ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάπτυξη αποκεντρωμένης εφαρμογής με χρήση Blockchain για τον διαμοιρασμό αρχείων πολυμέσων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βασίλειος Παπαευθυμίου

Επιβλέπουσα: Θεοδώρα Βραρβαρίγου
Καθηγήτρια Ε.Μ.Π.

Αθήνα, Ιούνιος 2018

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάπτυξη αποκεντρωμένης εφαρμογής με χρήση Blockchain για τον διαμοιρασμό αρχείων πολυμέσων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βασίλειος Παπαευθυμίου

Επιβλέπουσα: Θεοδώρα Βαρβαρίγου
Καθηγήτρια Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 26η Ιουνίου 2018.

.....	.....	.....
Θεοδώρα Βαρβαρίγου Καθηγήτρια Ε.Μ.Π.	Δημήτριος Ασκούνης Καθηγητής Ε.Μ.Π.	Συμεών Παπαβασιλείου Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούνιος 2018

.....

Βασίλειος Παπαευθυμίου
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Το διαδίκτυο αποτελεί αναμφισβήτητα μια από τις σπουδαιότερες εφευρέσεις του περασμένου αιώνα και έχει αναχθεί πλέον σε αναπόσπαστο κομμάτι της καθημερινότητας για μεγάλο μέρος του πληθυσμού. Έφερε μια επανάσταση στον τρόπο που οι άνθρωποι αντιλαμβάνονται τον κόσμο ενώ παράλληλα άλλαξε τον τρόπο που άνθρωποι και οργανισμοί αλληλεπιδρούν. Κοινό χαρακτηριστικό των προσφερόμενων διαδικτυακών εφαρμογών και υπηρεσιών είναι η ευρεία χρήση της αρχιτεκτονικής πελάτη-εξυπηρετητή. Πρόσφατα όμως έχει δημιουργηθεί μια νέα τάση αποκεντροποίησης της διαχείρισης των διαδικτυακών εφαρμογών. Προς αυτή την κατεύθυνση συντελεί και η ανάπτυξη ρηξικέλευθων τεχνολογιών όπως το Blockchain και ο σημασιολογικός ιστός.

Η τεχνολογία Blockchain εμφανίστηκε το 2008 ως η κινητήριος δύναμη του κρυπτονομίσματος Bitcoin. Από τότε έχει εξελιχθεί και έχει αποκτήσει ευρύ πεδίο εφαρμογών. Το blockchain είναι ένα ψηφιακό, κατανεμημένο, δημόσιο λογιστικό βιβλίο στο οποίο καταγράφονται συναλλαγές με τρόπο αδιάβλητο και το οποίο υποστηρίζεται από ένα δίκτυο ομότιμων κόμβων (P2P). Εκτός από μέσο για την λειτουργία κρυπτονομισμάτων, το Blockchain, μπορεί να αποτελέσει πυλώνα για την δημιουργία και λειτουργία αποκεντρωμένων εφαρμογών, εφαρμογών δηλαδή που βασίζονται σε ένα κατανεμημένο δίκτυο ομότιμων κόμβων και όχι στους εξυπηρετητές κάποιου οργανισμού. Χαρακτηριστικότερο παράδειγμα αποτελεί το Ethereum blockchain, μία πλατφόρμα ανάπτυξης αποκεντρωμένων εφαρμογών μέσω έξυπνων συμβολαίων.

Παρατηρώντας τις δυσκολίες που αντιμετωπίζονται από νέους και μη εμπορικούς καλλιτέχνες αλλά και την απουσία εφαρμογής που θα δίνει την δυνατότητα σε ερασιτέχνες μουσικούς, φωτογράφους κτλ. να κερδίζουν χρήματα μέσω της τέχνης τους, αποφάσισα να δημιουργήσω μια ηλεκτρονική πλατφόρμα ανταλλαγής αρχείων πολυμέσων μεταξύ χρηστών. Οι πωλητές θα έχουν την δυνατότητα να διανέμουν τα αρχεία τους στους καταναλωτές παρακάμπτοντας τους μεσάζοντες.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η διερεύνηση της τεχνολογίας του blockchain καθώς και η ανάπτυξη μιας αποκεντρωμένης εφαρμογής που θα βασίζεται στο Ethereum blockchain. Οι αγοραπωλησίες θα γίνονται απευθείας μεταξύ των χρηστών, οι οποίοι μπορούν να είναι ταυτόχρονα πωλητές και αγοραστές. Ο αγοραστής θα πρέπει, πριν αποκτήσει πρόσβαση στο περιεχόμενο, να πληρώσει απευθείας όλους τους συντελεστές – εμπλεκόμενους στην δημιουργία του με βάση τα προ-συμφωνημένα ποσοστά που αναλογούν στον κάθε ένα.

Κατά τη διάρκεια εκπόνησης της διπλωματικής, μελετήθηκε σε βάθος η τεχνολογία και ο τρόπος λειτουργίας του Ethereum blockchain, του σημασιολογικού ιστού (web3) και των αποκεντρωμένων εφαρμογών (Dapps). Επίσης, αναπτύχθηκαν έξυπνα συμβόλαια στην γλώσσα Solidity για την λειτουργία της εφαρμογής τα οποία «τρέχουν» στο Ethereum blockchain αλλά και ιστοσελίδες με γραφικό περιβάλλον για την χρήση της εφαρμογής. Τέλος δημιουργήθηκε ένα κρυπτονόμισμα ως εναλλακτική των Ethers, για τις συναλλαγές εντός εφαρμογής.

Λέξεις κλειδιά: Ethereum, blockchain, σημασιολογικός ιστός - Web3, έξυπνα συμβόλαια, αποκεντρωμένες εφαρμογές – Dapps.

Abstract

Undoubtedly the internet is one of the most important innovations of the previous century and it has become an integral part of our everyday life. It revolutionized the way people perceive the world, while at the same time, it has changed the way people and organizations interact. A common characteristic of most web applications and services is the wide use of the client-server architecture. However, there has been a recent trend to create decentralized web applications. The development of cutting-edge technologies such as the Blockchain and the semantic web also contributes to this direction.

Blockchain emerged in 2008 as the underlying technology of the cryptocurrency Bitcoin. Ever since, blockchain has evolved and has contributed to the creation of applications in manifold fields. Blockchain is a digital decentralized public ledger that records transactions in a verifiable and permanent way and works in a peer-to-peer system. Apart from being the means for the operation of cryptocurrencies, blockchain can be the pillar for the development and the operation of decentralized applications. These applications use a decentralized peer-to-peer system instead of the client-server scheme. The most typical example is the Ethereum blockchain, a platform for the deployment of decentralized applications through smart contracts.

Having noticed the difficulties faced by young and less commercial artists, as well as the inexistence of an application that would let amateur musicians, photographers etc. to earn money by selling their art, I decided to develop an electronic platform for the exchange of media files among users. In this way, sellers will have the ability to distribute their files to consumers without intermediaries.

The aim of this diploma thesis is to research the Blockchain technology as well as to develop a decentralized application based on the Ethereum blockchain. The transactions will be made directly between users, who can be simultaneously sellers and buyers. The buyer selects a file that they want to purchase and before gaining access to it, they pay directly all the contributors involved in its creation. For the files that have more than one intellectual property rights holder (e.g. singer, songwriter, guitar player etc.), the buyer has to pay directly each one of them with the pre-arranged amount of money, which is a percentage of the media file's total price.

During the development of this diploma thesis, I studied in depth the technologies of Ethereum blockchain, the semantic web (web3) and the way decentralized applications operate. Furthermore, I developed smart contracts in Solidity programming language for the operation of the decentralized application and created webpages for the users to interact with it. Finally, I created a specific-use cryptocurrency as an alternative to the Ether, which can be used in the transactions among users.

