίες αποκέντρωσης

Νικολαΐδης Παναγιώτης[1] και Φανάκης Απόστολος[2]

[1]nkpanagi@auth.gr, 7491
[2]apostolof@auth.gr, 8261

Επιβλέπων Χρήστος Ε. Δημάκης

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Πολυτεχνική Σχολή

Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Θεσσαλονίκη 2021

Περίληψη

Τις τελευταίες δεκαετίες, η ραγδαία ανάπτυξη του διαδικτύου μετέβαλε ριζικά τις ανθρώπινες κοινωνίες, μέσω μίας πληθώρας ψηφιακών εφαρμογών, οι οποίες, στη συντριπτική τους πλειοψηφία, προσφέρονται από παρόχους υπηρεσιών υπολογιστικού νέφους, ακολουθώντας την αρχιτεκτονική πελάτη-εξυπηρετητή.

Μολονότι αυτό το μοντέλο υλοποίησης έχει αποδειχθεί ιδιαίτερα λειτουργικό και έχει βελτιωθεί αξιοσημείωτα ανά τα χρόνια, η συγκεντρωτική του λογική συνοδεύεται από μία σειρά προβλημάτων. Πρώτα απ' όλα, ο χρήστης καλείται να εμπιστευθεί τα προσωπικά του δεδομένα σε μία τρίτη οντότητα. Εκείνη, διατηρώντας τον πλήρη έλεγχο επί αυτών, διαθέτει τη δυνατότητα να τα επεξεργάζεται, να τα διαμοιράζεται και να τα λογοκρίνει, είτε για να εξυπηρετήσει τα συμφέροντά της, είτε για να συμμορφωθεί με άλλες αρχές που της ασκούν εξουσία. Επιπλέον, απουσιάζει η εγγύηση της διαθεσιμότητας των δεδομένων, καθώς, ανά πάσα στιγμή, ο εξυπηρετητής μπορεί να αποσυνδεθεί για αόριστο χρονικό διάστημα και λόγω ποικίλων αιτιών, όπως κάποιας κυβερνοεπίθεσης ή κάποιας φυσικής καταστροφής.

Αυτοί είναι μερικοί βασικοί λόγοι που συνετέλεσαν στην ταχεία ανάπτυξη ενός συνόλου λογισμικών ανοιχτού κώδικα, όπως του Ethereum blockchain και του IPFS, τα οποία, αν και βρίσκονται σε σχετικά πρώιμο στάδιο, αποτελούν ήδη ικανά εργαλεία δημιουργίας κατανεμημένων και αποκεντρωμένων εφαρμογών.

Στόχος της παρούσας διπλωματικής εργασίας είναι η υλοποίηση μίας αυτόνομης κοινωνικής πλατφόρμας, η οποία, αξιοποιώντας τεχνολογίες αποκέντρωσης, αφενός θα επιστρέφει την κυριότητα των προσωπικών δεδομένων στον χρήστη, αφετέρου θα παρέχει τη δυνατότητα για διαφανείς δημοκρατικές ψηφοφορίες. Αυτά μέσα σε ένα πλαίσιο ανθεκτικό, τόσο σε σφάλματα και επιθέσεις, όσο και σε απόπειρες λογοκρισίας και παραποίησης.

Λέξεις-Κλειδιά: Αποκεντροποίηση, Ethereum, Blockchain, Έξυπνο Συμβόλαιο, Αποκεντρωμένη Εφαρμογή, IPFS, OrbitDB

Abstract

Don’t forget the keywords.

Ευχαριστίες

Ευχαριστούμε η Αθήνα. Ευχαριστούμε η Ελλάδα.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Γενικά

1.2 Ορισμός του προβλήματος

Στις μέρες μας τα περισσότερα δεδομένα των χρηστών βρίσκονται υπό τον έλεγχο συγκεντρωτικών συστημάτων. Σε τέτοια συστήματα οι χρήστες δεν είναι κύριοι των δεδομένων τους, δεν έχουν εγγύηση για την αυθεντικότητα αυτών που βλέπουν και υπόκεινται σε λογοκρισία, ενώ τα συστήματα αυτά δεν είναι ασφαλή και μπορεί να σταματήσουν να λειτουργούν προσωρινά ή μόνιμα για τεχνικούς/οικονομικούς/νομικούς λόγους.

Οι περισσότερες διαδεδομένες, συγκεντρωτικές μορφές πλατφόρμας επικοινωνίας (mailing list, forum, κοινωνικά δίκτυα και άλλες) χρειάζονται, τυπικά, τουλάχιστον τις εξής τεχνολογίες:

- μία πηγή επεξεργαστικής ισχύος (processing)
- μία βάση δεδομένων
- ένα πρωτόκολλο επικοινωνίας

Η επεξεργαστική ισχύς είναι αναγκαία για την περάτωση των λειτουργιών οι οποίες υλοποιούν τις υπηρεσίες της πλατφόρμας. Τις περισσότερες φορές η πηγή αυτή είναι ένας server ή μία cloud υπηρεσία.

Η βάση δεδομένων είναι απαραίτητη για την αποθήκευση της πληροφορίας. Σε μικρότερες εφαρμογές η βάση βρίσκεται στο ίδιο σύστημα που γίνεται και το processing, ενώ σε μεγαλύτερες ενδέχεται να υπάρχει για λόγους ασφάλειας ένα ξεχωριστό σύστημα αφιερωμένο στη βάση δεδομένων.

Το πρωτόκολλο επικοινωνίας αναλαμβάνει τη μετάδοση και ανάκτηση της πληροφορίας. Το πρωτόκολλό που χρησιμοποιείται σήμερα στη συντριπτική πλοιοψηφία των εφαρμογών είναι το HTTP.

Κάθε ένα από τα παραπάνω μέρη, εισάγει την ανάγκη ύπαρξης κεντρικών αρχών που τα διαχειρίζονται και τα συντηρούν. Η αρχή αυτή είναι συνήθως ο πάροχος της υπηρεσίας που διαχειρίζεται το processing και τη βάση δεδομένων, έχοντας έτσι πρόσβαση σε όλα τα δεδομένα που υπάρχουν στο σύστημα.

1.3 Προτεινόμενη λύση

Το Concordia είναι η εφαρμογή η οποία αναπτύσσουμε εμείς και στοχεύει να διορθώσει αυτά τα προβλήματα, επαναφέροντας στους χρήστες την κυριότητα των δεδομένων τους, εξασφαλίζοντας την πλήρη ελευθερία του λόγου και την αυθεντικότητα, ανοίγοντας τον δρόμο για αξιόπιστες ψηφοφορίες Όλα αυτά μέσα από δημόσιες, αποκεντρωτικές διαδικασίες.

1.3.1 Απαιτήσεις

1.3.2 Αποκέντρωση

Αποκέντρωση του συστήματος σημαίνει πρακτικά ότι το processing και η αποθήκευση των δεδομένων δε θα γίνονται από κάποια κεντρική αρχή αλλά θα είναι κατανεμημένα στο σύνολο των χρηστών (nodes). Με αυτόν τον τρόπο δεν υπάρχει ανάγκη για μία κεντρική αρχή και τα δεδομένα δεν είναι ελέγξιμα από κανέναν ατομικά, παρά μόνο από τη συναίνεση (consensus) του δικτύου.

Τα συγκεντρωτικά συστήματα έχουν μερικά θετικά χαρακτηριστικά που λείπουν από τα αποκεντρωτικά συστήματα, όπως ευκολία ανάπτυξης, συντήρησης και αποσφαλμάτωσης των εφαρμογών. Πάσχουν ωστόσο σε ό,τι αφορά την σταθερότητα, την ασφάλεια, την επεκτασιμότητα και την εξέλιξη, τομείς όπου τα αποκεντρωτικά συστήματα είναι ιδιαίτερα αποτελεσματικά.

Η ανάγκη για αποκέντρωση των εφαρμογών, ειδικά στην επικοινωνία, είναι μεγάλη και πηγάζει από την ανάγκη για ελευθερία του λόγου, ανωνυμία και ιδιωτικότητα. Χρησιμοποιώντας τεχνικές για κατανομή του processing, μία διανεμημένη βάση δεδομένων και αλγόριθμους κρυπτογραφίας δημόσιου κλειδιού μπορούμε να προστατεύσουμε την ανωνυμία του χρήστη αλλά και να εγγυηθούμε την ταυτοποίησή του.

1.3.3 Αμεσοδημοκρατικές διαδικασίες και αυτοδιαχείριση

Για την πλήρη επίτευξη του στόχου απαιτείται επίσης ένα σύστημα διαχείρισης της πλατφόρμας αυτής καθ' αυτής αλλά και των περιεχομένων της. Το σύστημα που επιλέγουμε για αυτούς τους σκοπούς είναι αυτό της άμεσης δημοκρατίας και αυτοδιαχείρισης. Αυτό σημαίνει ότι οι αποφάσεις θα παίρνονται μέσα από ψηφοφορίες στις οποίες θα μπορούν να συμμετέχουν όσα μέλη έχουν δικαίωμα ψήφου. Έτσι, λόγω της αποκέντρωσης και άρα της έλλειψης διοικούσας αρχής, η πλατφόρμα μπορεί να χρησιμοποιηθεί σαν μία εγγυημένα αμερόληπτη αρχή για ψηφοφορίες πάνω σε θέματα που αφορούν τη φοιτητική ζωή και όχι μόνο.

1.4 Στόχος

Στόχος του project είναι η δημιουργία μιας κοινωνικής πλατφόρμας, η οποία, βασιζόμενη σε τεχνολογίες αποκέντρωσης, αφενός θα παρέχει ελευθερία λόγου, εργαλεία αυτοδιαχείρισης και αμεσοδημοκρατικές διαδικασίες, αφετέρου θα διασφαλίζει την κυριότητα των δεδομένων του χρήστη από τον ίδιο και την ανεξαρτητοποίηση του συστήματος από κεντρικές οντότητες. Παράλληλα, θα παρέχει στους επαληθευμένους χρήστες του ΑΠΘ μια πλατφόρμα για ανώνυμες και αυθεντικές ψηφοφορίες, εν δυνάμει ικανών να αποτελέσουν ένα έγκυρο, έμπιστο και άμεσα δημοκρατικό βήμα λήψης αποφάσεων.

1.5 Μεθοδολογία ανάπτυξης

Για την επίτευξη των στόχων που ορίστηκαν και την οργάνωση της απαραίτητης δουλειάς σε διαχειρίσιμα μέρη σχεδιάστηκε η χρήση διάφορων εργαλείων και μεθόδων ανάπτυξης λογισμικού, όπως το σύστημα ελέγχου εκδόσεων (version control system) Git και η μέθοδος οργάνωσης Scrum. Τα εργαλεία αυτά είναι δοκιμασμένα και έχουν εδραιωθεί στη σύχρονη ανάπτυξη λογισμικού.

Το Git είναι δωρεάν λογισμικό ανοιχτού κώδικα το οποίο επιτρέπει και επικουρεί την απρόσκοπτη ανάπτυξη λογισμικού από πολλαπλά μέλη μίας ομάδας, ταυτόχρονα και διανεμημένα. Αυτό επιτυγχάνεται παρέχοντας ένα πλαίσιο από εργαλεία τα οποία βοηθούν την διαχείριση και ενσωμάτωση των διαφορετικών εκδόσεων του κώδικα τις οποίες αναπτύσει κάθε μέλος της ομάδας ξεχωριστά. Υπάρχουν διάφορα μοντέλα χρήσης του Git και πιο συγκεκριμένα της δυνατότητας που δίνει για δημιουργία, ανάπτυξη και ένωση κλαδιών (branches). Για τους σκοπούς της παρούσας διπλωματικής σχεδιάστηκε η χρήση του μοντέλου που έχει αναπτυχθεί από την εταιρία Github, Flow. Το μοντέλο αυτό ορίζει ότι κάθε προγραμματιστής θα ανοίγει ένα νέο branch για τη ανάπτυξη ενός νέου χαρακτηριστικού της εφαρρμογής ή την διόρθωση ενός μέρους του κώδικα. Έπειτα, όταν η δουλειά έχει ολοκληρωθεί το branch ενώνεται (merge) με το βασικό branch της εφαρμογής.

Το Scrum είναι μία μέθοδος οργάνωσης στην οποία ο/η επιμελητής/επιμελήτρια του Scrum (Scrum master) διαχωρίζει τα ανεξάρτητα μέρη εργασίας (tasks) που πρέπει να υλοποιηθούν για την ολοκλήρωση των στόχων ενός project. Τα μέρη αυτά περιγράφονται αναλυτικά μαζί με τις απαιτήσεις τους και κατατίθενται σε μία λίστα εργασιών (backlog). Έπειτα, μέσα από συσκέψεις (meetings), επιλέγεται ένας αριθμός από μέρη εργασίας τα οποία θα αποτελέσουν το επόμενο Sprint. Κάθε μέρος εργασίας ανατίθεται σε κάποιο μέλος για υλοποίηση και ορίζεται για το Sprint μία χρονική διάρκεια, στόχος της οποίας είναι η περάτωση όλων των μερών εργασίας πριν τη λήξη της. Στο τέλος προθεσμίας που ορίστηκε για το Sprint τα μέλη της ομάδας αποτιμούν τα αποτελέσματα και ορίζουν το επόμενο Sprint. Η διαδικασία επαναλαμβάνεται έως ότου το έργο ολοκληρωθεί.

Μέσα από την χρήση των παραπάνω εργαλείων επιτυγχάνεται η ομαλή συνεργασία στην ανάπτυξη του λογισμικού. Κάθε μέλος της ομάδας δύναται να εργαστεί ανεξάρτητα και χωρίς την ανάγκη διακρούς επικοινωνίας με τα υπόλοιπα μέλη. Οι στόχοι είναι ορισμένοι, σαφής και χωρισμένοι σε διαχειρίσιμα μέρη τα οποία δεν καταβάλουν τα μέλη. Ταυτόχρονα, έχοντας ως έδρα καθιερομένα πρότυπα ανάπτυξης, παρέχεται φορμαλισμός και έτοιμες μέθοδοι επίλυσης προβλημάτων, γεγονός που λειτουργεί καταλυτικά και βοηθά στην αποφυγεί τελμάτων κατά τη συγγραφή του κώδικα.

1.6 Οργάνωση κεφαλαίων

Κεφάλαιο 2

Θεωρητικό υπόβαθρο

2.1 Συναρτήσεις κατακερματισμού

Οι **κρυπτογραφικές συναρτήσεις κατακερματισμού** (cryptographic hash functions) είναι ειδική κατηγορία συναρτήσεων κατακερματισμού σχεδιασμένες για χρήση στην κρυπτογραφία. Αποτελούν μαθηματικές συναρτήσεις που δέχονται ως είσοδο δεδομένα τυχαίου μεγέθους και επιστρέφουν συμβολοσειρές σταθερού μήκους.

Οι τιμές που επιστρέφει η συνάρτηση κατακερματισμού ονομάζονται τιμές κατακερματισμού (hash values, digests ή απλά hashes). Μία ιδανική κρυπτογραφική συνάρτηση κατακερματισμού έχει τις εξής βασικές ιδιότητες:

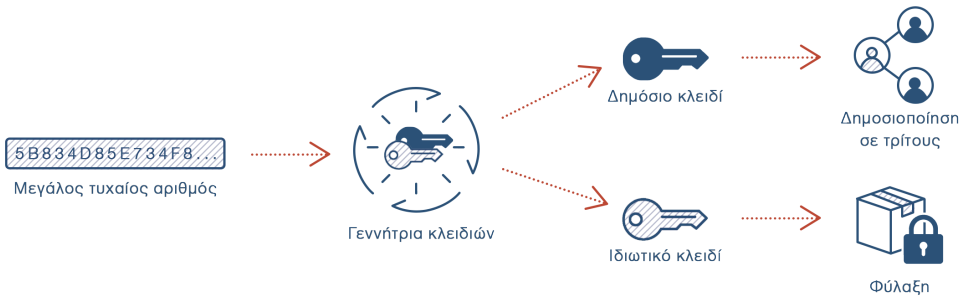
- Είναι ντετερμινιστική, δηλαδή η ίδια είσοδος παράγει πάντα την ίδια έξοδο.
- Είναι μη αντιστρέψιμη, δηλαδή είναι πρακτικά ανέφικτο να υπολογιστεί η είσοδος δεδομένης μιας εξόδου.
- Είναι αμφιμονοσήμαντη (1-1), δηλαδή σε δύο διαφορετικές εισόδους αντιστοιχούν πάντα δύο εντελώς διαφορετικές έξοδοι.
- Είναι αποδοτική, δηλαδή ο υπολογισμός του hash οποιασδήποτε εισόδου είναι γρήγορος.