Keywords: Ethereum, blockchain, semantic web - Web3, smart contracts, decentralized applications – Dapps, cryptocurrencies.

Ευχαριστίες

Για την έως τώρα πορεία μου θέλω να ευχαριστήσω πρώτα από όλα τους γονείς μου που είναι πάντα αρωγοί των προσπαθειών μου, με εμπιστεύονται και με στηρίζουν σε ό,τι απόφαση έχω πάρει. Μου εμφύσησαν τις αξίες και το ήθος τους και χωρίς την αγάπη, την καθοδήγηση, τις θυσίες και την υπομονή τους θα ήμουν σίγουρα διαφορετικός άνθρωπος.

Θέλω επίσης να ευχαριστήσω τους καθηγητές μου και ιδιαίτερα την καθηγήτρια και επιβλέπουσα της διπλωματικής μου εργασίας κα. Θεοδώρα Βαρβαρίγου που με εμπιστεύτηκε και μου έδωσε την δυνατότητα να ανακαλύψω έναν τόσο ενδιαφέροντα και πολλά υποσχόμενο κλάδο. Ακόμα, την ευχαριστώ για τον χρόνο που μου διέθεσε και τη βοήθειά της που ήταν πολύτιμη όποτε τη χρειάστηκα.

Επίσης, θέλω να ευχαριστήσω θερμά τον υποψήφιο διδάκτορα Γιώργο Παλαιοκρασσά, τον διδάκτορα ερευνητή Αντώνη Λίτκε καθώς και τα υπόλοιπα μέλη της ομάδας εργαστηρίου Κατανεμημένης Γνώσης και Συστημάτων Πολυμέσων για τη σημαντική βοήθεια και την καθοδήγηση που μου πρόσφεραν απλόχερα.

Τέλος θέλω να ευχαριστήσω πολύ τους φίλους και τους συμφοιτητές μου για την αλληλοϋποστήριξη και την αλληλοβοήθεια που είχαμε όλα αυτά τα χρόνια. Σίγουρα, ένα από τα σημαντικότερα μαθήματα των ακαδημαϊκών μου χρόνων είναι η ομαδικότητα και η συνεργασία που αυθόρμητα επιτύχαμε.

Βασίλειος Παπαευθυμίου

Αθήνα, Ιούνιος 2018

Περιεχόμενα

Κατάλογος Σχημάτων

1

Εισαγωγή

Ο κόσμος έχει προ πολλού εισέλθει στην εποχή της ψηφιοποίησης και του διαδικτύου. Το διαδίκτυο έφερε μία επανάσταση σε πολλές πτυχές της καθημερινής ζωής και της οικονομίας. Στην αρχή ψηφιοποιήθηκε η πληροφορία και έτσι έγινε δυνατή η ταχύτατη διάδοσή της σε πολύ μεγάλες αποστάσεις μέσω του διαδικτύου. Συνέπεια αυτού αποτέλεσε η παγκοσμιοποίηση της γνώσης. Έπειτα δημιουργήθηκαν τα ηλεκτρονικά καταστήματα, συμβάλλοντας σε μεγάλο βαθμό στην ανάπτυξη του διεθνούς εμπορίου. Ακολούθησαν τα κοινωνικά δίκτυα τα οποία συνέδεσαν ανθρώπους από όλο τον κόσμο και έδωσαν δυνατότητα σε νέες μορφές επικοινωνίας και διαλόγου.

Αρχής γενομένης της δημιουργίας του Bitcoin το 2009, παρατηρήθηκε ένας κλονισμός στον τρόπο που ο κόσμος αντιλαμβάνεται το χρήμα και τις οικονομικές συναλλαγές. Άρχισε να γεννιέται μια τάση ψηφιοποίησης του χρήματος, η δημιουργία νέων νομισμάτων τα οποία δεν εκδίδονται ούτε ελέγχονται από κάποια κυβέρνηση ή κεντρική τράπεζα, αλλά η λειτουργία τους βασίζεται πάνω στην τεχνολογία του blockchain. Αυτά ονομάστηκαν κρυπτονομίσματα, λόγω των κρυπτογραφικών πρωτοκόλλων που χρησιμοποιούν για να διασφαλίζουν την απρόσκοπτη λειτουργία τους.

Το blockchain είναι ένα ψηφιακό κατανεμημένο δημόσιο λογιστικό βιβλίο στο οποίο καταγράφονται συναλλαγές με τρόπο επαληθεύσιμο και αδιάβλητο. Κάθε νέα συστάδα καταχωρήσεων ονομάζεται μπλοκ και συνδέεται με τα προηγούμενα ως το επόμενο κομμάτι της αλυσίδας. Ένα σύστημα blockchain βασίζει την λειτουργία του σε πολλούς υπολογιστές ανά τον κόσμο που κάθε ένας από αυτούς αποτελεί έναν κόμβο. Κάθε κόμβος έχει ακριβές αντίγραφο όλης της πληροφορίας που είναι καταγεγραμμένη στο blockchain από την αρχή της δημιουργίας του. Ό,τι γράφεται στο blockchain είναι αδύνατο να διαγραφεί ή να τροποποιηθεί. Έτσι, εξασφαλίζεται ότι ανά πάσα στιγμή θα παρουσιάζεται μια κοινή αλήθεια σε όλους τους εμπλεκόμενους. Ακόμα, χάρη στο γεγονός ότι οποιαδήποτε κακόβουλη μεταγενέστερη αλλαγή θα απαιτούσε συνωμοσία της πλειοψηφίας των κόμβων του δικτύου -πράγμα αδύνατο-, παρέχει ασφάλεια ικανή να παρακάμψει τις ενδιάμεσες έμπιστες αρχές που επικυρώνουν τις συναλλαγές μέχρι σήμερα. Έχει λοιπόν την δυναμική να αποκεντροποιήσει την διαχείριση εφαρμογών και υπηρεσιών, κάνοντας μερικά βήματα προς τον εκδημοκρατισμό του διαδικτυακού τοπίου.

Το blockchain έχει την δυναμική να επηρεάσει διάφορους τομείς, πέρα από τον χρηματοπιστωτικό, που χάρη στο bitcoin οφείλει μεγάλο μέρος της φήμης του. Αποκεντρωμένες εφαρμογές βασισμένες στο blockchain μπορούν να δώσουν νέες δυνατότητες στις χρηματικές συναλλαγές και στις αγοραπωλησίες, να βελτιώσουν την γραμμή παραγωγής μειώνοντας λειτουργικά κόστη, να προσφέρουν καινοτόμες

εφαρμογές για την διαχείριση πνευματικών δικαιωμάτων, την ανιχνευσιμότητα της προέλευσης τροφών και άλλων αγαθών, να δώσουν δυνατότητα για ασφαλείς διαδικτυακές εκλογές και δημοπρασίες και να βοηθήσουν υπηρεσίες κτηματολογίου [1].