2.2 Ασύμμετρη κρυπτογραφία

Η **ασύμμετρη κρυπτογραφία** (asymmetric cryptography) ή κρυπτογραφία δημόσιου κλειδιού (public-key cryptography) αποτελεί κρυπτογραφικό σύστημα που βασίζεται στη χρήση ενός ζεύγους κλειδιών (key pair), του *δημόσιου* (public key) και του *ιδιωτικού* (private key). Αυτά τα κλειδιά είναι μαθηματικά συνδεδεμένα ως εξής:

- Το ιδιωτικό κλειδί δε μπορεί να προκύψει γνωρίζοντας το δημόσιό του
- Ό,τι κρυπτογραφηθεί από το ένα μπορεί να αποκρυπτογραφηθεί μόνο από το άλλο

Η δημιουργία ενός ζεύγους κλειδιών επιτυγχάνεται μέσω μιας *γεννήτριας κλειδιών* (key generation function), η οποία χρησιμοποιεί ειδικούς αλγορίθμους (π.χ. RSA), δεχόμενη ως είσοδο έναν τυχαίο αριθμό. Από τα παραχθέντα κλειδιά, το δημόσιο γνωστοποιείται σε τρίτους, ενώ το ιδιωτικό παραμένει μυστικό.



Σχήμα 2.1: Παραγωγή ασύμμετρου ζεύγους κλειδιών

Ο χρήστης μπορεί να χρησιμοποιήσει τα κλειδιά για δύο βασικούς σκοπούς:

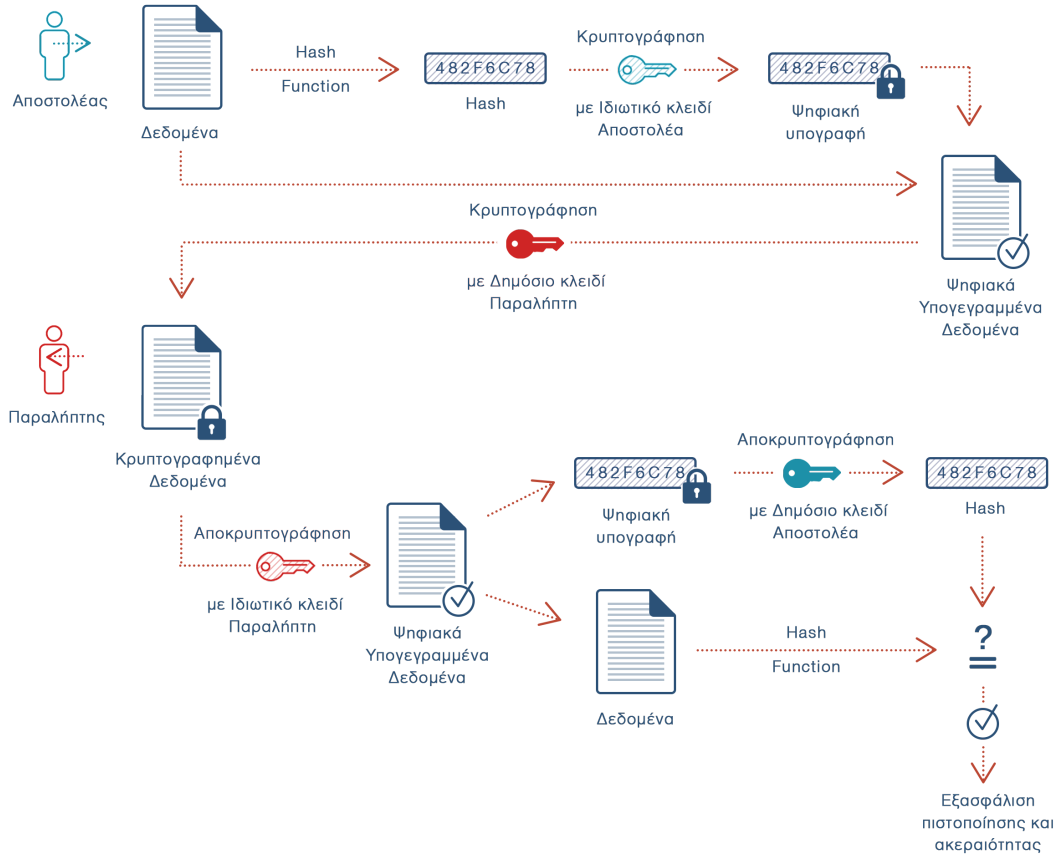
α) Για να αποκρυπτογραφήσει μηνύματα άλλων χρηστών, οι οποίοι τα κρυπτογράφησαν χρησιμοποιώντας το δημόσιο κλειδί του. Με αυτόν τον τρόπο εξασφαλίζεται η *εμπιστευτικότητα* (confidentiality).

β) Για να υπογράψει ψηφιακά ένα μήνυμα, κρυπτογραφώντας το hash των δεδομένων του με το ιδιωτικό του κλειδί. Έτσι, ο παραλήπτης του μηνύματος μπορεί μέσω της ληφθείσας *ψηφιακής υπογραφής* (digital signature):

   i. Να επαληθεύσει την ταυτότητα του αποστολέα, αποκρυπτογραφώντας επιτυχώς την ψηφιακή υπογραφή με το δημόσιο κλειδί του τελευταίου. Εξασφαλίζεται έτσι η *πιστοποίηση* (authenticity) της προέλευσης των δεδομένων.

   ii. Να επιβεβαιώσει ότι το μήνυμα έφτασε αναλλοίωτο, εφόσον το hash των δεδομένων συμπίπτει με το hash εντός της ψηφιακής υπογραφής. Με αυτόν τον τρόπο εξασφαλίζεται η *ακεραιότητα* (integrity) των δεδομένων.

Με τον συνδυασμό των παραπάνω, λέμε ότι δύο χρήστες μπορούν να επικοινωνούν μεταξύ τους με *κρυπτογράφηση απ' άκρη σ' άκρη* (end to end encryption).



Σχήμα 2.2: Κρυπτογράφηση απ' άκρη σ' άκρη

2.3 Δένδρα Merkle

Ένα δένδρο Merkle (Merkle tree ή hash tree) είναι μία δενδρική δομή δεδομένων, η οποία απαρτίζεται από φύλλα (leaf nodes), που περιέχουν hashes από blocks δεδομένων, και από άλλους κόμβους (non-leaf nodes), οι οποίοι περιέχουν τα hashes των θυγατρικών τους. Στην κορυφή του δένδρου βρίσκεται το λεγόμενο root hash[1].

Η πιο συνηθισμένη υλοποίηση είναι το δυαδικό (binary) δένδρο Merkle, το οποίο περιλαμβάνει δύο θυγατρικούς κόμβους (child nodes) κάτω από κάθε γονικό non-leaf κόμβο, και είναι αυτό που αναλύεται στη συνέχεια.

Τα Merkle trees επιτρέπουν την αποδοτική και ασφαλή επαλήθευση των περιεχομένων που ανήκουν σε σετ δεδομένων μεγάλου μεγέθους. Η βασική ιδιότητα είναι ότι για κάθε σετ δεδομένων υπάρχει ακριβώς ένα πιθανό δένδρο, το οποίο δε γίνεται να τροποποιηθεί χωρίς να αλλάξει ταυτόχρονα και το root hash.

Έτσι, μέσω των λεγόμενων Merkle proofs, μπορούμε:

- Να αποφανθούμε εάν κάποια δεδομένα ανήκουν στο δένδρο (με τον αριθμό των hashes που θα πρέπει να υπολογιστούν να είναι ανάλογος του λογαρίθμου του αριθμού των leaf nodes).

- Να αποδείξουμε συνοπτικά την εγκυρότητα ενός τμήματος κάποιου σετ δεδομένων, χωρίς

να χρειαστεί να αποθηκεύσουμε ολόκληρο το σύνολο δεδομένων.

- Να διασφαλίσουμε την εγκυρότητα ενός συγκεκριμένου συνόλου δεδομένων εντός ενός μεγαλύτερου σύνολου, χωρίς να χρειαστεί να αποκαλυφθεί το περιεχόμενο οποιουδήποτε εκ των δύο[2].

2.4 Δίκτυα Ομότιμων Κόμβων

Τα δίκτυα ομότιμων κόμβων ή Peer-to-Peer (P2P) networks αποτελούν μία κατανεμημένη αρχιτεκτονική δικτύων, οι συμμετέχοντες (κόμβοι) της οποίας μοιράζονται ένα τμήμα των πόρων τους, με στόχο την παροχή κάποιας υπηρεσίας (π.χ. τον διαμοιρασμό περιεχομένου). Εν αντιθέσει με συγκεντρωτικά δίκτυα τύπου client/server, οι κόμβοι (nodes) έχουν απευθείας πρόσβαση στους πόρους, χωρίς τη διαμεσολάβηση ενδιάμεσων οντοτήτων. Οι συμμετέχοντες ενός τέτοιου δικτύου είναι, δηλαδή, ταυτόχρονα, τόσο πάροχοι, όσο και αιτούντες των πόρων και της παρεχόμενης υπηρεσίας.[3]

Τα P2P networks μπορούν να χωριστούν σε δύο κατηγορίες:

- Στα «Καθαρά» (Pure) P2P networks, στα οποία ισχύει ότι η αφαίρεση ενός τυχαίου κόμβου από το δίκτυο δεν προκαλεί κάποιο πρόβλημα σε αυτό.
- Στα «Υβριδικά» (Hybrid) P2P networks, στα οποία συμμετέχουν επιπλέον και κεντρικές οντότητες, παρέχοντας απαραίτητα τμήματα των προσφερόμενων υπηρεσιών.

Από εδώ και στο εξής, εάν δεν αναφέρεται ρητά η κατηγορία κάποιου P2P network, θα εννοείται ότι ανήκει στην πρώτη.

2.5 Blockchain

Το blockchain αποτελεί μία διανεμημένη δημόσια σειρά δεδομένων, που διατηρεί έναν αμετάβλητο ως προς το ιστορικό του κατάλογο (immutable ledger) ψηφιακών συναλλαγών (digital transactions) ενός αγαθού (asset), π.χ. ενός νομίσματος (token). Περιγράφηκε για πρώτη φορά το 2008 από ένα άτομο (ή μία ομάδα ανθρώπων) γνωστό ως Satoshi Nakamoto, αποτελώντας τη βάση του κρυπτονομίσματος (cryptocurrency) Bitcoin.[4]

Δομικό στοιχείο του blockchain είναι το μπλοκ (block), το οποίο περιέχει μία ομάδα έγκυρων συναλλαγών που έχουν κατακερματιστεί και κωδικοποιηθεί σε ένα δένδρο Merkle, το hash του προηγούμενου μπλοκ και μερικά ακόμα μεταδεδομένα (π.χ. nonce, timestamp). Έτσι, κάθε νέο μπλοκ «δείχνει» στο προηγούμενό του μέσω του hash, επιβεβαιώνοντας την ακεραιότητά του, με τα διαδεχόμενα μπλοκ να σχηματίζουν τελικά μία αλυσίδα, μέχρι το αρχικό μπλοκ, το οποίο είναι γνωστό ως το μπλοκ γένεσης (genesis block).[5]

Ως προς την κυριότητα επί αυτού, το blockchain συνήθως[1] δεν ελέγχεται από κάποια κεντρική οντότητα, αλλά διατηρείται από ένα δημόσιο P2P δίκτυο. Οι κόμβοι (nodes) του δικτύου συμμορφώνονται συλλογικά με ένα πρωτόκολλο συναίνεσης (consensus) για την επικοινωνία και την επικύρωση νέων μπλοκ. Για παράδειγμα, στο Bitcoin, το consensus επιτυγχάνεται μέσω ενός Proof of Work (PoW) αλγορίθμου, όπου οι κόμβοι (miners) ανταγωνίζονται ο ένας τον άλλον για το ποιος θα λύσει πρώτος ένα σύνθετο αλγοριθμικό πρόβλημα που συσχετίζεται με το εκάστοτε block. Αυτός που θα τα καταφέρει επιβραβεύεται για την επεξεργαστική ισχύ που

[1] Υπάρχουν και κάποιες υλοποιήσεις ιδιωτικών blockchain που, όμως, δε θα μας απασχολήσουν.

δαπάνησε με ένα ποσό από bitcoin. Εκείνα είναι εν μέρει νέα νομίσματα που κόβονται ή «εξορύσσονται» εκείνη τη στιγμή (όπως ορίζεται από το πρωτόκολλο), αλλά και όσα τέλη (fees) κατέβαλαν οι κόμβοι για να πραγματοποιήσουν τις συναλλαγές του μπλοκ. Αξίζει να σημειωθεί πως δεν είναι αναγκαίο να διαθέτει κανείς ολόκληρο το blockchain (το οποίο είναι ογκώδες) - δηλαδή έναν πλήρη κόμβο - για να επικοινωνήσει με το δίκτυο, αλλά αρκεί ένας light node που απλά αναμεταδίδει την συναλλαγή που επιθυμεί να πραγματοποιήσει ο χρήστης.

Η διευθυνσιοδότηση σε ένα blockchain επιτυγχάνεται αξιοποιώντας την κρυπτογραφία δημόσιου κλειδιού. Το πρωτόκολλο του εκάστοτε blockchain ορίζει έναν αλγόριθμο για την παραγωγή ζευγών κλειδιών (π.χ. ECDSA στο Bitcoin). Το δημόσιο από αυτά ορίζει τη διεύθυνση, ενώ το ιδιωτικό παραμένει μυστικό, υπό την κατοχή του χρήστη. Με αυτό τον τρόπο προκύπτει ένα πρακτικά ανεξάντλητο πλήθος πιθανών έγκυρων δημόσιων διευθύνσεων (π.χ. 2^{160} για το Bitcoin), στις οποίες οι χρήστες μπορούν να στέλνουν και να λαμβάνουν ποσά του εκάστοτε κρυπτονομίσματος. Για την αποστολή ενός ποσού, δηλώνουν το fee που επιθυμούν να καταβάλουν και υπογράφουν την επιθυμητή συναλλαγή με το ιδιωτικό τους κλειδί. Η συναλλαγή αναμεταδίδεται στο δίκτυο και παραμένει στο transaction pool μέχρις ότου να γίνει αποδεκτή και να συμπεριληφθεί στο επόμενο block.

Από τεχνική σκοπιά, το blockchain μπορεί να θεωρηθεί ως μία μηχανή καταστάσεων βασισμένη σε συναλλαγές (transaction-based state machine). Δηλαδή, ξεκινάει από μία αρχική κατάσταση (genesis state), η οποία τροποποιείται σταδιακά με κάθε block, και περιλαμβάνει ανά πάσα στιγμή τις διευθύνσεις με τα ποσά των νομισμάτων που τις αντιστοιχούν.

2.6 Ethereum

Σχήμα 2.3: Ethereum logo

Το Ethereum είναι ένα δημόσιο blockchain ανοιχτού κώδικα με εγγενές κρυπτονόμισμα το Ether (ETH). Παρέχει μία προγραμματιστική πλατφόρμα με ενσωματωμένη μία Turing-complete γλώσσα προγραμματισμού, που μπορεί να χρησιμοποιηθεί για τη δημιουργία αποκεντρωμένων εφαρμογών (Decentralized Applications ή DApps) μέσω της χρήσης «έξυπνων συμβολαίων» (smart contracts).[6]

2.6.1 Smart Contracts & DApps

Με απλά λόγια, τα smart contracts αποτελούν αυτόνομα κομμάτια κώδικα τα οποία είναι αποθηκευμένα στο blockchain και ενεργοποιούνται μέσω συναλλαγών. Μπορούν να διαβάζουν και να γράφουν δεδομένα επί του blockchain, κληρονομώντας ιδιότητες όπως τη διαφάνεια (transparency), την εγκυρότητα (validability) και την αμεταβλητότητα (immutability).

Ένα παράδειγμα της καθημερινότητας που μπορεί να παρομοιασθεί λειτουργικά με smart contract είναι ο αυτόματος πωλητής[7]. Ένας αυτόματος πωλητής ορίζεται ως ένα αυτόνομο μηχάνημα που διανέμει αγαθά ή παρέχει υπηρεσίες όταν εισάγονται σε αυτόν κέρματα ή κάποια ηλεκτρονική πληρωμή. Οι αυτόματοι πωλητές είναι προγραμματισμένοι να εκτελούν συγκεκριμένους κανόνες που θα μπορούσαν να οριστούν σε ένα συμβόλαιο.