Έχοντας ως αφορμή προβλήματα της σύγχρονης ζωής έχουν δημιουργηθεί εφαρμογές και πρωτοβουλίες που έχουν ως στόχο να ωφελήσουν πιο δραστικά την ζωή των ανθρώπων. Σε αυτές συγκαταλέγεται μια σύγχρονη προσπάθεια για την δημιουργία ενός αποκεντρωμένου ασφαλούς συστήματος ταυτοποίησης με στόχο την παροχή ψηφιακών ταυτοτήτων και επίσημων εγγράφων σε ανθρώπους που τα στερούνται. Σύμφωνα με την Παγκόσμια Τράπεζα 1.5 εκατομμύρια πολίτες κυρίως από τις αναπτυσσόμενες χώρες στερούνται επίσημων εγγράφων που πιστοποιεί την ταυτότητά τους και εξαιτίας αυτού γίνονται συχνά αντικείμενο εκμετάλλευσης. [2] Άλλη εφαρμογή με κοινωνικό πρόσημο αποτελεί η δημιουργία ενός συστήματος παρακολούθησης της χώρας προέλευσης διαμαντιών ώστε να αποφεύγονται όσα προέρχονται από περιοχές στις οποίες το εμπόριό τους προκαλεί πολεμικές αναταραχές («ματωμένα» διαμάντια) [1]. Τέλος, πρόσφατα ανακοινώθηκε η χρήση ενός κρυπτονομίσματος από την Unicef με στόχο την συγκέντρωση πόρων για φιλανθρωπίες. [3] [4] Πιο συγκεκριμένα, η Unicef ζητά, αντί για δωρεά σε χρήματα, την «δωρεά» υπολογιστικής ισχύος από τους χρήστες για την εξόρυξη του κρυπτονομίσματος Monero. Έτσι, οι χρήστες μπορούν να επισκεφθούν την ιστοσελίδα The hopepage [5] και να επιλεξουν το ποσοστό της υπολογιστικής ισχύος που θέλουν να προσφέρουν για την εξόρυξη του συγκεκριμένου κρυπτονομίσματος. Όσο έχουν ανοιχτή την ιστοσελίδα θα γίνεται η εξόρυξη. Τα κρυπτονομίσματα που συγκεντρώνονται θα μετατρέπονται σε χρηματικά ποσά τα οποία θα δωρίζονται στην οργάνωση ώστε να χρησιμοποιηθούν για φιλανθρωπικό σκοπό.

Βλέπουμε λοιπόν ότι πολλές διαφορετικές τεχνολογίες αναπτύσσονται παράλληλα και υπόσχονται να βελτιώσουν και να αλλάξουν άρδην την μορφή και την ποιότητα των υπηρεσιών του διαδικτύου.

1.1 Αρχική ιδέα και κοινωνική αξία της εφαρμογής

Τον τελευτάιο καιρό γινόμαστε μάρτυρες φαινομένων που αποτρέπουν τους νέους καλλιτέχνες ή τους ερασιτέχνες δημιουργούς από την επαγγελματική ενασχόληση με την τέχνη τους. Ένα από αυτά είναι η απροθυμία τον δισκογραφικών εταιρειών να αναλάβουν νέους ή λιγότερο εμπορικούς καλλιτέχνες αυτούς. Ακόμα και όταν δέχονται να εκδώσουν το έργο τους, η αμοιβή που αξιώνουν αποτελεί μεγάλο ποσοστό των κερδών, πράγμα που είναι κρίσιμο για τους καλλιτέχνες. Κάτι τέτοιο προφανώς δεν έχει μεγάλο αντίκτυπο όταν πρόκειται για καταξιωμένους και εμπορικούς καλλιτέχνες μιας και τα έσοδα που αξιώνει η εταιρεία αποτελούν μικρότερο ποσοστό των κερδών του δημιουργού. Συνέπεια του παραπάνω φαινομένου είναι πολλοί καλλιτέχνες να εντοπίζουν την αδυναμία οικονομικής τους στήριξης μέσω της τέχνης τους και να διστάζουν να ασχοληθούν επαγγελματικά με αυτήν.

Από την άλλη, παρατηρείται μια έλλειψη όσον αφορά την δυνατότητα των ερασιτεχνών δημιουργών να κερδίζουν χρήματα μέσα από τα προϊόντα της τέχνης τους. Υπάρχουν πολλοί άνθρωποι που παράγουν διάφορα έργα (φωτογραφίες, μουσικά κομμάτια, βίντεο) και θα ενδιαφέρονταν να διαθέσουν την τέχνη τους προς

πώληση χωρίς αυτό να συνεπάγεται επαγγελματική ενασχόληση. Όμως μέχρι τώρα δεν υπάρχει μια πλατφόρμα με ευρεία χρήση που θα τους έδινε το σύνολο των κερδών από την πώληση των αρχείων τους. Όσες υπάρχουν λειτουργούν ως μεσάζοντες και σχεδόν πάντα παρακρατούν μερίδιο της αξίας πώλησης του αρχείου.

Τέλος, όσον αφορά τους επαγγελματίες δημιουργούς, πρόσφατα παρατηρήθηκαν φαινόμενα κακοδιαχείρισης των κερδών από την διαχείριση των πνευματικών δικαιωμάτων του έργου τους. Πιο συγκεκριμένα, το 2017 αποκαλύφθηκε ότι η Ανώνυμη Εταιρεία Πνευματικής Ιδιοκτησίας είχε εισπράξει αλλά δεν είχε αποδώσει δικαιώματα στους δικαιούχους καλλιτέχνες ενώ ταυτόχρονα είχε οφειλές προς εργαζομένους, προς τρίτους και προς το ελληνικό Δημόσιο [6] [7].

Με αφορμή τις παραπάνω σκέψεις αλλά και τις δυνατότητες των αποκεντρωμένων εφαρμογών που βασίζονται στην τεχνολογία του blockchain, γεννήθηκε η ιδέα για την ανάπτυξη μιας εφαρμογής που θα παρακάμπτει τους μεσάζοντες στην διαχείριση πνευματικών δικαιωμάτων αλλά και στην πώληση των αρχείων πολυμέσων. Αυτή θα δίνει την δυνατότητα στο σύνολο των ατόμων που συνετέλεσαν στην δημιουργία ενός αρχείου να πληρωθούν άμεσα από τους καταναλωτές αλλά και στους ερασιτέχνες δημιουργούς να διαθέσουν τα αρχεία τους προς πώληση και να λάβουν το σύνολο της χρηματικής αξίας που δικαιούνται. Επιπλέον στόχο αποτέλεσε η δυνατότητα οι χρήστες να μην χρειάζεται να αποκαλύψουν την ταυτότητά τους ή να συνδέονται (sign up & sign in) όσο χρησιμοποιούν την εφαρμογή, αρκεί να πραγματοποιούν τις συναλλαγές τους μέσω του Ethereum λογαριασμού τους για να στέλνουν ή να δέχονται άμεσες και απευθείας πληρωμές.

1.2 Αντικείμενο της Διπλωματικής Εργασίας

Στην παρούσα διπλωματική εργασία πρωταγωνιστικό ρόλο θα καταλάβει η καινοτόμα τεχνολογία του blockchain και πιο συγκεκριμένα το Ethereum blockchain.

Σκοπός της διπλωματικής αποτελεί η αξιοποίηση της τεχνολογίας του Ethereum blockchain για την ανάπτυξη μίας πρωτοποριακής αποκεντρωμένης εφαρμογής (DApp – Decentralized Application) η οποία θα αποτελέσει μία πλατφόρμα διαμοιρασμού αρχείων πολυμέσων.