Με όμοιο τρόπο, σε ένα smart contract μπορούν να κωδικοποιηθούν αυθαίρετες συναρτήσεις μετάβασης, επιτρέποντας τη δημιουργία ποικίλων αποκεντρωμένων εφαρμογών. Οι εφαρμογές αυτές μπορούν να χωριστούν σε τρεις κατηγορίες:

- Οικονομικές εφαρμογές, οι οποίες παρέχουν στους χρήστες ισχυρότερους τρόπους διαχείρισης και σύναψης συμβάσεων χρησιμοποιώντας τα χρήματά τους. Αυτό περιλαμβάνει υπο-νομίσματα, χρηματοοικονομικά παράγωγα, συμβάσεις αντιστάθμισης κινδύνου, πορτοφόλια αποταμίευσης, διαθήκες, και, τελικά, ακόμη και ορισμένες κατηγορίες συμβάσεων εργασίας πλήρους κλίμακας.

- Ημι-οικονομικές εφαρμογές, όπου εμπλέκονται χρήματα, αλλά η λειτουργία τους εμπεριέχει παράλληλα και μία αξιοσημείωτη μη νομισματική πλευρά. Ένα τέτοιο παράδειγμα είναι οι αυτόματες πληρωμές για λύσεις σε υπολογιστικά προβλήματα (βλ. Gitcoin).

- Μη οικονομικές εφαρμογές, όπως η αποκεντρωμένη αποθήκευση δεδομένων, συστήματα ταυτότητας (identity) και φήμης (reputation), διαδικτυακές ψηφοφορίες και αποκεντρωμένη διακυβέρνηση. Οι τελευταίες εισάγουν και την έννοια των Αποκεντρωμένων Αυτόνομων Οργανώσεων (Decentralized Autonomous Organizations ή DAOs), οντοτήτων που ενεργούν αυτόνομα, χωρίς την ανάγκη διαμεσολάβησης κάποιας συγκεντρωτικής (centralized) αρχής.[6]

2.6.2 EVM

Τα smart contracts εκτελούνται από την εικονική μηχανή του Ethereum (Ethereum Virtual Machine ή EVM). Η EVM αποτελεί μία quasi[2]–Turing-complete μηχανή καταστάσεων αρχιτεκτονικής βασισμένης σε στοίβα (stack-based architecture). Σε υψηλό επίπεδο, η EVM μπορεί να θεωρηθεί ως ένας παγκόσμιος αποκεντρωμένος υπολογιστής που περιέχει εκατομμύρια εκτελέσιμα αντικείμενα, το καθένα με τη δική του μόνιμη αποθήκη δεδομένων.

Η EVM αποθηκεύει όλες τις τιμές της μνήμης σε μια στοίβα και λειτουργεί με μέγεθος λέξης 256 bit, κυρίως για τη διευκόλυνση των εγγενών λειτουργιών κατακερματισμού και ελλειπτικής καμπύλης. Διαθέτει ένα σύνολο διευθυνσιοδοτήσιμων στοιχείων δεδομένων:

- Έναν αμετάβλητο κώδικα προγράμματος σε εικονική μνήμη ROM, φορτωμένο με τον bytecode του smart contract προς εκτέλεση.

- Μία πτητική (volatile) μνήμη, με κάθε θέση ρητά αρχικοποιημένη στο μηδέν.

- Ένας χώρος μόνιμης αποθήκευσης, που είναι μέρος του Ethereum state, επίσης αρχικά μηδενισμένος.

Υπάρχει, επίσης, ένα σύνολο μεταβλητών περιβάλλοντος και δεδομένων, που είναι διαθέσιμα κατά την εκτέλεση.[8]

[2] «quasi» επειδή όλες οι διαδικασίες εκτέλεσης περιορίζονται σε έναν πεπερασμένο αριθμό υπολογιστικών βημάτων από την ποσότητα gas που είναι διαθέσιμη για οποιαδήποτε εκτέλεση ενός smart contract.

2.7 IPFS

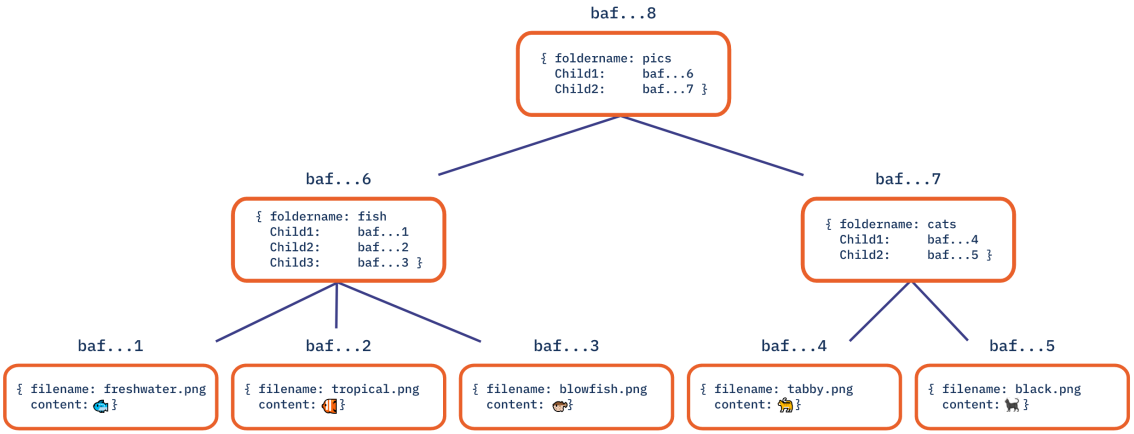
Σχήμα 2.4: IPFS logo

Το IPFS (InterPlanetary File System) είναι *ένα P2P πρωτόκολλο υπερμέσων, σχεδιασμένο για να διατηρήσει και να αυξήσει τη γνώση της ανθρωπότητας κάνοντας το διαδίκτυο αναβαθμίσιμο, ανθεκτικό και πιο ανοιχτό.*[9] Πρακτικά πρόκειται για ένα κατανεμημένο σύστημα για αποθήκευση και πρόσβαση σε αρχεία, ιστότοπους, εφαρμογές και δεδομένα. Το περιεχόμενο είναι προσβάσιμο μέσω ενός δικτύου ομότιμων κόμβων που βρίσκονται οπουδήποτε στον κόσμο, οι οποίοι ενδέχεται να να αποθηκεύουν πληροφορία, να τη μεταφέρουν (relay nodes) ή και τα δύο.[10]

Ο τρόπος λειτουργίας του IPFS βασίζεται στα εξής:

- **Μοναδική ταυτοποίηση μέσω διευθυνσιοδότησης περιεχομένου (content addressing)**. Το περιεχόμενο δεν προσδιορίζεται από την τοποθεσία του (π.χ. https://...), αλλά από το τι περιλαμβάνει. Κάθε κομμάτι περιεχομένου έχει ένα μοναδικό αναγνωριστικό (Content ID ή CID), το οποίο είναι το hash του σε μορφή multihash (`https://multiformats.io/multihash/`).

- **Σύνδεση περιεχομένου μέσω κατευθυνόμενων άκυκλων γράφων (Directed Acyclic Graphs ή DAGs)**. Το IPFS αξιοποιεί DAGs (και συγκεκριμένα Merkle DAGs), μίας δομής δεδομένων της οποίας κάθε κόμβος έχει ως μοναδικό αναγνωριστικό το hash του περιεχομένου του (το CID).



Σχήμα 2.5: Merkle DAG[11]

Στο παραπάνω Merkle DAG τα baf...i αποτελούν τα CID των αρχείων και των φακέλων. Το δένδρο δημιουργείται από κάτω προς τα πάνω, υπολογίζοντας κάθε φορά τα κατάλληλα hashes/CIDs. Σε περίπτωση που το περιεχόμενο ενός κόμβου αλλάξει, τότε αλλάζει τόσο το CID του, όσο και τα CIDs όλων των προγόνων του.

- **Ανακάλυψη περιεχομένου μέσω κατανεμημένων πινάκων κατακερματισμού (Distributed Hash Tables ή DHTs)**. Ο DHT είναι ένα κατανεμημένο σύστημα για την αντιστοίχιση κλειδιών σε τιμές. Στο IPFS αποτελεί το θεμελιώδες συστατικό του συστήματος

δρομολόγησης περιεχομένου και λειτουργεί ως διασταύρωση μεταξύ καταλόγου και συστήματος πλοήγησης. Πρακτικά πρόκειται για ένα πίνακα που αποθηκεύει ποιος έχει ποια δεδομένα και, μέσω του οποίου, ο χρήστης βρίσκει τον peer που έχει αποθηκευμένο το επιθυμητό περιεχόμενο.

Κάτι που θα πρέπει να σημειωθεί είναι πως, σαν προεπιλογή, οι IPFS κόμβοι αντιμετωπίζουν τα αποθηκευμένα δεδομένα ως προσωρινή μνήμη (cache), πράγμα που σημαίνει ότι δεν υπάρχει καμία εγγύηση ότι εκείνα θα συνεχίσουν να παραμένουν αποθηκευμένα σε αυτούς. Για την αποφυγή της διαγραφής τους από τον garbage collector, τα δεδομένα θα πρέπει να σημανθούν ως σημαντικά, μέσω του «καρφιτσώματος» (pinning). Έτσι, για την ομαλή λειτουργία π.χ. μίας DApp που βασίζεται στο IPFS, θα πρέπει το περιεχόμενό της να είναι pinned από αρκετούς peers και/ή να γίνεται χρήση κάποιου pinning service, ώστε να εξασφαλίζεται διαθεσιμότητά του.

Βασικό πλεονέκτημα του IPFS είναι ότι ο αποκεντρωτικός του χαρακτήρας δίνει τη δυνατότητα να παρέχεται το περιεχόμενο από πολλούς κόμβους, οι οποίοι βρίσκονται σε διαφορετικές τοποθεσίες και δεν υπάγονται σε κάποια συγκεκριμένη κεντρική αρχή. Με αυτόν τον τρόπο, τα δεδομένα είναι πιο ανθεκτικά τόσο από άποψη διαθεσιμότητας (αν ένας κόμβος αποσυνδεθεί, θα υπάρχει κάποιος άλλος), όσο και από άποψη αντοχής στη λογοκρισία. Μπορεί, επίσης, να ανακτώνται γρηγορότερα, εφόσον τα διαθέτουν κάποιοι κοντινοί peers, πράγμα ιδιαίτερα πολύτιμο εάν για κοινότητες που είναι καλά δικτυωμένες τοπικά αλλά δεν έχουν καλή σύνδεση με το ευρύτερο διαδίκτυο.

2.8 OrbitDB

Σχήμα 2.6: OrbitDB logo

Η OrbitDB είναι μία P2P βάση δεδομένων ανοιχτού κώδικα. Χρησιμοποιεί το IPFS για την αποθήκευση των δεδομένων και το IPFS Pubsub για τον αυτόματο συγχρονισμό των βάσεων δεδομένων μεταξύ των peers. Είναι τελικά συνεπής (eventually consistent) και χρησιμοποιεί CRDTs (Conflict-Free Replicated Data Types) για συγχωνεύσεις βάσεων δεδομένων χωρίς συγκρούσεις, πράγμα που την καθιστά εξαιρετική επιλογή για DApps και offline-first web applications.[12]

Κάποια Βασικά χαρακτηριστικά της είναι τα εξής:

- **Stores**: Η OrbitDB παρέχει διάφορους τύπους βάσεων (stores) για διαφορετικά μοντέλα δεδομένων και περιπτώσεις χρήσης:
  - log: ένα αμετάβλητο (μόνο για προσάρτηση) ημερολόγιο με ανιχνεύσιμο ιστορικό.
  - feed: ένα μεταβλητό αρχείο καταγραφής με ανιχνεύσιμο ιστορικό, του οποίου οι καταχωρήσεις μπορούν να προστεθούν και να αφαιρεθούν.
  - keyvalue: μία βάση δεδομένων κλειδιών-τιμών.
  - docs: μία βάση δεδομένων εγγράφων στην οποία μπορούν να αρχειοθετηθούν έγγραφα JSON με ένα καθορισμένο κλειδί.
  - counter: μία βάση δεδομένων για καταμέτρηση συμβάντων.

Όλα τα stores υλοποιούνται πάνω στο `ipfs-log`, μία αμετάβλητη, operation-based CRDT για κατανεμημένα συστήματα, ενώ υπάρχει και η δυνατότητα δημιουργίας προσαρμοσμένων stores ανάλογα με την περίπτωση.

- **Address**: Κάθε βάση δεδομένων λαμβάνει κατά τη δημιουργία της μία διεύθυνση της μορφής:
`/orbitdb/CID/DATABASE_NAME`, όπου CID είναι το IPFS multihash του μανιφέστου της και DATABASE_NAME το όνομα της βάσης[13]. Το μανιφέστο είναι ένα IPFS object που περιέχει πληροφορίες της βάσης όπως το όνομα, τον τύπο και έναν δείκτη στον ελεγκτή πρόσβασης (access controller).

- **Identity**: Κάθε φορά που προστίθεται μία εγγραφή στη βάση υπογράφεται από τον δημιουργό της, ο οποίος προσδιορίζεται από μία ταυτότητα (identity). Το Identity object, πέρα από τον προεπιλεγμένο τρόπο λειτουργίας, μπορεί να προσαρμοστεί έτσι ώστε να συνδέεται με κάποιο εξωτερικό αναγνωριστικό. Η μορφή του έχει ως εξής (βλ. και `https://github.com/orbitdb/orbit-db-identity-provider`):

```
{
   _id: '<the ID of the external identity>',
   // Auto-generated by OrbitDB
   _publicKey: '<signing key used to sign OrbitDB entries>',
   signatures: {
     //Allows the owner of id to prove they own the private key
     associated with publicKey
     id: '<signature of _id signed using publicKey>',
     //This links the two ids
     publicKey: '<signature of signatures.id + _publicKey using _id>'
   },
   type: 'orbitdb'
}
```

Σχήμα 2.7: OrbitDB Identity

- **Access Control**: Κατά τη δημιουργία μίας βάσης μπορούν να οριστούν όσοι θα έχουν δικαίωμα να γράψουν σε αυτή μέσω ενός ελεγκτή πρόσβασης (access controller). Ο ελεγκτής θα περιλαμβάνει τα public keys τους, τα οποία μπορούν να ανακτηθούν από το identity του καθενός. Από προεπιλογή και αν δεν ορίζεται διαφορετικά, δίνεται πρόσβαση εγγραφής μόνο στον δημιουργό της βάσης.

Κεφάλαιο 3

Σχεδίαση εφαρμογής

3.1 Λογικά μέρη

Η πλατφόρμα μπορεί να διαχωριστεί σε δύο λογικά μέρη:

1) Το πρώτο μέρος αποτελεί μία αυτοτελή και πλήρως αποκεντρωτική πλατφόρμα που στόχος της είναι να παρέχει τη δυνατότητα ελευθερίας λόγου απρόσβλητου σε λογοκρισία και διαγραφή από κεντρικές οντότητες εξουσίας. Στην ουσία εδώ θα μπορεί - σε μια απλοποιημένη εκδοχή - οποιοσδήποτε να δημιουργήσει topics ή να απαντήσει σε άλλα. Επεξεργαστικός πυρήνας θα είναι ένα smart contract το οποίο θα δέχεται από τους χρήστες transactions και θα τα εγγράφει στο Storage Layer:

Το μειονέκτημα αυτού του κομματιού είναι πως για να λειτουργήσει απαιτεί για κάθε transaction οι χρήστες να καταβάλουν κάποιο τέλος για τα fees. Αν και υπάρχουν σχέδια από την ομάδα ανάπτυξης του Ethereum για τη μείωσή τους στο μέλλον (έως και 10-2), τα fees στη χρήση της EVM θα είναι αναπόφευκτα. Ωστόσο θα πρέπει να σημειωθεί ότι αφενός παρέχουν μια μορφή προστασίας απέναντι σε κακόβουλους χρήστες που θα πλημμύριζαν την εφαρμογή με ανεπιθύμητο ποσοτικά/ποιοτικά περιεχόμενο (θα τους ήταν οικονομικά ασύμφορη μια τέτοια επίθεση), αφετέρου υπάρχουν κάποια workarounds για να μειωθεί δραστικά η εμπλοκή του χρήστη με την καταβολή τελών, κάτι που περιγράφεται παρακάτω στις κατηγορίες χρηστών. Τα παραπάνω πρακτικά σημαίνουν ότι το Smart Contract θα πρέπει ανά διαστήματα να φορτίζεται (από οποιονδήποτε) με Ethereum ή οι χρήστες (βλ. κατηγορίες χρηστών) να καταβάλουν τα δικά τους έξοδα.