Οι κάτοχοι των πνευματικών δικαιωμάτων αρχείων πολυμέσων (αρχεία εικόνας, ήχου, βίντεο κτλ.) θα καταχωρούν τα αρχεία τους μαζί με τις απαραίτητες πληροφορίες (τίτλος, καλλιτέχνης, έτος δημιουργίας, τελική τιμή) αλλά και τους υπόλοιπους κατόχους πνευματικών δικαιωμάτων με τα ποσοστά της αμοιβής που πρέπει να λάβει ο καθένας από την κάθε πώληση του αρχείου. Οι υποψήφιοι αγοραστές μπορούν να βλέπουν όλα τα διαθέσιμα αρχεία με τις απαραίτητες πληροφορίες και να διαλέγουν σε ποιο επιθυμούν να αποκτήσουν πρόσβαση. Έτσι, πριν μπορέσουν να αποκτήσουν πρόσβαση στο επιθυμητό αρχείο πρέπει να καταβάλλουν το αντίτιμο που αναλογεί σε κάθε κάτοχο πνευματικού δικαιώματος του αρχείου εκτελώντας απευθείας συναλλαγές προς αυτούς. Οι αγοραστές έχουν την επιλογή να πληρώσουν είτε με Ethers (το προεπιλεγμένο κρυπτονόμισμα του Ethereum) είτε να πληρώσουν την αντίστοιχη τιμή σε NtuaTokens.

Είναι σημαντικό να τονιστεί ότι δημιουργήσαμε το κρυπτονόμισμα NtuaToken στα πλάισια της παρούσας εργασίας και προκειμένου να υλοποιηθεί η εφαρμογή χωρίς τη χρήση κάποιου από τα κρυπτονομίσματα που προσφέρονται προς πώληση

σε διάφορες πλατφόρμες αγοραπωλησίας κρυπτονομισμάτων. Η χρήση κάποιου άλλου κρυπτονομίσματος θα συνεπαγόταν το κόστος αγοράς του αλλά και χρήσης του για την υλοποίηση των συμβολαίων. Το NtuaToken έχει αναπτυχθεί αποκλειστικά για ερευνητικούς και πειραματικούς σκοπούς και δεν διατίθεται προς πώληση. Η ονομασία του προκύπτει από τα αρχικά National Technical University of Athens.

Η καινοτομία έγκειται στο γεγονός ότι για την υλοποίηση της εφαρμογής δεν θα χρησιμοποιηθούν αποκλειστικά παραδοσιακές μέθοδοι ανάπτυξης δικτυακού λογισμικού της αρχιτεκτονικής πελάτη – εξυπηρετητή, αλλά θα γίνει χρήση νέων τεχνολογιών, κυρίως του οικοσυστήματος του Ethereum, οι οποίες βασίζονται στην αρχιτεκτονική ομότιμων κόμβων (P2P). Δηλαδή η εφαρμογή θα ανήκει στην νέα κατηγορία εφαρμογών που ονομάζονται αποκεντρωμένες εφαρμογές (DApp – Decentralized Application).

Για τον ανωτέρω σκοπό θα γίνουν δύο διακριτές εργασίες. Η μία είναι η ανάπτυξη των έξυπνων συμβολαίων (smart contracts), τα οποία θα αποτελέσουν τον βασικό κορμό του συστήματος. Τα έξυπνα συμβόλαια θα περιέχουν τον κώδικα που θα εκτελείται στο Ethereum blockchain και στην ουσία θα υλοποιούν όλη την λειτουργικότητα και θα δώσουν στην εφαρμογή αποκεντρωμένο χαρακτήρα. Η άλλη εργασία είναι η ανάπτυξη του ιστοτόπου (website), μέσω του οποίου θα παρέχεται στους χρήστες ένα εύχρηστο περιβάλλον για την αλληλεπίδραση με το blockchain.

1.3 Οργάνωση κειμένου

Το κείμενο της διπλωματικής αποτελείται από 8 Κεφάλαια και 2 Παραρτήματα.

Το παρόν Κεφάλαιο, το οποίο αποτελεί την εισαγωγή.

Στο Κεφάλαιο 2 παρουσιάζεται το θεωρητικό υπόβαθρο της εργασίας. Πιο συγκεκριμένα, γίνεται σύντομη παρουσίαση του Web3, το οποίο είναι η πιο σύγχρονη εξέλιξη του διαδικτύου και αυτή που κάνει δυνατή την ύπαρξη αποκεντρωμένων εφαρμογών. Στην συνέχεια, παρουσιάζονται τα δίκτυα ομότιμων κόμβων (P2P) που αποτελούν συστατικό στοιχείο του Web3 και του blockchain. Εν συνεχεία, γίνεται αναφορά στην κρυπτογραφία ελλειπτικών καμπυλών, την οποία χρησιμοποιεί το Ethereum για την ασφάλεια του δικτύου. Τέλος, γίνεται αναλυτική παρουσίαση του Ethereum blockchain που αποτελεί την βασική τεχνολογία που χρησιμοποιήθηκε στην παρούσα διπλωματική εργασία.

Στο Κεφάλαιο 3 γίνεται περιγραφή των σημαντικότερων εργαλείων και τεχνολογιών που χρησιμοποιήθηκαν κατά την ανάπτυξη της εφαρμογής. Αρχικά, παρουσιάζονται εργαλεία που ανήκουν στην τεχνολογική στοίβα του Ethereum. Στην συνέχεια, παρουσιάζεται το σύστημα διαχείρισης πακέτων NPM, ακολουθούμενο από το server-side περιβάλλον εκτέλεσης JavaScript, Node.js και από τα εργαλεία ανάπτυξης ιστοσελίδων που χρησιμοποιήθηκαν. Τέλος, παρουσιάζεται η βάση δεδομένων MongoDB που χρησιμοποιήθηκε για την αποθήκευση των αρχείων πολυμέσων και η τεχνολογία GridFS που αξιοποιήθηκε για το ανέβασμα των αρχείων στην βάση δεδομένων και το κατέβασμα τους από αυτή.

Στο Κεφάλαιο 4 γίνεται αναφορά στο γενικότερο πλαίσιο ένταξης της εφαρμογής και αναλύονται οι λειτουργικές και μη λειτουργικές απαιτήσεις του συστήματος.

Στο Κεφάλαιο 5 γίνεται λεπτομερής παρουσίαση της διαδικασίας σχεδίασης και υλοποίησης των επιμέρους συστατικών της εφαρμογής. Αρχικά, γίνεται επεξήγηση της διαδικασίας σχεδίασης και ανάπτυξης των έξυπνων συμβολαίων και τονίζεται η σημασία τους. Εν συνεχεία, γίνεται αναλυτική παρουσίαση των δύο έξυπνων συμβολαίων που αναπτύχθηκαν στα πλαίσια της παρούσας διπλωματικής εργασίας, NtuaToken και Uploader. Έπειτα, γίνεται μια αναλυτική περιγραφή των βασικών διαδικασιών που γίνονται στα πλαίσια της εφαρμογής. Επίσης, εξηγείται ο λόγος ύπαρξης της βάσης δεδομένων για το ανέβασμα των αρχείων πολυμέσων αλλά και η επιλογή κάποια στοιχεία των αρχείων να αποθηκεύονται προσωρινά στους φυλλομετρητές. Τέλος παρουσιάζονται ορισμένα ποσοτικά χαρακτηριστικά που αφορούν ζητήματα απόδοσης και κόστους των συναλλαγών των έξυπνων συμβολαίων αλλά και τον χρόνο διάσχισης των blocks.

Στο Κεφάλαιο 6 παρουσιάζεται ο τρόπος λειτουργίας της εφαρμογής όσον αφορά τους χρήστες της. Παρατίθενται εικόνες από διάφορα σενάρια χρήσης της εφαρμογής. Έτσι, φαίνονται οι ιστοσελίδες που δημιουργήθηκαν, ο σκοπός τους και η λειτουργία τους.

Το Κεφάλαιο 7 αποτελεί τον επίλογο του κειμένου. Σε αυτό συνοψίζονται οι παρατηρήσεις και τα συμπεράσματα του συγγραφέα όσον αφορά την τεχνολογία του Ethereum blockchain αλλά και τις αποκεντρωμένες εφαρμογές. Επίσης, σημειώνονται ορισμένες μελλοντικές επεκτάσεις του συστήματος.