2) Το δεύτερο μέρος αποτελεί μια μερικώς αποκεντρωτική και μη αυτοτελή πλατφόρμα που έρχεται να λειτουργήσει επιπρόσθετα στην πρώτη. Το κομμάτι αυτό απευθύνεται αποκλειστικά σε επικυρωμένα μέλη του ΑΠΘ και συνιστά ένα αμεσοδημοκρατικό σύστημα ψηφοφορίας που θα εγγυάται σε υψηλό βαθμό την εγκυρότητα και την ανωνυμία των διαδικασιών του. Με λίγα λόγια, θα δημιουργούνται θέματα προς ψηφοφορία (στο πρώτο κομμάτι), πάνω στα οποία θα ψηφίζουν όσοι έχουν το δικαίωμα (αυτοί θα ορίζονται με την κατοχή ενός ανάλογου token στο Ethereum).

Ο χρήστης μέσω ενός Frontend (μιας κλασικής ιστοσελίδας ουσιαστικά) θα μπορεί να πιστοποιήσει μέσω login στο it.auth.gr* την ακαδημαϊκή του ταυτότητα. Στη συνέχεια το TDS (Token Distribution Service, ελέγχοντας το admin account των tokens θα παρέχει στο χρήστη δύο tokens. Ένα το οποίο θα του δίνει voting rights (verified user) και ένα που θα κάνει τη πλατφόρμα να του επιστρέφει τα τέλη τα οποία πληρώνουν σε κάθε post τους (trusted user).

*Ιδανικά, με τη συνεργασία του ΑΠΘ, το UAS θα αποτελεί υπηρεσία συνεργαζόμενη με την Υποδομή πιστοποίησης και εξουσιοδότησης (βλ. https://it.auth.gr/el/infrastructure/aai). Αυτό σημαίνει ότι οι χρήστες δε θα χρειάζεται να εμπιστευτούν τον UAS με τα it credential τους.

3.2 Κατηγορίες χρηστών

3.2.1 Πρώτο μέρος

Όπως προειπώθηκε, το πρώτο μέρος της πλατφόρμας θα είναι ανοιχτό σε όλους. Ωστόσο, οι χρήστες του μπορούν να διακριθούν στις εξής δύο κατηγορίες:

- Έμπιστα μέλη του δικτύου (trusted users)
- Μη έμπιστα μέλη (untrusted users)

Βασική διαφορά των δύο κατηγοριών είναι πως ενώ οι trusted users θα αποζημιώνονται αυτόματα από το (ενν. φορτισμένο) smart contract και άρα θα είναι απαλλαγμένοι από τέλη, οι δεύτεροι θα πρέπει να καταβάλουν μόνοι τους τα τέλη τους (fees) για κάθε ενέργεια (π.χ. posting).

Η εμπιστοσύνη ενός χρήστη (trust) μπορεί να οριστεί ως ένας ακέραιος αριθμός με κάποιο κατώφλι, πάνω από το οποίο ο χρήστης θα θεωρείται trusted. Το trust θα μπορεί να αυξομειώνεται ανά πάσα στιγμή, με τους χρήστες να δίνουν και να παίρνουν βαθμούς trust σε/από άλλους.

Αξίζει να σημειωθεί επίσης ότι επειδή όλες οι συναλλαγές καταγράφονται, είναι δυνατό να υπολογιστεί το συνολικό ποσό του Ethereum που έχει ξοδέψει ένας common user μέχρι να γίνει trusted και έτσι μπορεί σταδιακά να του επιστραφεί μέσω του contract. Αυτό δίνει κίνητρο στους χρήστες να διατηρούν σωστή συμπεριφορά και να συμβάλλουν θετικά στην πλατφόρμα.

3.2.2 Δεύτερο μέρος

Για το δεύτερο μέρος της πλατφόρμας, οι χρήστες του μπορούν να διακριθούν στις εξής κατηγορίες:

- Πιστοποιημένα μέλη του Α.Π.Θ. (verified users)
- Μη πιστοποιημένα μέλη του Α.Π.Θ. (untrusted users)

Σχετικά με το δικαίωμα ψήφου για αυθεντικές δημοκρατικές αποφάσεις σχετικές με το ΑΠΘ, αυτό θα το έχουν μόνο οι verified χρήστες του δικτύου. Οι verified χρήστες θα μπορούν επιπλέον να αλληλεπιδρούν με την πλατφόρμα εξαρχής χωρίς την καταβολή τελών (θα ξεκινάνε ως trusted), πράγμα που βέβαια δε σημαίνει ότι χάνοντας trust δε θα μπορούν να χάσουν αυτό το προνόμιο.

3.2.3 Σύνοψη χρηστών

Συμπερασματικά προκύπτουν τέσσερις διακριτές κατηγορίες χρηστών με ξεχωριστά δικαιώματα που φαίνονται στο παρακάτω διάγραμμα:

3.3 Απαιτήσεις λογισμικού

Στην παρούσα ενότητα περιγράφονται οι βασικές απαιτήσεις λογισμικού (software requirements) της εφαρμογής.

Η πρώτη κατηγορία είναι αυτή των Λειτουργικών Απαιτήσεων (ΛΑ), η οποία αναφέρεται στη συμπεριφόρα του συστήματος, δηλαδή στον τρόπο που θα αντιδρά και στις εξόδους που θα παράγει ανάλογα με τις εισόδους.

<ΛΑ-1> Ο χρήστης πρέπει να μπορεί να εγγραφεί στην εφαρμογή με τον Ethereum λογαριασμό του.

Περιγραφή: Ο χρήστης πρέπει να μπορεί να εγγραφεί στην εφαρμογή, πατώντας το κουμπί «Sign Up» και συμπληρώνοντας τα απαραίτητα πεδία σύμφωνα με τις οδηγίες. Το πεδίο «Username» είναι υποχρεωτικό να συμπληρωθεί και ορίζεται με μοναδικό τρόπο. Σε περίπτωση που ο χρήστης εισάγει μη διαθέσιμο Username, το σύστημα θα πρέπει να μην επιτρέπει στον χρήστη να συνεχίσει και να προβάλει αντίστοιχο μήνυμα λάθους. Τα πεδία «Profile picture URL» και «Location» είναι προαιρετικά.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-2> Ο χρήστης πρέπει να συνδέεται αυτόματα, εφόσον είναι εγγεγραμμένος.

Περιγραφή: Το σύστημα πρέπει να διαπιστώνει αυτόματα εάν το τρέχον Ethereum address έχει λογαριασμό στην εφαρμογή και εάν ναι, να συνδέει να τον χρήστη, ανακτώντας το Username του από το blockchain και προβάλλοντάς το στο μενού.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-3> Το σύστημα πρέπει να δημιουργεί τις βάσεις δεδομένων του χρήστη.

Περιγραφή: Το σύστημα πρέπει να δημιουργεί τις βάσεις δεδομένων του χρήστη, εάν αυτές δεν υπάρχουν ήδη τοπικά. Όταν ο χρήστης ξεκλειδώσει τον Ethereum λογαριασμό του, το σύστημα θα πρέπει να τον προτρέπει να υπογράψει με το ιδιωτικό του κλειδί μία συναλλαγή που θα εξασφαλίζει τη γνησιότητα των βάσεών του και των δεδομένων που αυτές θα εμπεριέχουν.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-4> Ο εγγεγραμμένος χρήστης πρέπει να μπορεί να δημιουργεί θέματα (topics).

Περιγραφή: Ο εγγεγραμμένος χρήστης πρέπει να μπορεί να δημιουργεί νέα θέματα. Αυτό το επιτυγχάνει πατώντας το κουμπί «New Topic», συμπληρώνοντας τα υποχρεωτικά πεδία της φόρμας («Topic subject» και «First post content»), πατώντας το κουμπί «Create Topic» και επιβεβαιώνοντας τη συναλλαγή στο Ethereum.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-5> Ο χρήστης πρέπει να μπορεί να περιηγείται σε θέματα.

Περιγραφή: Ο χρήστης πρέπει να μπορεί να περιηγείται σε θέματα, πατώντας σε κάποιο θέμα της αρχικής οθόνης και, έπειτα, χρησιμοποιώντας τα βέλη, να περιηγηθεί στο ιστορικό των μηνυμάτων του θέματος.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-6> Ο εγγεγραμμένος χρήστης πρέπει να μπορεί να δημιουργεί μηνύματα (posts).

Περιγραφή: Ο εγγεγραμμένος χρήστης πρέπει να μπορεί να δημιουργεί μηνύματα στο θέμα που επιθυμεί. Αυτό επιτυγχάνεται συμπληρώνοντας το πεδίο νέου μηνύματος στην οθόνη του θέματος, πατώντας το κουμπί «Post» και επιβεβαιώνοντας τη συναλλαγή στο Ethereum.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-7> Ο χρήστης πρέπει να μπορεί να τροποποιεί τα μηνύματά του.

Περιγραφή: Ο χρήστης πρέπει να μπορεί να τροποποιεί τα μηνύματά του. Αυτό το επιτυγχάνει πατώντας το κουμπί επεξεργασίας στο εκάστοτε μήνυμα, τροποποιώντας το μήνυμα και πατώντας το κουμπί επιβεβαίωσης. Στη συνέχεια, το σύστημα τροποποιεί το περιεχόμενο του μηνύματος στη βάση δεδομένων του χρήστη ανάλογα. Σε περίπτωση που ο χρήστης αλλάξει γνώμη κατά τη διάρκεια της διαδικασίας της επεξεργασίας, μπορεί να πατήσει το κουμπί ακύρωσης και να αναιρέσει τις αλλαγές που πραγματοποίησε.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-8> Ο χρήστης πρέπει να μπορεί να διαγράφει τα μηνύματά του.

Περιγραφή: Ο χρήστης πρέπει να μπορεί να διαγράφει τα μηνύματά του. Αυτό το επιτυγχάνει πατώντας το κουμπί διαγραφής στο εκάστοτε μήνυμα. Στη συνέχεια, το σύστημα αφαιρεί το περιεχόμενο του μηνύματος από τη βάση δεδομένων του χρήστη.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-9> Ο εγεγγραμμένος χρήστης πρέπει να μπορεί να ψηφίζει σε μηνύματα άλλων χρηστών.

Περιγραφή: Ο εγεγγραμμένος χρήστης πρέπει να μπορεί να υπερψηφίζει ή να καταψηφίζει μηνύματα άλλων χρηστών. Αυτό το επιτυγχάνει πατώντας τα παρακείμενα κουμπιά «+» ή «-» αντίστοιχα και επιβεβαιώνοντας τη συναλλαγή στο Ethereum (οι ψήφοι αποθηκεύονται εκεί). Η διαδικασία ισχύει και για την τροποποίηση ή την αφαίρεση μίας ψήφου από τον χρήστη.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-10> Ο εγεγγραμμένος χρήστης πρέπει να μπορεί να δημιουργεί ψηφοφορίες (polls).

Περιγραφή: Ο εγεγγραμμένος χρήστης πρέπει να μπορεί να δημιουργεί ψηφοφορίες. Αυτό το επιτυγχάνει πατώντας «Add Poll» στην οθόνη δημιουργία θέματος και συμπληρώνοντας τα απαραίτητα πεδία.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-11> Ο εγεγγραμμένος χρήστης πρέπει να μπορεί να ψηφίζει σε ψηφοφορίες.

Περιγραφή: Ο εγεγγραμμένος χρήστης πρέπει να μπορεί να ψηφίζει σε ψηφοφορίες, σύμφωνα με τους εκάστοτε κανόνες.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-12> Ο χρήστης πρέπει να μπορεί να διαγράφει τα τοπικά δεδομένα.

Περιγραφή: Ο χρήστης πρέπει να μπορεί να διαγράφει τα τοπικά δεδομένα. Αυτό το επιτυγχάνει πατώντας στο κουμπί «Clear databases» του μενού και επιβεβαιώνοντας τη διαγραφή μέσω ενός pop-up διαλόγου.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-13> Ο χρήστης πρέπει να μπορεί να δημιουργεί υποκοινότητες.

Περιγραφή: Ο χρήστης πρέπει να μπορεί να δημιουργεί υποκοινότητες, μέσω κουμπιού της αρχικής οθόνης.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΛΑ-14> Κατά τη δημιουργία υποκοινότητας, ο χρήστης πρέπει να έχει τη δυνατότητα να ορίσει ένα contract που θα παρέχει προσαρμοσμένα tokens για αυτήν.

Περιγραφή: Κατά τη δημιουργία υποκοινότητας, ο χρήστης πρέπει να έχει τη δυνατότητα να ορίσει ένα contract που θα παρέχει προσαρμοσμένα tokens για αυτήν. Τα tokens αυτά θα διαμοιράζονται με τον τρόπο που επιθυμεί η κοινότητα και θα είναι εκείνα τα οποία θα καθορίζουν τους έγκυρους ψηφοφόρους της.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

Η δεύτερη κατηγορία είναι αυτή των Μη Λειτουργικών Απαιτήσεων (ΜΛΑ). Περιλαμβάνει απαιτήσεις αρχιτεκτονικής σημασίας, οι οποίες καθορίζουν κριτήρια ή περιορισμούς του τρόπου λειτουργίας του συστήματος και σχετίζονται με χαρακτηριστικά όπως η αποδοτικότητα, η αξιοπίστία και η ευχρηστία του.

<ΜΛΑ-1> Τα fees για τη χρήση του Ethereum blockchain πρέπει να ελαχιστοποιούνται.

Περιγραφή: Τα fees που πρέπει να καταβάλλονται για τη χρήση του Ethereum blockchain εξαρτώνται άμεσα τόσο από τον όγκο των δεδομένων προς αποθήκευση, όσο και από τους κύκλους επεξεργασίας των smart contracts της εφαρμογής. Ως προς τα δεδομένα, οι προγραμματιστές θα πρέπει να μεριμνούν ώστε ο κύριος όγκος τους να αποθηκεύεται επί του IPFS, ενώ επί του blockchain να αποθηκεύονται μόνο όσα πραγματικά χρειάζονται. Ως προς την απαιτούμενη επεξεργαστική ισχύ, πρέπει να βελτιστοποιείται ο κώδικας των smart contracts, έτσι ώστε οι διάφορες λειτουργίες τους να εκτελούνται με τους λιγότερους δυνατούς επεξεργαστικούς κύκλους.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

<ΜΛΑ-2> Τα contracts της εφαρμογής πρέπει να είναι αναβαθμίσιμα.

Περιγραφή: Τα contracts της εφαρμογής πρέπει μπορούν να αναβαθμιστούν, έτσι ώστε να μπορούν να προστίθενται λειτουργίες και να διορθώνονται σφάλματα. Η αναβαθμισιμότητά τους θα πρέπει να επιτυγχάνεται με μεθόδους που να μην υπονομεύουν τη λειτουργικότητα των προηγούμενων εκδόσεων.

User Priority (?/5): TODO

Technical Priority (?/5): TODO

3.4 Σενάρια χρήσης

Βασικό μέρος της σχεδίασης της πλατφόρμας ήταν η καταγραφή των απαιτήσεων η οποία έγινε στο προηγούμενο κεφάλαιο (3.3) καθώς και η σχεδίαση και ανάπτυξη των σεναρίων χρήσης. Τα σενάρια χρήσης αντιστοιχίζουν πιθανές ενέργειες των χρηστών με αποκρίσεις του συστήματος. Μέσω της αντιστοίχισης αυτής παρουσιάζεται η λειτουργικότητα του συστήματος και περιγράφονται τόσο οι λειτουργικές όσο και οι μη λειτουργικές απαιτήσεις του συστήματος.

Παρατίθενται εδώ τα σενάρια χρήσης που δίνουν τις απαραίτητες πληροφορίες για την κατανόηση της λειτουργίας του συστήματος.

3.4.1 Σενάριο χρήσης 1: Εγγραφή χρήστη

	Εγγράφομαι στο σύστημα
Σύντομη περιγραφή	Στόχος του σεναρίου χρήσης είναι ο επισκέπτης να μπορεί να εγγραφεί στο σύστημα ως χρήστης.
Αναφορά ΛΑ	TODO
Αναφορά ΜΛΑ	TODO
Πυροδότηση δραστηριότητας	Ο επισκέπτης πατάει το κουμπί εγγραφή.
Προϋπόθεση	Ο επισκέπτης πρέπει να έχει ανοίξει την σελίδα της εφαρμογής.