Το Κεφάλαιο 8 αποτελείται από την βιβλιογραφία που χρησιμοποιήθηκε για την σύνταξη του κειμένου της διπλωματικής εργασίας και την ανάπτυξη της εφαρμογής.

Το Παράρτημα Ι περιέχει αναλυτικές οδηγίες τόσο για την εγκατάσταση όσο για την χρήση της εφαρμογής. Στην πρώτη ενότητα αναφέρονται με σαφήνεια τα προαπαιτούμενα εργαλεία για την εγκατάσταση και λειτουργία της εφαρμογής. Στην δεύτερη ενότητα αναφέρονται οι απαιτήσεις από τον χρήστη, προκειμένου αυτός να είναι σε θέση να χρησιμοποιήσει την εφαρμογή.

Τέλος, στο Παράρτημα ΙΙ υπάρχει ο κώδικας των έξυπνων συμβολαίων καθώς και των ιστοσελίδων που αναπτύχθηκαν. Περιλαμβάνεται επίσης και υπερσύνδεσμος για τον υπόλοιπο κώδικα της εφαρμογής, ο οποίος δεν θα ήταν δυνατόν να συμπεριληφθεί στο παρόν κείμενο.

2

Θεωρητικό υπόβαθρο και σχετικές εργασίες

Στο κεφάλαιο αυτό διατυπώνεται το θεωρητικό υπόβαθρο σχετικά με την τεχνολογία του Ethereum blockchain. Παρουσιάζονται επίσης συγγενικές στο Blockchain τεχνολογίες ή τεχνολογίες που αποτελούν συστατικά του. Αρχικά, παρουσιάζεται η εξέλιξη του παγκόσμιου ιστού έως την σημερινή εποχή καθώς και η σύγχρονη τάση που επικρατεί, η οποία προστάζει την αποκεντροποίηση του παγκόσμιου ιστού – Web3. Στόχος είναι να τονιστεί η σημασία της τεχνολογίας του blockchain και ιδιαιτέρως των αποκεντρωμένων εφαρμογών (Decentralized Applications - DApp) του Ethereum. Εν συνεχεία, περιγράφεται η τεχνολογία των δικτύων ομότιμων κόμβων (P2P), καθώς αποτελούν θεμέλιο λίθο για την λειτουργία της τεχνολογίας blockchain. Έπειτα, γίνεται μία σύντομη αναφορά στην κρυπτογραφία ελλειπτικών καμπυλών, ως τον κύριο αλγόριθμο κρυπτογράφησης που χρησιμοποιείται από το Ethereum για την ασφάλεια των συναλλαγών και του δικτύου γενικότερα. Στην συνέχεια, γίνεται μια αναλυτική παρουσίαση του Ethereum Blockchain και των έξυπνων συμβολαίων που αποτελούν το σημαντικότερο κομμάτι της παρούσας διπλωματικής. Τέλος, γίνεται αναφορά σε εφαρμογές και εργασίες που είναι σχετικές με την παρούσα διπλωματική εργασία.

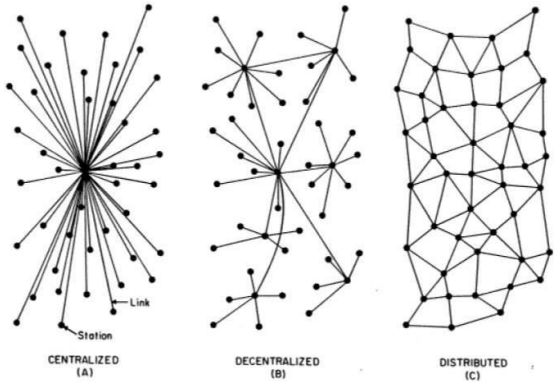
2.1 Web3 – Ο αποκεντρωμένος ιστός

Στις αρχές του 1990 ο παγκόσμιος ιστός WWW έφερε μία επανάσταση στην μετάδοση της πληροφορίας. Το Web1 αποτελεί την έκδοση του διαδικτύου που είναι σε ισχύ από το 1991 έως και το 2003. Αυτό είχε κατά βάση “read-only” δυνατότητες. Στην συνέχεια ο παγκόσμιος ιστός αναπτύχθηκε, έγινε πιο ώριμος και μετεξελίχθηκε στο λεγόμενο Web2, το οποίο εισήγαγε εφαρμογές όπως τα κοινωνικά δίκτυα και το ηλεκτρονικό εμπόριο. Το Web2, που έκανε την εμφάνισή του το 2004, έδωσε την δυνατότητα “read-write” και οι χρήστες μπορούν πλέον και να συνεισφέρουν στο περιεχόμενο του ιστού. Το βασικό χαρακτηριστικό του Web2 είναι πως σχεδόν σε κάθε εφαρμογή υπάρχει μία κεντρική οντότητα η οποία έχει τον πλήρη έλεγχο της λειτουργίας της εφαρμογής. Η οντότητα αυτή μπορεί να είναι εταιρεία, οργανισμός, κρατικός φορέας κ.ο.κ. Είναι χαρακτηριστικό ότι ακόμα και στις εφαρμογές P2P συνήθως υπάρχει κάποια οντότητα που εκτελεί χρέη διαχειριστή. Ο τρόπος λειτουργίας αυτός, προϋποθέτει την ύπαρξη εμπιστοσύνης από τους χρήστες προς τον οργανισμό αυτό.

Εικόνα 2-1 Χρονοδιάγραμμα εμφάνισης σημαντικών τεχνολογιών αποκέντρωσης (Decentralization Technologies)

Τον παραπάνω περιορισμό έρχεται να λύσει το Blockchain, το οποίο δείχνει να αποτελεί τον κύριο οδηγό της νέας γενιάς του διαδικτύου, του αποκεντρωμένου ιστού «Decentralized Web» ή αλλιώς Web3 [8]. Το Blockchain παρέχει την δυνατότητα P2P συναλλαγών με πλήρη εξάλειψη του ενδιάμεσου φορέα. Πρώτη εφαρμογή της τεχνολογίας blockchain αποτελεί το Bitcoin.

Διαπιστώνουμε πως διαμορφώνεται μία νέα τάση, η οποία επιτάσσει την μείωση της χρήσης κεντρικών εξυπηρετητών, χάριν της υιοθέτησης ενός αποκεντρωμένου μοντέλου. Με τον τρόπο αυτόν ελπίζουμε πως θα αντιμετωπίσουμε τα κακώς κείμενα του σημερινού κυρίαρχου μοντέλου πελάτη-εξυπηρετητή, που αφορούν σε μεγάλο βαθμό την ασφάλεια των δεδομένων, όπως η δυνατότητα κατάχρησής τους από τους οργανισμούς που τα κατέχουν, υποκλοπής τους, απώλειάς τους λόγω του μοναδικού σημείου αποτυχίας (Single Point of Failure) κτλ.



Εικόνα 2-2 Web3 Centralized vs Decentralized vs Distributed

Το blockchain θα αποτελέσει θεμέλιο λίθο του Web3 [9], όμως δεν θα είναι το μοναδικό «συστατικό» του, αφού δεν ενδείκνυται για την αποθήκευση μεγάλων ποσοτήτων δεδομένων για δύο λόγους: Ο πρώτος λόγος αποτελεί την επεκτασιμότητα και την ταχύτητά του. Το blockchain είναι αργό όταν το διατρέχουμε και δύσκολα επεκτάσιμο. Δεύτερος λόγος είναι η ιδιωτικότητα, μιας και όλη η αποθηκευμένη πληροφορία στο blockchain είναι ορατή σε όλους.

Σήμερα η τεχνολογία έχει φτάσει σε τέτοιο σημείο, ώστε να είναι δυνατή η ανάπτυξη και λειτουργία μίας αποκεντρωμένης εφαρμογής «DApp» (Decentralized Application) με μικρές υπολογιστικές και αποθηκευτικές δυνατότητες, καθώς και δυνατότητες πληρωμών, παράδειγμα τέτοιας εφαρμογής αποτελεί η εφαρμογή της παρούσας διπλωματικής. Παρακάτω παρουσιάζονται ενδεικτικά μερικές τεχνολογίες του οικοσυστήματος Web3 [10].

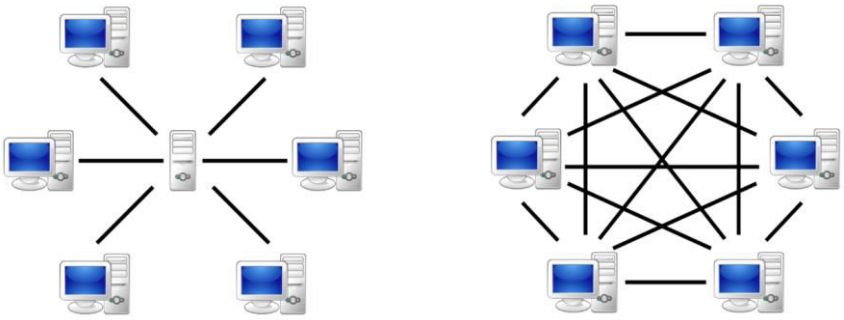
Εικόνα 2-3 Αφηρημένη στοίβα τεχνολογιών του Web3

Στην ενότητα αυτή, θα γίνει μία σύντομη παρουσίαση της αρχιτεκτονικής ομότιμων κόμβων, δεδομένου ότι αποτελεί την αρχιτεκτονική πάνω στην οποία βασίζεται η τεχνολογία του blockchain.

Τα κυρίαρχα μοντέλα δικτυακών εφαρμογών στο διαδίκτυο είναι δύο:

I. η αρχιτεκτονική πελάτη-εξυπηρετητή (client-server) και
II. η αρχιτεκτονική ομότιμων (peer-to-peer), για συντομία P2P.

Στην αρχιτεκτονική πελάτη-εξυπηρετητή υπάρχει πάντα ένας ενεργός υπολογιστής, ο εξυπηρετητής, ο οποίος εξυπηρετεί αιτήσεις για υπηρεσίες από άλλους υπολογιστές, τους πελάτες. Φαίνεται λοιπόν πως ο ρόλος του εξυπηρετητή για την λειτουργία σε τέτοιου είδους δίκτυα είναι καθοριστικός. Αντιθέτως, στην αρχιτεκτονική ομότιμων κόμβων υπάρχει μικρή ή καθόλου στήριξη σε αποκλειστικούς εξυπηρετητές ή σε κέντρα δεδομένων [11].



Εικόνα 2-4 α. Αρχιτεκτονική πελάτη-εξυπηρετητή β. Αρχιτεκτονική ομότιμων

Η αρχή λειτουργίας των δικτύων ομότιμων κόμβων είναι η απευθείας επικοινωνία ανάμεσα σε ζεύγη συνδεδεμένων υπολογιστών, που καλούνται ομότιμοι (peers). Οι ομότιμοι δεν ανήκουν σε κάποιον οργανισμό, αλλά είναι κατά κανόνα υπολογιστές που ελέγχονται από χρήστες. Οι ομότιμοι κόμβοι έχουν ίδια προνόμια, ίδιες δυνατότητες και ίδιο ρόλο στο δίκτυο. Οι κόμβοι λοιπόν διαθέτουν έναν μέρος των υπολογιστικών τους πόρων για την λειτουργία του δικτύου, απαλείφοντας έτσι την ανάγκη ύπαρξης μίας κεντρικής αρχής υπεύθυνης για τον συντονισμό και λειτουργία του δικτύου. Οι ομότιμοι κόμβοι έχουν διττό ρόλο. Λειτουργούν τόσο ως πελάτες (που καταναλώνουν πόρους) όσο και ως εξυπηρετητές (που προσφέρουν πόρους). Να σημειωθεί πως, πολύ συχνά, εφαρμογές βασίζονται σε μία υβριδική

αρχιτεκτονική, συνδυάζοντας αυτήν του πελάτη-εξυπηρετητή και των ομότιμων κόμβων.

Παραδείγματα εφαρμογών που βασίζονται σε αρχιτεκτονικές P2P περιλαμβάνουν μεταξύ άλλων κατηγορίες όπως, διανομή αρχείων (π.χ. BitTorrent), επιτάχυνση κατεβάσματος αρχείων μέσω βοήθειας ομότιμων κόμβων (π.χ. Xunlei) και τηλεφωνία διαδικτύου (π.χ. Skype). Το blockchain αποτελεί και αυτό μία τεχνολογία που βασίζεται στην αρχιτεκτονική ομότιμων κόμβων.

Τα δίκτυα ομότιμων χωρίζονται σε δύο μεγάλες κατηγορίες, στα αδόμητα και στα δομημένα.

Τα αδόμητα P2P δίκτυα δεν επιβάλλουν κάποια συγκεκριμένη δομή στο ανώτερο επίπεδο του δικτύου, αλλά σχηματίζονται μέσω κόμβων οι οποίοι πραγματοποιούν συνδέσεις τυχαία μεταξύ τους (πχ. Gnutella). Δεν υπάρχει κανένας συσχετισμός μεταξύ δεδομένων και των κόμβων που τα προσφέρουν. Μερικά πλεονεκτήματά τους είναι η εύκολη δημιουργία τους, η δυνατότητα τοπικών βελτιστοποιήσεων σε διαφορετικές περιοχές της τοπολογίας και τέλος η ευρωστία τους σε συνθήκες υψηλού ρυθμού αύξησης και αναχώρησης κόμβων (churn). Ορισμένα βασικά μειονεκτήματά τους είναι ότι προκαλούν πλημμύρα στο δίκτυο σε κάθε ερώτημα αναζήτησης δεδομένων. Το γεγονός αυτό αφενός οδηγεί σε αυξημένη κίνηση και κατασπατάληση των πόρων του δικτύου, αφετέρου δεν εξασφαλίζει την άφιξη του ερωτήματος στον κατάλληλο κόμβο και επομένως την απάντηση στο ερώτημα.

Στα δομημένα P2P δίκτυα το ανώτερο επίπεδο του δικτύου οργανώνεται σχηματίζοντας μία ορισμένη τοπολογία σύμφωνα με κάποιο πρωτόκολλο, γεγονός που εξασφαλίζει την αποδοτική αναζήτηση οποιωνδήποτε δεδομένων, από οποιονδήποτε κόμβο. Ο πιο διαδεδομένος τύπος δομημένου P2P δικτύου υλοποιεί έναν κατανεμημένο πίνακα κατακερματισμού (Distributed Hash Table – DHT), στον οποίο μέσω της χρήσης μίας παραλλαγής της «συνεπούς» συνάρτησης κατακερματισμού (consistent hashing[1]) πραγματοποιείται η ανάθεση κάθε αρχείου (δεδομένων) σε συγκεκριμένο κόμβο. Έτσι, η αναζήτηση αρχείων (δεδομένων) γίνεται αποτελεσματικά από κάθε κόμβο, με την χρησιμοποίηση του DHT. Το παραπάνω αποτελεί και το μεγαλύτερο πλεονέκτημα των δομημένων P2P δικτύων έναντι των αδόμητων. Λόγω αυτής της οργάνωσης του δικτύου, για την ομαλή δρομολόγηση της κίνησης, απαιτείται η τήρηση από κάθε κόμβο λιστών γειτόνων που ικανοποιούν συγκεκριμένα κριτήρια. Η ανάγκη αυτή κάνει το δίκτυο περισσότερο ευάλωτο σε συνθήκες υψηλού ρυθμού αύξησης και αναχώρησης κόμβων (churn).