Πίνακας 3.1: Σενάριο χρήσης 1, εγγραφή χρήστη στο σύστημα..

3.4.2 Σενάριο χρήσης 2: Περιήγηση στα θέματα

	Περιηγούμαι στα θέματα
Σύντομη περιγραφή	Στόχος του σεναρίου χρήσης είναι ο επισκέπτης ή ο χρήστης να μπορεί να περιηγηθεί στη λίστα με τα θέματα.
Αναφορά ΛΑ	TODO
Αναφορά ΜΛΑ	TODO
Πυροδότηση δραστηριότητας	Δεν απαιτείται πυροδότηση.
Προϋπόθεση	Ο επισκέπτης ή χρήστης πρέπει να έχει ανοίξει την σελίδα της εφαρμογής.

Πίνακας 3.2: Σενάριο χρήσης 2, περιήγηση στα θέματα..

3.4.3 Σενάριο χρήσης 3: Δημιουργία νέου θέματος

Δημιουργώ νέο θέμα	
Σύντομη περιγραφή	Στόχος του σεναρίου χρήσης είναι ο χρήστης να μπορεί να δημιουργήσει νέο θέμα.
Αναφορά ΛΑ	TODO
Αναφορά ΜΛΑ	TODO
Πυροδότηση δραστηριότητας	Ο χρήστης πατάει το κουμπί δημιουργίας νέου θέματος.
Προϋπόθεση	Ο χρήστης να έχει συνδεθεί στην εφαρμογή και να βρίσκεται στην αρχική σελίδα.

Πίνακας 3.3: Σενάριο χρήσης 3, δημιουργία νέου θέματος..

3.4.4 Σενάριο χρήσης 4: Ανάκτηση θέματος

Ανακτώ ένα θέμα	
Σύντομη περιγραφή	Στόχος του σεναρίου χρήσης είναι ο επισκέπτης ή ο χρήστης να μπορεί να ανακτήσει ένα θέμα.
Αναφορά ΛΑ	TODO
Αναφορά ΜΛΑ	TODO
Πυροδότηση δραστηριότητας	Ο επισκέπτης ή χρήστης πατάει σε ένα από τα θέματα.
Προϋπόθεση	Ο επισκέπτης ή χρήστης πρέπει να έχει ανοίξει την σελίδα της εφαρμογής.

Πίνακας 3.4: Σενάριο χρήσης 4, ανάκτηση θέματος..

3.4.5 Σενάριο χρήσης 5: Δημιουργία νέου μηνύματος

Δημιουργώ νέο μήνυμα	
Σύντομη περιγραφή	Στόχος του σεναρίου χρήσης είναι ο χρήστης να μπορεί να δημιουργήσει νέο μήνυμα.
Αναφορά ΛΑ	TODO
Αναφορά ΜΛΑ	TODO
Πυροδότηση δραστηριότητας	Ο χρήστης πατάει το κουμπί δημιουργίας νέου μηνύματος.
Προϋπόθεση	Ο χρήστης να έχει συνδεθεί στην εφαρμογή και να βρίσκεται στην σελίδα ενός θέματος.

Πίνακας 3.5: Σενάριο χρήσης 5, δημιουργία νέου μηνύματος..

3.4.6 Σενάριο χρήσης 6: Ψήφιση σε ψηφοφορία

Ψηφίζω σε ψηφοφορία	
Σύντομη περιγραφή	Στόχος του σεναρίου χρήσης είναι ο χρήστης να μπορεί να ψηφίσει σε μία ψηφοφορία.
Αναφορά ΛΑ	TODO
Αναφορά ΜΛΑ	TODO
Πυροδότηση δραστηριότητας	Ο χρήστης πατάει το κουμπί ψηφοφορίας.
Προϋπόθεση	Ο χρήστης να έχει συνδεθεί στην εφαρμογή και να βρίσκεται στην σελίδα ενός θέματος το οποίο περιλαμβάνει ψηφοφορία.

Πίνακας 3.6: Σενάριο χρήσης 6, ψήφιση σε ψηφοφορία..

3.4.7 Σενάριο χρήσης 7: Ψήφιση μηνύματος

Ψηφίζω σε μήνυμα	
Σύντομη περιγραφή	Στόχος του σεναρίου χρήσης είναι ο χρήστης να μπορεί να υπερψηφίσει ή καταψηφίσει ένα μήνυμα.
Αναφορά ΛΑ	TODO
Αναφορά ΜΛΑ	TODO
Πυροδότηση δραστηριότητας	Ο επισκέπτης πατάει το κουμπί υπερψήφισης ή καταψήφισης.
Προϋπόθεση	Ο χρήστης να έχει συνδεθεί στην εφαρμογή και να βρίσκεται στην σελίδα ενός θέματος το οποίο περιλαμβάνει τουλάχιστον ένα μήνυμα το οποίο δεν έχει δημιουργήσει ο ίδιος.

Πίνακας 3.7: Σενάριο χρήσης 7, ψήφιση μηνύματος..

3.5 Τεχνολογίες

3.5.1 Ethereum

Ξεκινώντας την σχεδίαση της πλατφόρμας πραγματοποιήσαμε έρευνα ώστε να ανακαλύψουμε τις πιθανές επιλογές για το κομμάτι της διανεμημένης επεξεργασίας (distributed computing). Αναλογιστήκαμε τα προτερήματα και μειονεκτήματα διάφορων επιλογών, συμπεριλαμβανομένων των ...

Επιλέξαμε να προχωρήσουμε με το Ethereum και όχι κάποια άλλη πλατφόρμα επειδή ...

Το Ethereum είναι ... Παρέχει Smart Contracts ακολουθώντας το μοντέλο ... Proof of work είναι ... Ο τρόπος που υπολογίζεται και πληρώνεται η καταναλώμενη επεξεργαστική ισχύς είναι ... Αυτά εισάγουν τους εξής περιορισμούς που πρέπει να ληφθούν υπόψιν κατά την υλοποίηση ...

Προχωρώντας την τεχνολογία του blockchain ένα βήμα παραπέρα, ξεκίνησαν να δημιουργούνται προγραμματιστικές πλατφόρμες για την ανάπτυξη αποκεντρωτικών εφαρμογών (Decentralized Applications ή DApps). Η πρώτη και, μέχρι τώρα, πιο δημοφιλής, ισχυρή και λειτουργική πλατφόρμα είναι το Ethereum.

Στο Ethereum υπάρχουν δύο είδη λογαριασμών: οι Externally Owned Accounts και οι Contracts Accounts. Η διαφορά τους είναι ότι ενώ οι πρώτοι ελέγχονται από τους χρήστες, οι δεύτεροι διαθέτουν ένα αμετάβλητο (immutable) κομμάτι κώδικα το οποίο αποτελεί ένα Smart Contract. Όταν μια συναλλαγή σταλεί σε ένα Smart Contract, ο συσχετιζόμενος κώδικας εκτελείται από την "Ethereum Virtual Machine (EVM)" σε κάθε κόμβο (και εν τέλει όλοι έρχονται σε consensus). Φυσικά, για την εκτέλεση των operations στην EVM (όπως και για τα απλά transactions) χρειάζεται να δοθούν κάποια fees στους miners ως ανταμοιβή για την εργασία τους.

3.5.2 IPFS, OrbitDB

Όπως η επιλογή του Blockchain, που περιγράφηκε στο προηγούμενο κεφάλαιο (insert reference), ομοίως και η επιλογή του λογισμικού που θα χρησιμοποιηθεί για την κατανεμημένη

αποθήκευση δεδομένων ξεκίνησε με μία έρευνα των επιλογών που υπάρχουν. Αναλογιστήκαμε τα προτερήματα και μειονεκτήματα διάφορων επιλογών, συμπεριλαμβανομένων των ...

Επιλέξαμε να προχωρήσουμε με το IPFS και την OrbitDB έναντι άλλων λύσεων επειδή ...

Το IPFS είναι ... Παρέχει ... με τον εξής τρόπο ... Δωρεάν ... Αυτά τα χαρακτηριστικά εισάγουν τους εξής περιορισμούς που πρέπει να ληφθούν υπόψιν κατά την υλοποίηση ...

Η OrbitDB είναι ... και χρησιμοποιεί το IPFS για να καταφέρει τα εξής χαρακτηριστικά ... Περιορισμοί πάλι κλπ ...

Ένα από τα προβλήματα που προκύπτουν με την αποκέντρωση εφαρμογών είναι αυτό της εύρεσης των resources που χρειαζόμαστε. Το πρόβλημα έγκειται στο γεγονός ότι δεν υπάρχει ένας server και άρα μία μοναδική διεύθυνση IP από την οποία μπορούμε να πάρουμε το αντικείμενο που ψάχνουμε, διότι όλα είναι διανεμημένα στο δίκτυο. Επιπλέον η αποθήκευση μεγάλου όγκου πληροφοριών στο Ethereum on-chain έχει απαγορευτικά μεγάλο κόστος οπότε απορρίπτεται.

Το πρόβλημα ανάγεται σε αυτό της επικοινωνίας μεταξύ των κόμβων. Αν καθοριστεί ένα πρότυπο επικοινωνίας μεταξύ των κόμβων τότε τα αντικείμενα μπορούν να βρεθούν ζητώντας τα από τους γειτονικούς κόμβους, οι οποίοι με τη σειρά τους θα τα ζητήσουν από τους δικούς τους γείτονες και ου το κάθε εξής, μέχρι το αντικείμενο να βρεθεί και να σταλεί στον κόμβο που αρχικά το ζήτησε.

Το IPFS είναι ένα νέο πρωτόκολλο μεταφοράς υπερκειμένου που βρίσκεται ακόμα υπό ανάπτυξη. Όπως το γνωστό πρωτόκολλο HTTP, για συγκεντρωτικά συστήματα, έτσι και το IPFS, για αποκεντρωτικά συστήματα, καθορίζει το πρότυπο το οποίο θα χρησιμοποιούν τα τερματικά του δικτύου για να επικοινωνούν μεταξύ τους. Χρησιμοποιεί κρυπτογραφικές τεχνικές, όπως αυτές που είδαμε παραπάνω, καθώς και διάφορες άλλες τεχνολογίες για να:

- συντονίσει τη παράδοση περιεχομένου
- υλοποιήσει στρώμα σύνδεσης μέσω οποιουδήποτε πρωτοκόλλου δικτύου
- ορίσει ένα σύστημα ονομάτων τομέων (DNS)
- υλοποιήσει στρώμα δρομολόγησης
- διαμοιράσει αρχεία με peer-to-peer (P2P) τεχνικές

Έτσι το IPFS ορίζει ένα νέο δίκτυο υπολογιστών που συμπληρώνει τα ήδη υπάρχοντα (www, Tor, .bit) διατηρώντας πλήρως αποκεντρωμένη αρχιτεκτονική και κανένα κεντρικό σημείο αποτυχίας.

Συνοπτικά, δηλαδή, στο IPFS αποθηκεύονται διευθυνσιοδοτημένα βάσει περιεχομένου (content addressable) αρχεία, τα οποία ανακτώνται βάσει του hash των περιεχομένων τους (αντί για την τοποθεσία τους), ενώ ανακαλύπτονται και διαμοιράζονται μέσω του παρεχόμενου P2P network layer. Αξίζει, ωστόσο, να σημειωθεί πως το layer αυτό δεν εγγυάται από μόνο του το hosting των αρχείων και το διαθέσιμο bandwidth για την ανάκτηση τους, πράγμα που σημαίνει ότι θα πρέπει να γίνει παροχή υποδομής από τους χρήστες ικανής να υποστηρίξει έναν ελάχιστο αριθμό κόμβων αποθήκευσης.

Πρόκειται για συμπληρωματικές τεχνολογίες στις παραπάνω που στόχο έχουν την οργάνωση των αποθηκευμένων πληροφοριών σε μορφή βάσεων δεδομένων.

3.6 Προδιαγραφή μεθόδου υλοποίησης και χρονοπρογραμματισμός

3.6.1 Προδιαγραφή κύκλων

Εποπτικά, η διαδικασία της υλοποίησης περιγράφεται ως εξής:

3.6.2 Πρώτη φάση

Στήνεται ένα Ethereum Private Network ως βάση πάνω στην οποία θα δουλέψουμε. Πάνω σε αυτό γράφουμε τα contracts που θα είναι υπεύθυνα για διεκπεραίωση ή μη των posts. Στη συνέχεια αναπτύσσεται ο απαραίτητος κώδικας που υλοποιεί το posting χρησιμοποιώντας τις βιβλιοθήκες που δίνονται από το IPFS για την επικοινωνία μεταξύ των κόμβων του δικτύου και αυτές που δίνονται από τη BigChainDB για την αποθήκευση των πληροφοριών με διανεμημένο τρόπο. Γίνονται δοκιμές για την εξακρίβωση της σωστής λειτουργίας του αποτελέσματος και διορθώνονται τυχόν λάθη στο κώδικα.

3.6.3 Δεύτερη φάση

Υλοποιείται το δικαίωμα ψήφου και posting χωρίς fees. Αυτό γίνεται μέσω δύο contracts που θα δημιουργούν δύο διαφορετικά tokens (voting token, feeless token) και θα τα αποδίδουν στον εκάστοτε χρήστη που πρέπει να πάρει το δικαίωμα. Αναπτύσσεται κώδικας που να υλοποιεί τη διαδικασία ψηφοφορίας. Γίνονται δοκιμές για την εξακρίβωση της σωστής λειτουργίας του αποτελέσματος και διορθώνονται τυχόν λάθη στο κώδικα. Σε αυτή τη φάση η απόδοση των tokens θα γίνει χειροκίνητα για το σκοπό της δοκιμής.

3.6.4 Τρίτη φάση

Υλοποιείται ένα σύστημα απόδοσης εμπιστοσύνης (ΣΑΠ). Αναπτύσσονται τα contracts που είναι απαραίτητα για τη λειτουργία του ΣΑΠ καθώς και για την αυτόματη απόδοση feeless token στους trusted χρήστες. Γίνονται δοκιμές για την εξακρίβωση της σωστής λειτουργίας του αποτελέσματος και διορθώνονται τυχόν λάθη στο κώδικα. Εφόσον η εφαρμογή περάσει το στάδιο των δοκιμών είναι έτοιμη για alpha deployment, είναι δηλαδή έτοιμη για χρήση από το κοινό, υπολείπονται όμως χαρακτηριστικά που είναι ιδιαίτερα θεμιτά αλλά όχι απαραίτητα για τη λειτουργία.

3.6.5 Τέταρτη φάση

Αναπτύσσεται ο κώδικας του (μοναδικού) συγκεντρωτικού τμήματος του συστήματος το οποίο ανήκει στο δεύτερο κομμάτι - του UAS: Έτσι αυτοματοποιείται η διαδικασία απόδοσης των token, που στην προηγούμενη φάση έγινε χειροκίνητα. Γίνονται δοκιμές για την εξακρίβωση της σωστής λειτουργίας του αποτελέσματος και διορθώνονται τυχόν λάθη στο κώδικα. Εφόσον η εφαρμογή περάσει το στάδιο των δοκιμών είναι έτοιμη για ένα beta deployment, ώστε να γίνει πιο ευρύς έλεγχος από μία ομάδα δοκιμών και να παρθεί feedback για την εμπειρία χρήστη.

Για το τελικό deployment θα μπορούσε να τεθεί ως στόχος η κατά το δυνατόν μείωση των τελών για τη λειτουργία της πλατφόρμας, ανεπτυγμένα χαρακτηριστικά επικοινωνίας όπως δόμηση των συζητήσεων σε κατηγορίες, προφίλ χρηστών και άλλα χαρακτηριστικά ευκολίας χρήσης.

3.7 Αρχιτεκτονική

Κεφάλαιο 4

Υλοποίηση εφαρμογής

4.1 Χαρακτηριστικά που υλοποιήθηκαν

4.2 Μεθοδολογία υλοποίησης

4.3 Τεχνολογίες υλοποίησης

4.4 Αρχιτεκτονική υλοποίησης

Το σύστημα υλοποιήθηκε χρησιμοποιώντας το μοντέλο αρχιτεκτονικής των μικροϋπηρεσιών. Το μοντέλο των μικροϋπηρεσιών βασίζεται στην αποδόμηση του συστήματος σε μικρές μονάδες, οι οποίες συνεργάζονται ώστε να προσφέρουν ένα ενιαίο αποτέλεσμα. Η προσέγγιση αυτή έχει πολλά πλεονεκτήματα σε σύγκριση με την ανάπτυξη μονολιθικών εφαρμογών . Ο βασικός λόγος για τον οποίο επιλέχθηκε η αρχιτεκτονική μικροϋπηρεσιών είναι η ευκολία που προσφέρει στη γρήγορη ανάπτυξη καινούριων χαρακτηριστικών, ταυτόχρονα από διαφορετικά μέλη μίας ομάδας, ασύγχρονα και χωρίς την ανάγκη συνεχής επικοινωνίας και συνεννόησης μεταξύ τους. Αυτό συμβαίνει επειδή κάθε μέρος του συστήματος (υπηρεσία) είναι αυτόνομο και η ανάπτυξή του είναι διαχωρισμένη από το υπόλοιπο σύστημα με το οποίο είναι αδύναμα συνδεδεμένο (loosely coupled).

Το σύστημα συντίθεται από διάφορες μικροϋπηρεσίες, κάποιες από τις οποίες αναπτύχθηκαν στα πλαίσια αυτής της εργασίας ενώ άλλες αποτελούν δωρεάν λογισμικό ανοιχτού κώδικα. Οι μικροϋπηρεσίες αυτές συνοψίζονται στον παρακάτω πίνακα (πίνακας 4.1).

Μικροϋπηρεσία	**Σύντομη περιγραφή - Αντικείμενο/Στόχος**
Concordia Application	Υπηρεσία με την οποία αλληλεπιδρούν οι χρήστες.
Concordia Contracts Migrator	Υπηρεσία μεταφόρτωσης των συμβολαίων (contracts) στο blockchain.
Concordia Pinner	Υπηρεσία καρφιτσώματος δεδομένων.
Concordia Contracts Provider	Υπηρεσία διαμοιρασμού των contract artifacts μέσω HTTP.
Ganache	Τοπικό, ιδιωτικό Ethereum blockchain.
Rendezvous Server	Υπηρεσία εύρεσης ομότιμων χρηστών.

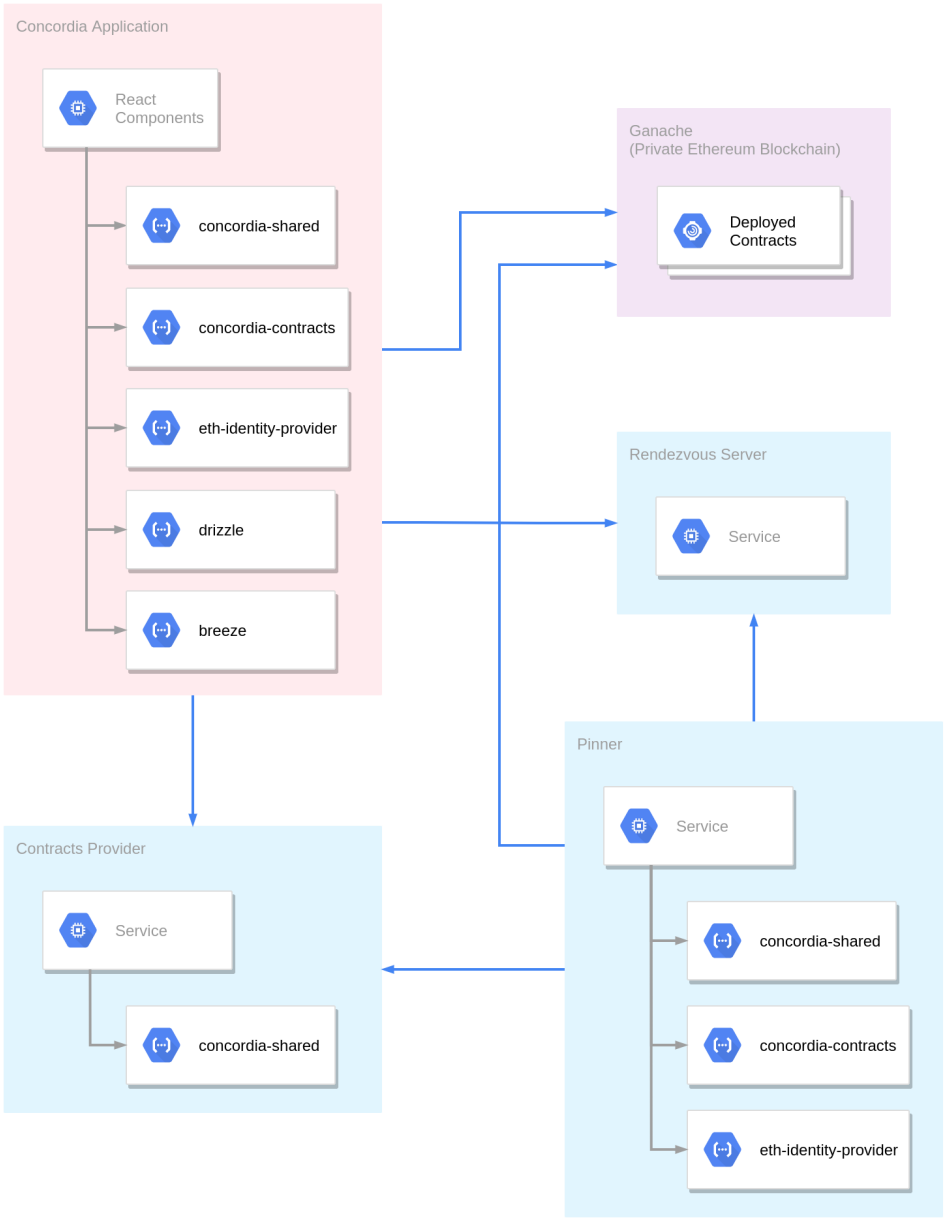
Πίνακας 4.1: Σύντομη περιγραφή υπηρεσιών συστήματος.

Στα πλαίσια της εργασίας αναπτύχθηκαν επίσης διάφορα αρθρώματα, κυρίως με τη μορφή βιβλιοθηκών Javascript. Τα αρθρώματα χρησιμοποιούνται από τις υπηρεσίες για την επίτευξη των επιμέρους εργασιών. Η ανάπτυξη του λογισμικού σε ξεχωριστά αρθρώματα επιτρέπει την εύκολη επαναχρησιμοποίηση του κώδικα καθώς και τον διαχωρισμό των αυτόνομων τμημάτων κώδικα. Τα αρθρώματα συνοψίζονται στον παρακάτω πίνακα (πίνακας 4.2).

Άρθρωμα	**Σύντομη περιγραφή - Αντικείμενο/Στόχος**
Άρθρωμα concordia-shared	Χρήσιμα εργαλεία και σταθερές συστήματος.
Άρθρωμα concordia-contracts	Μεταγλώττιση των contracts και διάθεση των artifacts.
Άρθρωμα eth-identity-provider	Δημιουργία μοναδικού αναγνωριστικού χρήστη για τη βάση OrbitDB.
Άρθρωμα drizzle	Βελτιωμένη προγραμματιστική διεπαφή επικοινωνίας με το blockchain.
Άρθρωμα breeze	Βελτιωμένη προγραμματιστική διεπαφή χρήσης της βάσης OrbitDB.

Πίνακας 4.2: Σύντομη περιγραφή αρθρωμάτων συστήματος.

Τα αρθρώματα και οι υπηρεσίες θα περιγραφούν σε μεγαλύτερη ανάλυση στα επόμενα κεφάλαια. Στο παρακάτω σχήμα (σχήμα 4.1) φαίνεται η συνολική αρχιτεκτονική του συστήματος.



Σχήμα 4.1: Διάγραμμα αρχιτεκτονικής συστήματος

4.4.1 Αρθρώματα

Στο κεφάλαιο αυτό θα περιγραφούν με μεγαλύτερη λεπτομέρεια τα αρθρώματα που αναπτύχθηκαν.

Άρθρωμα concordia-shared

Το άρθρωμα concordia-shared αποτελεί μία βιβλιοθήκη χρήσιμων εργαλείων και σταθερών. Εδώ περιέχεται όλο το λογισμικό το οποίο πρέπει ή είναι επιθυμητό να συμπεριφέρεται με τον ίδιο τρόπο συνολικά στο σύστημα, όπως για παράδειγμα μέθοδοι παραμετροποίησης των υπηρεσιών και μέθοδοι καταγραφής (logging). Το άρθρωμα αυτό χρησιμοποιείται από το άρθρωμα concordia-contracts καθώς και από τις υπηρεσίες Concordia Application, Concordia Pinner και Concordia Contracts Provider.

Το άρθρωμα αυτό γίνεται διαθέσιμο για χρήση με τη μορφή τοπικής βιβλιοθήκης με τη χρήση της βιβλιοθήκης διαχείρισης μοναδικού αποθετηρίου κώδικα (monorepo) lerna.

Άρθρωμα concordia-contracts

Το άρθρωμα αυτό επιτελεί δύο ενέργειες. Αρχικά, είναι το άρθρωμα στο οποίο αναπτύσσονται τα contracts που χρησιμοποιούνται από την εφαρμογή. Το άρθρωμα αυτό αναλαμβάνει τη μεταγλώττιση των contracts από κώδικα γλώσσας Solidity, στην κατάλληλη τελική μορφή JSON. Παρέχονται επίσης σενάρια ενεργειών (scripts) ώστε τα contracts να μεταφορτωθούν σε blockchain καθώς και στην υπηρεσία Concordia Contracts Provider. Αποτελεί επίσης βιβλιοθήκη η οποία μετά τη μεταγλώττιση και μεταφόρτωση των contracts σε blockchain παρέχει τα contract artifacts. Χρησιμοποιείται από τις υπηρεσίες Concordia Application και Concordia Pinner.

Το άρθρωμα αυτό γίνεται διαθέσιμο για χρήση με τη μορφή τοπικής βιβλιοθήκης με τη χρήση της βιβλιοθήκης διαχείρισης monorepo lerna.

Άρθρωμα eth-identity-provider

Η λειτουργία της βάση OrbitDB απαιτεί τη δημιουργία ενός μοναδικού αναγνωριστικού χρήστη (identity). Για την εύκολη εξαγωγή ενός αναγνωριστικού χρήστη το οποίο να είναι μεν μοναδικό αλλά να είναι δυνατός ο επανυπολογισμός, χρησιμοποιήθηκε ο συνδυασμός της διεύθυνσης του χρήστη στο δίκτυο Ethereum με τη διεύθυνση του βασικού contract που χρησιμοποιεί η εφαρμογή. Ο υπολογισμός του συνδυασμού αυτού υλοποιείται από αυτό το άρθρωμα.

Το άρθρωμα αυτό γίνεται διαθέσιμο για χρήση με τη μορφή βιβλιοθήκης μέσω του αποθετηρίου λογισμικού npm.

Άρθρωμα drizzle

Το άρθρωμα drizzle που χρησιμοποιείται στην υπηρεσία Concordia Application είναι μία τροποποιημένη έκδοση της Javascript βιβλιοθήκης Drizzle που προσφέρεται από τη σουίτα εργαλείων Truffle. Η τροποποιημένη βιβλιοθήκη αναπτύχθηκε στα πλαίσια της διπλωματικής με στόχο τη διευκόλυνση της χρήσης του Drizle και την επιδιόρθωση προβληματικών σημείων της πρωτότυπης βιβλιοθήκης.

Το άρθρωμα drizzle υλοποιεί τις προγραμματιστικές διεπαφές μέσω των οποίων πραγματοποιείται η επικοινωνία της εφαρμογής με το blockchain. Για την επίτευξη της επικοινωνίας αυτής, η βιβλιοθήκη χρησιμοποιεί τη συλλογή βιβλιοθηκών web3.js η οποία αποτελεί τον πιο διαδεδομένο τρόπο διεπαφής με το blockchain σε αποκεντρωτικές εφαρμογές.

Το άρθρωμα αυτό γίνεται διαθέσιμο για χρήση με τη μορφή βιβλιοθήκης μέσω του αποθετηρίου λογισμικού npm.

Άρθρωμα breeze

Το άρθρωμα αυτό αποτελεί μία βιβλιοθήκη περίβλημα (wrapper) της βιβλιοθήκης OrbitDB. Η OrbitDB είναι μία βιβλιοθήκη η οποία προσφέρει τις απαραίτητες προγραμματιστικές διεπαφές για τη χρήση της βάσης δεδομένων με το ίδιο όνομα. Μέσα από τη χρήση των βιβλιοθηκών που προσφέρονται από το IPFS για την αποθήκευση δεδομένων, η OrbitDB καταφέρνει να υλοποιήσει μία αποκεντρωμένη βάση δεδομένων.

Το άρθρωμα breeze κάνει χρήση της βιβλιοθήκης OrbitDB, προσφέρει ωστόσο συγκεκριμένες προγραμματιστικές διεπαφές που διευκολύνουν τόσο την παραμετροποίηση της βάσης όσο και τη χρήση της, ενώ όπως και στο άρθρωμα drizzle το άρθρωμα breeze αναλαμβάνει να διορθώσει ορισμένα προβλήματα της πρωτότυπης βιβλιοθήκης.

Το άρθρωμα αυτό γίνεται διαθέσιμο για χρήση με τη μορφή βιβλιοθήκης μέσω του αποθετηρίου λογισμικού npm.

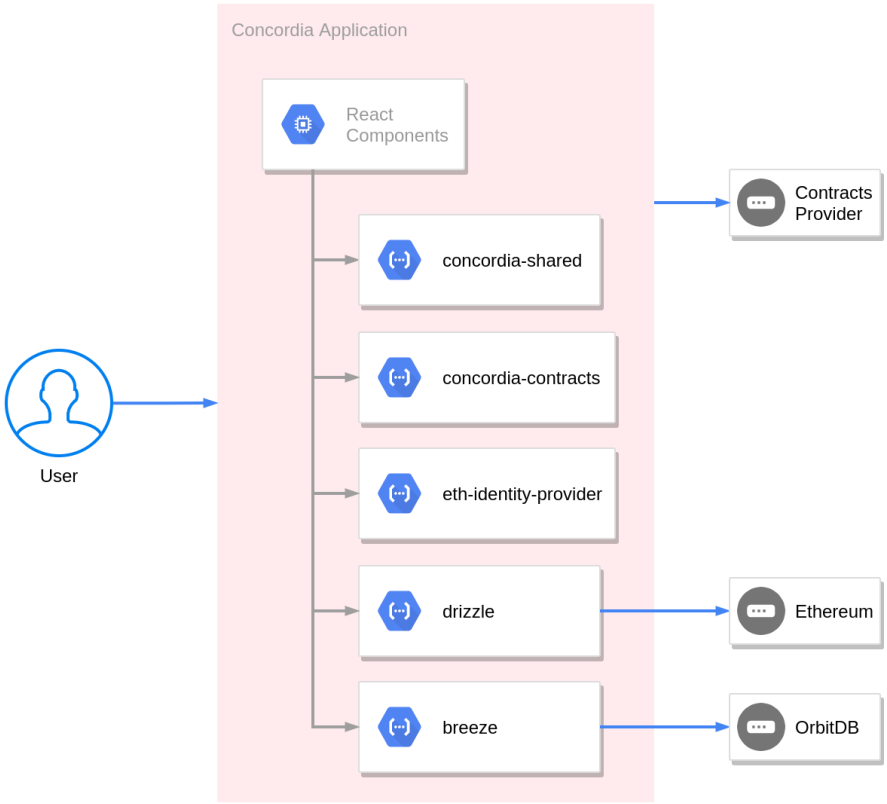
4.4.2 Concordia Application

Περιγραφή - Στόχοι υπηρεσίας
Η εφαρμογή Concordia (Concordia Application) εκθέτει τις γραφικές διεπαφές μέσω των οποίων αλληλεπιδρούν οι χρήστες με το σύστημα. Αποτελεί τον δίαυλο επικοινωνίας του τελικού χρήστη με το blockchain και με τη βάση OrbitDB. Η αρχιτεκτονική της υπηρεσίας φαίνεται στο σχήμα 4.2. Μέσω της εφαρμογής Concordia οι χρήστες μπορούν να:

- περιηγηθούν και διαβάσουν το περιεχόμενο της πλατφόρμας
- δημιουργήσουν λογαριασμό χρήστη
- δημοσιεύσουν και τροποποιήσουν προσωπικές τους πληροφορίες όπως η τοποθεσία και η εικόνα προφίλ
- δημιουργήσουν θέματα (topics)
- δημιουργήσουν ψηφοφορίες (polls), καθώς και να ψηφίσουν σε αυτές
- δημιουργήσουν και τροποποιήσουν μηνύματα (posts)
- υπερψηφίσουν (up-vote) ή καταψηφίσουν (down-vote) μηνύματα άλλων χρηστών

Η υπηρεσία αποτελείται από κώδικα γραμμένο σε Javascript ο οποίος γίνεται διαθέσιμος στους τελικούς χρήστες με τη μορφή εφαρμογής διαδικτύου (web application) μέσω ενός διακομιστή (server). Παρόλο που η υπηρεσία προσφέρει τη γραφική διεπαφή χρήστη μόνο στην αγγλική γλώσσα, έχει παραμετροποιηθεί ώστε να είναι δυνατή η εύκολη μεταγλώττιση της χωρίς την ανάγκη πραγματοποίησης μεγάλων αλλαγών στον κώδικα.