Τα βασικότερα πλεονεκτήματα της αρχιτεκτονικής ομότιμων κόμβων:

- **Αυτο-κλιμακωσιμότητα** (self-scalability) **του δικτύου**, αποτελεί εγγενές χαρακτηριστικό της αρχιτεκτονικής P2P. Όσο περισσότεροι κόμβοι προστίθενται στο δίκτυο, τόσο αυξάνεται ο φόρτος εργασίας, η απαίτηση

[1] consistent hashing: Ένα ειδικό είδος κατακερματισμού, σύμφωνα με το οποίο όταν ένας πίνακας κατακερματισμού αλλάξει μέγεθος, μόνον K/n τα κλειδιά πρέπει να ξανά υπολογιστούν κατά μέσο όρο, όπου K είναι ο αριθμός των κλειδιών και n ο αριθμός των θέσεων. Σε αντίθεση με τους παραδοσιακούς πίνακες κατακερματισμού όπου αλλαγή του αριθμού των θέσεων συνεπάγεται επανυπολογισμό σχεδόν όλων των κλειδιών.

δηλαδή σε πόρους. Ταυτόχρονα αυξάνονται και οι διαθέσιμοι πόροι λόγω της διττής φύσης του κάθε ομότιμου κόμβου.

- **Μείωση κόστους**, καθώς συνήθως δεν απαιτείται σημαντική υποδομή και εύρος ζώνης εξυπηρετητή.
- **Μη ύπαρξη μοναδικού σημείου αστοχίας του δικτύου**. Δηλαδή η βλάβη σε έναν κόμβο δεν επηρεάζει την λειτουργία του υπόλοιπου δικτύου, όπως γίνεται στην αρχιτεκτονική πελάτη-εξυπηρετητή, όπου αστοχία του εξυπηρετητή, συνεπάγεται μη διαθεσιμότητα της εφαρμογής.

Μερικές από τις βασικές προκλήσεις που αντιμετωπίζουν οι εφαρμογές αρχιτεκτονικής ομότιμων κόμβων:

- **Μη φιλικότητα προς τους ISP (Internet Service Providers).** Οι περισσότεροι ISP έχουν διαστασιοποιηθεί για ασύμμετρη χρησιμοποίηση του εύρους ζώνης, δηλαδή για περισσότερη συρρευματική, παρά αντιρρευματική κίνηση, δυσκολεύοντας έτσι την παροχή πόρων προς το σύστημα από τους ομότιμους κόμβους.
- **Ασφάλεια.** Λόγω της κατανεμημένης και ανοικτής φύσης τους, οι εφαρμογές P2P μπορούν να δημιουργήσουν προβλήματα ασφαλείας. Θα δούμε στην συνέχεια πώς στο blockchain το πρόβλημα αυτό λύνεται με την χρήση της κρυπτογραφίας.
- **Κίνητρα.** Ποια κίνητρα έχουν οι κόμβοι, ώστε να παρέχουν πόρους στο σύστημα. Και πάλι θα δούμε πως τα blockchain συστήματα κρυπτονομισμάτων (Ethereum, Bitcoin) παρέχουν (χρηματικά) κίνητρα στους κόμβους που υποστηρίζουν την λειτουργία του δικτύου (miners).

2.2 Κρυπτογραφία ελλειπτικών καμπυλών

Το Ethereum (όπως και το Bitcoin) χρησιμοποιεί την κρυπτογραφία ελλειπτικών καμπυλών [12] για την υπογραφή των συναλλαγών, συμβάλλοντας έτσι στην ασφάλεια των συναλλαγών των χρηστών.

Η θεωρία της κρυπτογραφίας ελλειπτικών καμπυλών (Elliptic Curve Cryptography – ECC) προτάθηκε πρώτη φορά το 1985 από τους Victor Miller (IBM) and Neil Koblitz (University of Washington) ως ένας εναλλακτικός μηχανισμός για την υλοποίηση της κρυπτογραφίας δημόσιου κλειδιού [13]. Το μεγάλο πλεονέκτημα της ECC είναι το γεγονός ότι βασίζεται σε διακριτούς λογαρίθμους και συνεπώς είναι πολύ δυσκολότερο να παραβιαστεί σε σχέση με άλλους γνωστούς αλγορίθμους δημόσιων κλειδιών όπως ο RSA.

Στην παρούσα εργασία δεν θα αναλυθεί ο τρόπος λειτουργίας του αλγορίθμου της κρυπτογραφίας ελλειπτικών καμπυλών, αφού αφορά ένα πολύ εξειδικευμένο πεδίο το οποίο δεν εμπίπτει στο εύρος της παρούσας διπλωματικής. Λεπτομέρειες για τον τρόπο λειτουργίας μπορούν να βρεθούν στις εξής πηγές: [14] και [15].

Θεωρείται ότι σήμερα ο αλγόριθμος ψηφιακής υπογραφής ελλειπτικών καμπυλών (Elliptic Curve Digital Signature Algorithm – ECDSA), που χρησιμοποιεί Ethereum για κρυπτογράφηση, είναι απαραβίαστος. Έτσι λοιπόν διασφαλίζεται η ασφάλεια των συναλλαγών και θεωρούμε ότι το Ethereum είναι θωρακισμένο απέναντι σε επιθέσεις. Σε περίπτωση που υπάρχει μεγάλη ανάπτυξη στην κβαντική υπολογιστική (Quantum Computing), ο αλγόριθμος αυτός θα σταματήσει να

προσφέρει ασφάλεια. Για την αντιμετώπιση αυτού του ενδεχομένου, αναμένεται να παρουσιαστούν κάποιες λύσεις (Lamport signatures), στην έκδοση Constantinople [12].

2.3 Ethereum Blockchain

Ερμηνεία: Το **blockchain** είναι ένα ψηφιακό, κατανεμημένο, δημόσιο καθολικό (ή λογιστικό βιβλίο - ledger), μέσω του οποίου καταγράφονται με ασφάλεια συναλλαγές, συμφωνίες, συμβόλαια και γενικώς οτιδήποτε χρειάζεται να έχει καταγραφεί και να είναι διαθέσιμο προς επαλήθευση.

Ερμηνεία: **Κατανεμημένο σύστημα** είναι μία συλλογή από ανεξάρτητους υπολογιστές, οι οποίοι εμφανίζονται στους χρήστες τους ως ένα ενιαίο συνεκτικό σύστημα [16].

Ερμηνεία: **Κρυπτονόμισμα** είναι ένα ψηφιακό νόμισμα το οποίο ασφαλίζει τις συναλλαγές με κρυπτογραφικό κώδικα, που βασίζεται στην υπολογιστική ισχύ του hardware για την εκτέλεσή του (proof of work) ή λιγότερο ενεργειακά απαιτητικούς τρόπους, όπως το proof of stake [12].