Χρησιμοποιείται η βιβλιοθήκη React για την οργάνωση και ανάπτυξη των συνθετικών τμημάτων (components) του γραφικού περιβάλλοντος. Για το γραφικό περιβάλλον γίνεται χρήση του framework της Semantic UI. Χρησιμοποιείται η βιβλιοθήκη Redux για τη διαχείριση κατάστασης της εφαρμογής (state management), καθώς και η βιβλιοθήκη Redux-Saga για τη διαχείριση ασύγχρονων παράπλευρων ενεργειών (side-effects) σε ένα σύστημα βασισμένο σε συμβάντα (event-based). Άλλες βιβλιοθήκες χρησιμοποιούνται για διάφορα μέρη της υπηρεσίας, ενώ χρησιμοποιούνται επίσης τα αρθρώματα που περιγράφηκαν προηγουμένως για την επίτευξη διαφορετικών στόχων. Ο πλήρης κατάλογος των βιβλιοθηκών και αρθρωμάτων μπορεί να βρεθεί στον κώδικα της υπηρεσίας στο παράρτημα.



Σχήμα 4.2: Αρχιτεκτονική υπηρεσίας Concordia Application

Για τη λειτουργία της υπηρεσία Concordia Application είναι απαραίτητα τα αντικείμενα (artifacts) που προκύπτουν από τη μεταγλώττιση των contracts και τη μεταφόρτωση/δημοσίευσή τους στο blockchain. Για την εισαγωγή των artifacts στην υπηρεσία έχουν αναπτυχθεί δύο μέθοδοι.

Η πρώτη μέθοδος είναι η μεταγλώττιση και μεταφόρτωση των contracts πριν την παραγωγή του πακέτου λογισμικού της υπηρεσίας για τελική χρήση (production build). Με αυτό τον τρόπο η υπηρεσία θα έχει πρόσβαση στα artifacts μέσω της βιβλιοθήκης που παράγεται από το άρθρωμα concordia-contracts. Αυτή η μέθοδος έχει το μειονέκτημα ότι το τελικό πακέτο λογισμικού (production build) "δένεται" με όποια συγκεκριμένη έκδοση των contracts είναι διαθέσιμη κατά τη δημιουργία του πακέτου. Αυτό σημαίνει ότι σε ενδεχόμενη ενημέρωση των contracts πρέπει αναγκαστικά να δημιουργηθεί και νέα έκδοση του πακέτου λογισμικού της υπηρεσίας Concordia Application.

Για την αποφυγή του παραπάνω προβλήματος αναπτύχθηκε η δεύτερη μέθοδος προσκόμισης των contract artifacts, η οποία είναι η λήψη τους (download) από μία άλλη τοποθεσία στο διαδίκτυο. Σε αυτή τη μέθοδο, η εφαρμογή κατά την εκκίνησή της πραγματοποιεί ένα HTTP αίτημα (HTTP request) σε διεύθυνση η οποία δίνεται ως μεταβλητή περιβάλλοντος (environment variable). Η απάντηση του αιτήματος αναμένεται να περιέχει τα artifacts ώστε η εφαρμογή να τα χρησιμοποιήσει.

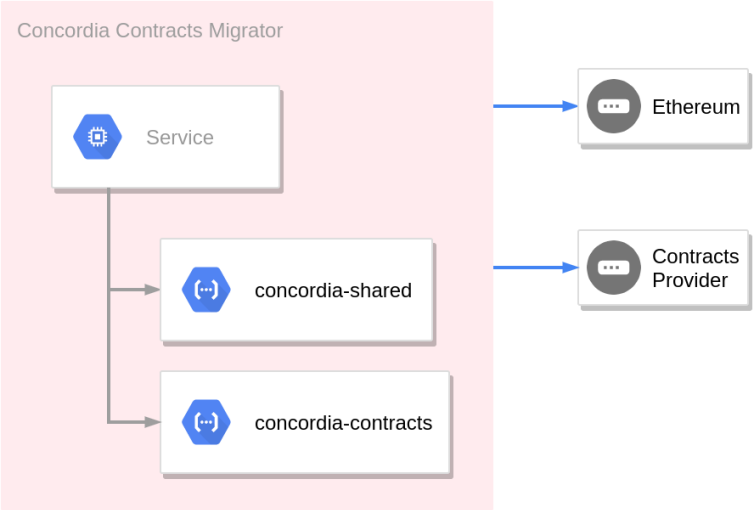
Διανομή

Η υπηρεσία Concordia Application πακετάρεται μαζί με τον διακομιστή nginx και γίνεται διαθέσιμη για χρήση ως εικόνα docker (docker image) μέσω του αποθετηρίου εικόνων dockerhub. Κατά την εκτέλεση της εικόνας οι χρήστες μπορούν μέσω μεταβλητών περιβάλλοντος να ορίσουν παραμέτρους της εκτέλεσης όπως η διεύθυνση του εξυπηρετητή (host location) της εφαρμογής και οι τοποθεσίες των υπηρεσιών Rendezvous Server και Contracts Provider.

4.4.3 Concordia Contracts Migrator

Περιγραφή - Στόχοι υπηρεσίας

Η υπηρεσία αυτή αποτελείται από ένα εκτελέσιμο πρόγραμμα γραμμής εντολών βασισμένο στο άρθρωμα concordia-contracts που αναλύθηκε σε προηγούμενο κεφάλαιο (κεφάλαιο 4.4.1). Το πρόγραμμα, κατά την εκτέλεσή του, μεταγλωττίζει τα contracts και έπειτα τα μεταφορτώνει στο blockchain το οποίο είναι ορισμένο με χρήση μεταβλητών περιβάλλοντος. Τέλος, αν οι κατάλληλες μεταβλητές περιβάλλοντος είναι ορισμένες, το πρόγραμμα μεταφορτώνει τα τελικά artifacts σε αποθετήριο Concordia Contracts Provider. Η αρχιτεκτονική της υπηρεσίας φαίνεται στο παρακάτω σχήμα (σχήμα 4.3).



Σχήμα 4.3: Αρχιτεκτονική υπηρεσίας Concordia Contracts Migrator

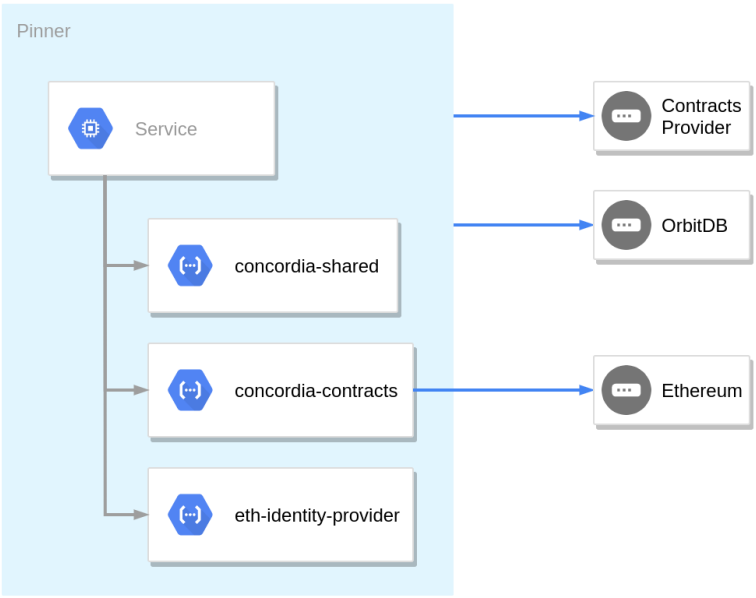
Διανομή

Η υπηρεσία αυτή γίνεται διαθέσιμη για χρήση ως docker image μέσω του αποθετηρίου εικόνων dockerhub. Οι χρήστες μπορούν χρησιμοποιώντας μεταβλητές περιβάλλοντος να αλλάξουν τη διεύθυνση του blockchain και την τοποθεσία της υπηρεσίας Contracts Provider στην οποία το πρόγραμμα θα μεταφορτώσει τα contracts και τα artifacts.

4.4.4 Concordia Pinner

Περιγραφή - Στόχοι υπηρεσίας

Η υπηρεσία καρφιτσώματος περιεχομένου (Concordia Pinner) αποτελεί μία εφαρμογή τερματικού (temrinal application/cmd application) η οποία στοχεύει στο καρφίτσωμα (pinning) του περιεχομένου που αποθηκεύεται στο IPFS μέσω της βάσης OrbitDB. Η υπηρεσία είναι γραμμένη στη γλώσσα προγραμματισμού Javascript. Η αρχιτεκτονική της υπηρεσίας φαίνεται το σχήμα 4.4.



Σχήμα 4.4: Αρχιτεκτονική υπηρεσίας Concordia Pinner

Η υπηρεσία αυτή υλοποιήθηκε για να εγγυηθεί η διαθεσιμότητα του περιεχομένου του συστήματος που αποθηκεύεται στο IPFS (τίτλοι θεμάτων, περιεχόμενο μηνυμάτων και άλλα). Λόγω του τρόπου λειτουργίας του IPFS, το περιεχόμενο που αναρτούν οι χρήστες πρέπει να καρφιτσώνεται από άλλους χρήστες ή αυτόνομες εφαρμογές, όπως η υπηρεσία Concordia Pinner, ώστε να είναι διαθέσιμο. Αν το περιεχόμενο δεν καρφιτσωθεί, τότε θα είναι διαθέσιμο στους υπόλοιπους χρήστες μόνο από τον/τη δημιουργό, έτσι αν αυτός/αυτή δεν είναι ενεργός/ενεργή στο δίκτυο, το περιεχόμενο θα είναι αδύνατο να βρεθεί.

Η υπηρεσία συνδέεται στο blockchain από όπου παρακολουθεί την κατάσταση του συστήματος και "ακούει" για νέους χρήστες, θέματα και μηνύματα. Η υπηρεσία συνδέεται επίσης στο IPFS, έτσι όταν δημιουργηθεί νέο περιεχόμενο στο σύστημα το καρφιτσώνει αυτόματα. Με αυτό τον τρόπο, διατηρώντας την υπηρεσία πάντα διαθέσιμη, για παράδειγμα εκτελώντας τη σε περιβάλλον διακομιστή (server), διαβεβαιώνεται η διαθεσιμότητα του περιεχομένου.

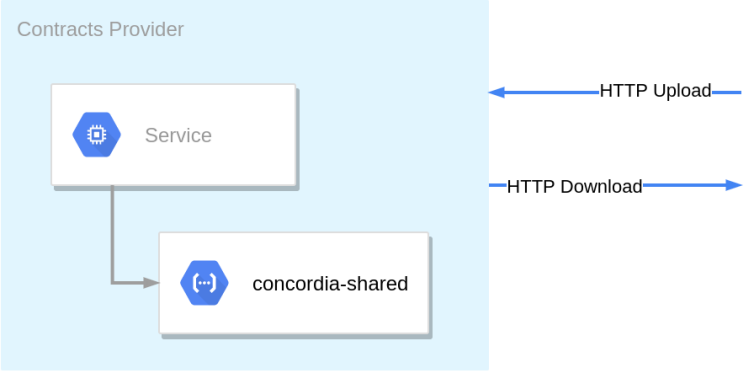
Διανομή

Η υπηρεσία αυτή γίνεται διαθέσιμη για χρήση ως docker image μέσω του αποθετηρίου εικόνων dockerhub. Κατά την εκτέλεση της εικόνας οι χρήστες μπορούν μέσω μεταβλητών περιβάλλοντος να ορίσουν παραμέτρους της υπηρεσίας όπως τη διεύθυνση του εξυπηρετητή (host location), τη διεύθυνση του blockchain, τις διαδρομές αποθήκευσης των δεδομένων στο σύστημα και τις τοποθεσίες των υπηρεσιών Rendezvous Server και Contracts Provider.

4.4.5 Concordia Contracts Provider

Περιγραφή - Στόχοι υπηρεσίας

Η υπηρεσία Contracts Provider αποτελεί μία βοηθητική υπηρεσία η οποία υλοποιεί ένα απλό αποθετήριο για τα contract artifacts. Είναι γραμμένη σε Javascript και διαθέτει δύο HTTP endpoints, ένα για τη μεταφόρτωση (upload) των artifacts προς την υπηρεσία και ένα για τη λήψη (download) από την υπηρεσία. Η υπηρεσία υποστηρίζει επίσης την επισύναψη ετικετών στα artifacts, όπως η έκδοση (version) ή το κλαδί ανάπτυξης (branch, για παράδειγμα master/develop). Η αρχιτεκτονική της υπηρεσίας φαίνεται το σχήμα 4.5.



Σχήμα 4.5: Αρχιτεκτονική υπηρεσίας Concordia Contracts Provider

Η υπηρεσία χρησιμοποιείται σε μία προσπάθεια αποσύνδεσης της βασικής εφαρμογής που υλοποιεί η υπηρεσία Concordia Application από μία συγκεκριμένη έκδοση των contracts. Οι λόγοι που αυτό είναι επιθυμητό αναπτύχθηκαν στην περιγραφή της υπηρεσίας Concordia Application (κεφάλαιο 4.4.2). Ωστόσο, η υπηρεσία Contracts Provider αποτελεί σημείο κεντροποίησης του συστήματος, για το λόγο αυτό θεωρείται προσωρινή λύση η οποία θα μπορούσε να αντικατασταθεί από αποκεντρωτικές λύσεις όπως η μεταφόρτωση των artifacts στο IPFS και ο διαμοιρασμός τους από εκεί.

Διανομή

Η υπηρεσία αυτή γίνεται διαθέσιμη για χρήση ως docker image μέσω του αποθετηρίου εικόνων dockerhub. Οι χρήστες μπορούν χρησιμοποιώντας μεταβλητές περιβάλλοντος να αλλάξουν παραμέτρους της εκτέλεσης όπως η διαδρομή αποθήκευσης των μεταφορτωμένων contract artifacts.

4.4.6 Ganache

Περιγραφή - Στόχοι υπηρεσίας

Η υπηρεσία Ganache αποτελεί μία εφαρμογή τερματικού η οποία είναι μέρος της δωρεάν σουίτας ανοιχτού λογισμικού Truffle. Η εφαρμογή δημιουργεί ένα τοπικό, ιδιωτικό blockchain το οποίο ακολουθεί το πρότυπο του Ethereum. Επίσης, η εφαρμογή δρα ως minner στο δίκτυο, διεκπεραιώνοντας όλες τις συναλλαγές.

Διανομή

Για τη χρήση της υπηρεσίας αυτής αναπτύχθηκε μία νέα εικόνα docker που βασίζεται στην επίσημη εικόνα που διατίθεται από τη σουίτα και προσθέτει μερικές χρήσιμες λειτουργικότητες όπως η δυνατότητα αποκάλυψης των κλειδιών που δημιουργούνται κατά την εκτέλεση. Η υπηρεσία γίνεται διαθέσιμη για χρήση ως docker image μέσω του αποθετηρίου εικόνων dockerhub. Η εικόνα παρέχει τη δυνατότητα τροποποίησης των παραμέτρων εκτέλεσης με χρήση μεταβλητών περιβάλλοντος. Με αυτό τον τρόπο οι χρήστες μπορούν να αλλάξουν τον αριθμό των λογαριασμών που θα δημιουργηθούν, το ποσό του Ether που θα λάβει κάθε λογαριασμός καθώς και άλλες μεταβλητές.

4.4.7 Rendezvous Server

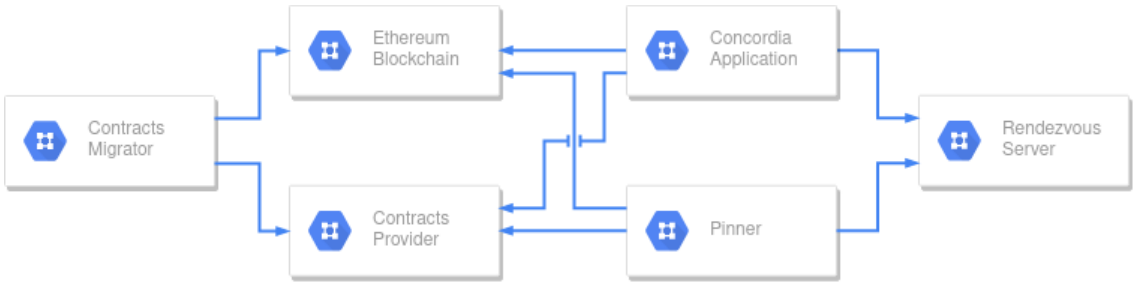
Περιγραφή - Στόχοι υπηρεσίας
Η υπηρεσία Rendezvous Server αποτελεί δωρεάν λογισμικό ανοιχτού κώδικα το οποίο χρησιμοποιήθηκε (αλλά δεν αναπτύχθηκε) στα πλαίσια της διπλωματικής και υλοποιεί το πρωτόκολλο rendezvous για την εύρεση ομότιμων χρηστών (peers). Η υπηρεσία είναι απαραίτητη για τη λειτουργία του IPFS, ώστε οι ομότιμοι χρήστες (peers) να μπορούν να ανακαλύψουν τις διευθύνσεις των υπόλοιπων χρηστών του δικτύου.

Διανομή
Η υπηρεσία αυτή είναι διαθέσιμη για χρήση από τους δημιουργούς της τόσο ως εφαρμογή μέσω του αποθετηρίου λογισμικού npm αλλά και ως docker image μέσω του αποθετηρίου εικόνων dockerhub.

4.4.8 Διασύνδεση υπηρεσιών

Στο μοντέλο των μικροϋπηρεσιών, βασικό χαρακτηριστικό είναι η επικοινωνία των ξεχωριστών υπηρεσιών και η ανταλλαγή μηνυμάτων για την επίτευξη των λειτουργικοτήτων του συστήματος. Σε αυτό το κεφάλαιο θα αναλυθεί ο τρόπος με τον οποίο οι μικροϋπηρεσίες επικοινωνούν μεταξύ τους καθώς και η φύση και το περιεχόμενο των μηνυμάτων που ανταλλάσουν.

Στο παρακάτω σχήμα (σχήμα 4.6) φαίνεται ο γράφος που οπτικοποιεί τα κανάλια επικοινωνίας μεταξύ των μικροϋπηρεσιών, καθώς και τα κανάλια επικοινωνίας των μικροϋπηρεσιών με το blockchain.



Σχήμα 4.6: Γράφος οπτικοποίησης των καναλιών επικοινωνίας των μικροϋπηρεσιών

Εδώ αναλύεται η επικοινωνία κάθε μικροϋπηρεσίας:

- **Contracts Migrator**: η υπηρεσία εκτελεί αίτημα HTTP κατά την μεταφόρτωση των contracts στο Ethereum blockchain, επίσης εκτελεί αίτημα HTTP για την μεταφόρτωση των contract artifacts στην υπηρεσία Contracts Provider
- **Concordia Application**: η υπηρεσία εκτελεί αίτημα HTTP για την λήψη των contract artifacts από την υπηρεσία Contracts Provider, εκτελεί αιτήματα HTTP για την διενέργεια συναλλαγών στο Ethereum blockchain και τέλος δημιουργεί κανάλι UDP επικοινωνίας με την υπηρεσία Rendezvous Server για την ανακάλυψη ομότιμων χρηστών (peers) στο δίκτυο IPFS
- **Pinner**: η υπηρεσία εκτελεί αίτημα HTTP για την λήψη των contract artifacts από την υπηρεσία Contracts Provider, εκτελεί αιτήματα HTTP για την ανανέωση και παρατήρηση της κατάστασης του contract στο Ethereum blockchain και τέλος δημιουργεί κανάλι UDP

επικοινωνίας με την υπηρεσία Rendezvous Server για την ανακάλυψη peers στο δίκτυο IPFS

- **Rendezvous Server**: η υπηρεσία διατηρεί ανοιχτά κανάλια UDP επικοινωνίας με τους ομότιμους χρήστες μέσω των οποίων ενημερώνει την λίστα των διαθέσιμων, ενεργών χρηστών
- **Contracts Provider**: η υπηρεσία δεν υποκινεί καμία επικοινωνία παρά μόνο απαντά σε αιτήματα επικοινωνία από άλλες υπηρεσίες

4.4.9 Ροή πληροφορίας

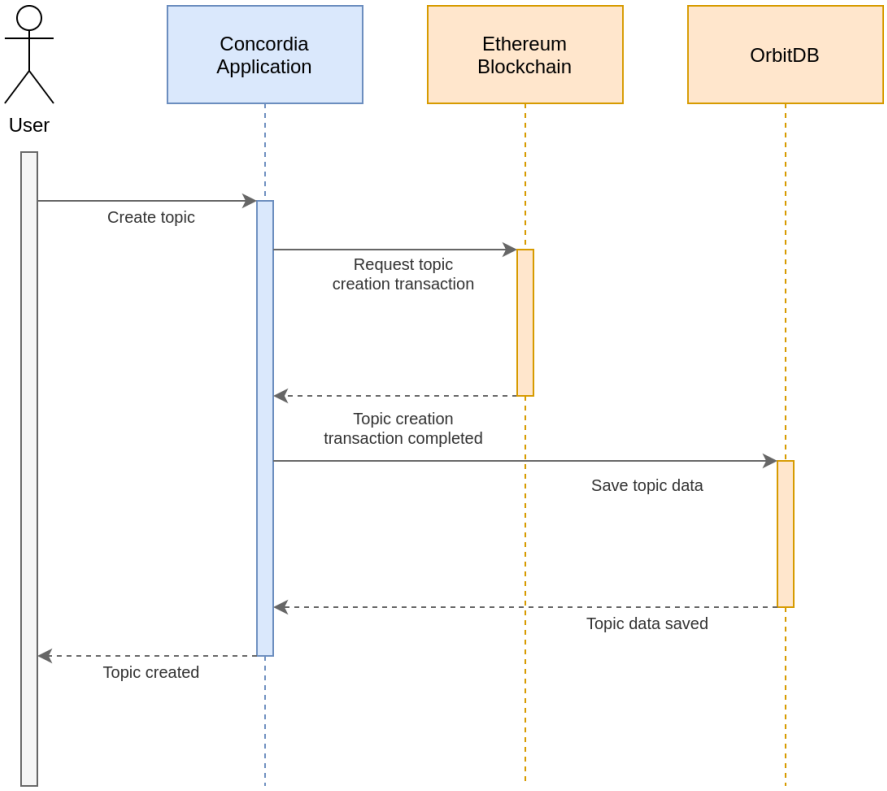
Στο κεφάλαιο αυτό θα αναλυθεί η ροή της πληροφορίας στο σύστημα. Λόγω των πολλαπλών υπηρεσιών, της κατάτμησης την πληροφορίας και των διαφορετικών σημείων αποθήκευσης της, η ροή της πληροφορίας στο σύστημα ακολουθεί ένα σχετικά περίπλοκο μονοπάτι (σε σχέση με κλασσικές, μονολιθικές, κεντροποιημένες εφαρμογές).

Αρχικά θα γίνει αναφορά στη διαδικασία αποθήκευσης των νέων πληροφοριών. Η μοναδική πηγή παραγωγής δεδομένων στο σύστημα είναι οι χρήστες και κατ' επέκταση η υπηρεσία Concordia Application, εφόσον είναι η μοναδική υπηρεσία με την οποία αυτοί αλληλεπιδρούν. Τα δεδομένα που δημιουργούν οι χρήστες (πληροφορίες χρηστών, τίτλοι θεμάτων και περιεχόμενο μηνυμάτων) κατατμήζονται πριν αποθηκευτούν. Η πληροφορία που εισάγεται στο σύστημα κατατμήζεται σε δύο μέρη. Στο blockchain αποθηκεύεται ένας δείκτης προς τα δεδομένα, ενώ τα πραγματικά δεδομένα αποθηκεύονται στη βάση OrbitDB. Ο δείκτης εκτός από την άμεση χρησιμότητα στην εύρεση των δεδομένων, παρέχει και την έμμεση λειτουργικότητα της δημιουργίας απαραίτητων μεταδομένων όπως ο αριθμός των θεμάτων στο σύστημα ή των μηνυμάτων σε ένα θέμα.

Από την πλευρά της εύρεση των πληροφοριών στο σύστημα, η ροή είναι ως εξής. Αρχικά, είναι απαραίτητη η αναζήτηση στο blockchain για την εύρεση του δείκτη προς τα δεδομένα. Έπειτα, τα δεδομένα μπορούν να ανακτηθούν μέσω του IPFS από τον εκάστοτε χρήστη ή από κάποιον Pinner.

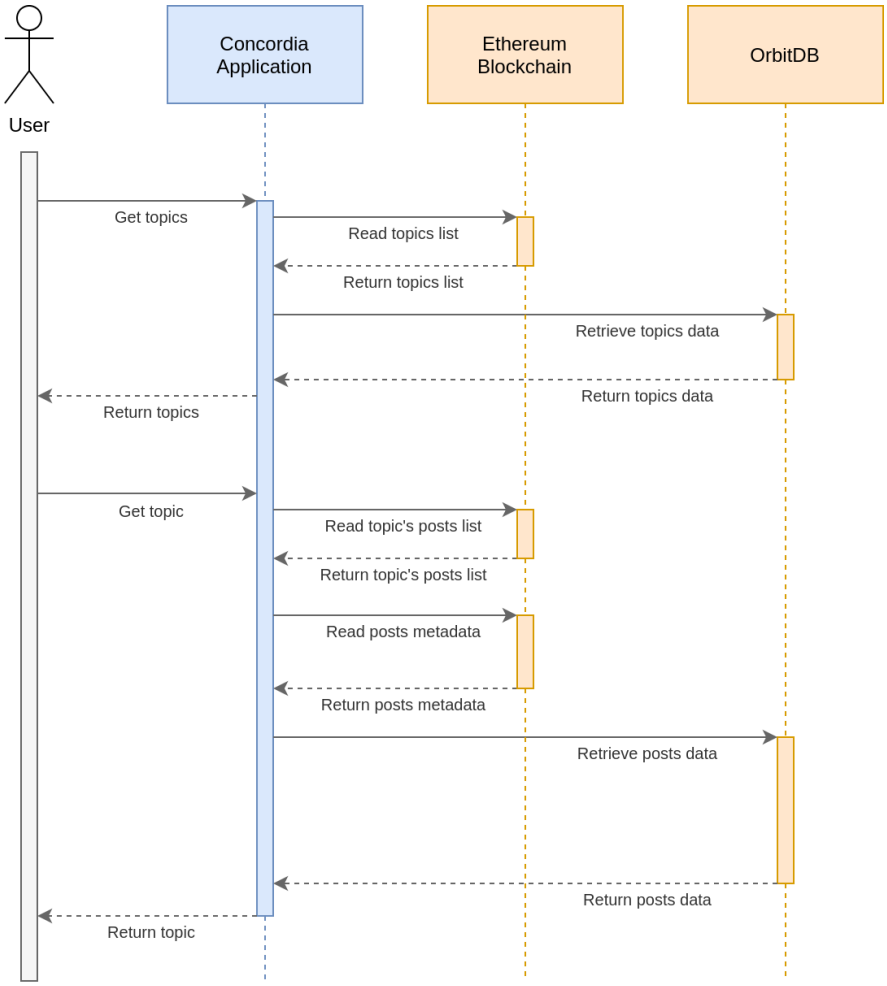
Τέλος, παρακάτω δίνεται ένα παράδειγμα εισαγωγής πληροφορίας στο σύστημα και έπειτα ανάκτησης της ίδιας πληροφορίας.

Έστω, χρήστης που δημιουργεί νέο θέμα. Τα δεδομένα που παράγονται είναι ο τίτλος του θέματος και το περιεχόμενο του πρώτου μηνύματος. Μεταδεδομένα της δημιουργίας είναι η διεύθυνση του/της δημιουργού του θέματος. Για την αποθήκευση του θέματος στο σύστημα δημιουργείται πρώτα συναλλαγή στο blockchain ώστε να δημιουργηθεί μία νέα εγγραφή στον πίνακα των θεμάτων. Η εγγραφή αυτή δεν περιέχει τίποτα παρά μόνο τη διεύθυνση του/της δημιουργού χρήστη. Αν η συναλλαγή είναι επιτυχής, θα επιστραφεί ο αύξων αριθμός του νέου θέματος. Έπειτα, στην προσωπική βάση OrbitDB του/της χρήστη και στον πίνακα των θεμάτων θα προστεθεί εγγραφή με αναγνωριστικό τον αύξων αριθμό του θέματος όπου θα αποθηκευτούν τα δεδομένα του τίτλου και πρώτου μηνύματος. Στο σχήμα 4.7 παρουσιάζεται γραφικά η διαδικασία.



Σχήμα 4.7: Διάγραμμα ακολουθίας δημιουργίας θέματος

Έστω, χρήστης που επιθυμεί να διαβάσει το προηγούμενο μήνυμα. Αρχικά, πρέπει να διαβαστεί ο πίνακας θεμάτων από το blockchain. Η πληροφορία αυτή εμπλουτίζεται από τα δεδομένα του κάθε θέματος, τα οποία ανακτώνται από τις προσωπικές βάσεις Orbit κάθε χρήστη. Έπειτα, εφόσον το θέμα βρεθεί και ο αύξων αριθμός του είναι γνωστός, πρέπει να διαβαστούν από το blockchain τα μεταδομένα των μηνυμάτων του θέματος και συγκεκριμένα η διευθύνσεις των δημιουργών τους. Τέλος, μέσω του IPFS πρέπει να γίνει αντιγραφή των προσωπικών βάσεων των δημιουργών του κάθε μηνύματος και να αναζητηθούν σε αυτές τα εκάστοτε μηνύματα. Στο σχήμα 4.8 φαίνεται το διάγραμμα ροής της πληροφορίας κατά την ανάκτηση πληροφοριών από το σύστημα.



Σχήμα 4.8: Διάγραμμα ακολουθίας εύρεσης και ανάκτησης θέματος

Κεφάλαιο 5

Συμπεράσματα

5.1 Προβλήματα ανάπτυξης

5.2 Διαφορές σχεδιασμού-υλοποίησης

5.3 Ανοιχτά θέματα

5.4 Επίλογος

Βιβλιογραφία

[1] Wikipedia. *Merkle tree*. URL: https://en.wikipedia.org/wiki/Merkle_tree.

[2] Belavadi Prahalad. *Merkle proofs Explained.* 7 Ιαν. 2018. URL: https://medium.com/crypto-0-nite/merkle-proofs-explained-6dd429623dc5.

[3] R. Schollmeier. «A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications». Στο: *Proceedings First International Conference on Peer-to-Peer Computing*. 2001, σσ. 101–102. DOI: 10.1109/P2P.2001.990434.

[4] Satoshi Nakamoto. «Bitcoin: A Peer-to-Peer Electronic Cash System». Στο: *Cryptography Mailing list at https://metzdowd.com* (31 Οκτ. 2008).

[5] Wikipedia. *Blockchain*. URL: https://en.wikipedia.org/wiki/Blockchain.

[6] Vitalik Buterin. *Ethereum Whitepaper*. 2013. URL: https://ethereum.org/en/whitepaper (επίσκεψη 28/06/2021).

[7] Alexander Savelyev. «Contract law 2.0: 'Smart' contracts as the beginning of the end of classic contract law». Στο: *Information & Communications Technology Law* 26 (Απρ. 2017), σσ. 1–19. DOI: 10.1080/13600834.2017.1301036.

[8] Gavin Wood Andreas M Antonopoulos. *Mastering Ethereum: Building Smart Contracts and DApps*. O'Reilly Media, 2018. ISBN: 1491971940.

[9] *IPFS*. URL: https://ipfs.io/.

[10] *IPFS documentation*. URL: https://docs.ipfs.io/.

[11] ProtoSchool. *Merkle DAGs: Structuring Data for the Distributed Web*. URL: https://proto.school/merkle-dags/.

[12] *OrbitDB*. URL: https://orbitdb.org.

[13] *Getting Started with OrbitDB*. URL: https://github.com/orbitdb/orbit-db/blob/main/GUIDE.md.