Όπως αναφέρθηκε και προηγουμένως, το blockchain είναι μία από τις τεχνολογίες που αποτελούν τον κορμό ανάπτυξης του Web3, της νέας γενιάς του διαδικτύου. Αποτελεί μία πολύ αποτελεσματική λύση στο πρόβλημα της εμπιστοσύνης που αντιμετωπίζουν οι άνθρωποι (όπως αυτό αναφέρεται στην προηγούμενη ενότητα). Η τεχνολογία του blockchain μας δίνει την δυνατότητα να εμπιστευόμαστε τις εξόδους του δικτύου, χωρίς παράλληλα να εμπιστευόμαστε κανέναν από τους συμμετέχοντες του αυτού.

Το blockchain λειτουργεί στο διαδίκτυο, πάνω σε ένα δίκτυο ομότιμων (P2P) κόμβων που «τρέχουν» το πρωτόκολλο και διατηρούν ένα πιστό αντίγραφο του καθολικού των συναλλαγών/δεδομένων, επιτρέποντας την πραγματοποίηση συναλλαγών χωρίς την παρουσία κάποιας ενδιάμεσης έμπιστης αρχής (Trusted Third Party TTP) παρά μόνο με την βοήθεια των υπολοίπων κόμβων του δικτύου.

Στην συνέχεια θα εξετάσουμε το Ethereum και θα αναδείξουμε τις αρετές του έναντι του Bitcoin, του κρυπτονομίσματος που «γέννησε» την τεχνολογία του blockchain [17].

Η καινοτομία του Ethereum σε σχέση με το Bitcoin, είναι πως το Ethereum αποτελεί μία πιο ευέλικτη και προσαρμόσιμη πλατφόρμα, πάνω στην οποία μπορούν να δημιουργηθούν και να λειτουργήσουν με ασφάλεια αποκεντρωμένες εφαρμογές, ενώ το Bitcoin παρέχει κυρίως την δυνατότητα (οικονομικών) συναλλαγών με χρήση του κρυπτονομίσματος (Bitcoin) [18].

Ακόμα, το Ethereum από την ίδρυσή του είχε σαν στόχο να αποτελέσει την κατάλληλη πλατφόρμα για την ανάπτυξη έξυπνων συμβολαίων και αποκεντρωμένων εφαρμογών. [19]

Το blockchain του Ethereum είναι μία Turing complete κατανεμημένη υπολογιστική αρχιτεκτονική, στην οποία κάθε κόμβος του δικτύου εκτελεί και καταγράφει τις ίδιες συναλλαγές, οι οποίες οργανώνονται σε μπλοκ και προστίθενται στο blockchain. Μόνο ένα μπλοκ μπορεί να προστεθεί κάθε φορά και κάθε μπλοκ περιέχει την απόδειξη εργασίας (Proof of Work) [20] [21], αν και λόγω προβλημάτων που έχουν προκύψει (μεγάλη κατανάλωση ισχύος) συζητείται η μετάβαση από το

Proof of Work στο Proof of Stake [22]. Οι κόμβοι που συντηρούν το δίκτυο, δηλαδή αυτοί που δημιουργούν τα μπλοκ ονομάζονται miners.

Το Ethereum Virtual Machine

Όπως αναφέρθηκε και πριν, το Ethereum δεν παρέχει στους χρήστες απλά ένα προκαθορισμένο σύνολο λειτουργιών, όπως κάνει το Bitcoin, αλλά αντιθέτως τους παρέχει την δυνατότητα να ορίσουν δικές τους λειτουργίες και κατ' επέκταση έξυπνα συμβόλαια και αποκεντρωμένες εφαρμογές DApps, είναι δηλαδή ένα προγραμματιζόμενο blockchain.

Το Ethereum είναι κατά μία έννοια μία σουίτα πρωτοκόλλων που καθορίζουν μία πλατφόρμα για την ανάπτυξη και λειτουργία αποκεντρωμένων εφαρμογών. Στο επίκεντρο βρίσκεται το Ethereum Virtual Machine ("EVM"), το οποίο μπορεί να εκτελεί κώδικα αυθαίρετης αλγοριθμικής πολυπλοκότητας. Το EVM είναι Turing complete.

Όπως και τα άλλα blockchain, το Ethereum περιλαμβάνει ένα πρωτόκολλο δικτύου ομότιμων κόμβων. Το πρωτόκολλο αυτό είναι υπεύθυνο για τον συντονισμό των συνδεδεμένων κόμβων, με σκοπό την απρόσκοπτη λειτουργία του δικτύου. Κάθε κόμβος «τρέχει» το EVM και εκτελεί τις ίδιες εντολές. Λόγω αυτού, το Ethereum αναφέρεται συχνά και ως «παγκόσμιος υπολογιστής».

Η μεγάλη αυτή παραλληλοποίηση των υπολογισμών δεν έχει σκοπό την απόδοση, αφού οι υπολογισμοί είναι πολύ πιο αργοί και ακριβοί απ' ότι θα γινόταν σε έναν απλό υπολογιστή, σκοπό έχει την ύπαρξη ομοφωνίας/συναίνεσης (consensus). Αυτό το χαρακτηριστικό δίνει στο Ethereum πολύ μεγάλη αντοχή σε σφάλματα, αδιάλειπτη λειτουργία και εγγυάται ότι τα δεδομένα μέσα στο blockchain θα μείνουν αμετάβλητα.

Ο τρόπος λειτουργίας του Ethereum

Η βασική μονάδα στο Ethereum είναι ο λογαριασμός (account), σε αντίθεση με το Bitcoin blockchain όπου βασική μονάδα είναι η συναλλαγή. Στο Ethereum blockchain, παρακολουθείται η κατάσταση κάθε λογαριασμού και όλες οι μεταβάσεις κατάστασης είναι μεταβιβάσεις αξίας και πληροφορίας μεταξύ λογαριασμών. Υπάρχουν δύο είδη λογαριασμών:

- Οι Externally Owned λογαριασμοί (EOAs), οι οποίοι ελέγχονται από ιδιωτικά κλειδιά
- Οι λογαριασμοί συμβολαίων, οι οποίοι ελέγχονται από τον κώδικα του συμβολαίου και μπορούν να ενεργοποιηθούν μόνο από έναν EOA.

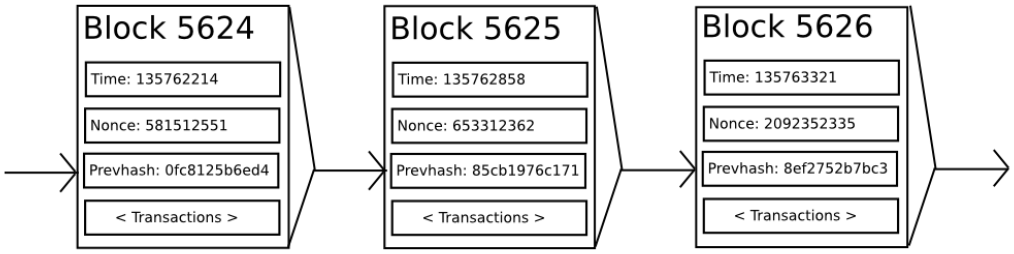
Δηλαδή, οι EOA ελέγχονται από τους ανθρώπους που κατέχουν και ελέγχουν τα ιδιωτικά κλειδιά, ενώ οι λογαριασμοί συμβολαίων ελέγχονται από τον κώδικά τους. Το πολυσυζητημένο έξυπνο συμβόλαιο «smart contract» είναι πρακτικά ο κώδικας του λογαριασμού συμβολαίου, δηλαδή το πρόγραμμα που εκτελείται όταν μία συναλλαγή σταλεί στον λογαριασμό αυτόν. Οι χρήστες που κατέχουν EOA, μπορούν να δημιουργήσουν νέα συμβόλαια, δημοσιεύοντάς τα στο blockchain.

Οι χρήστες (λογαριασμοί EOA) στέλνουν συναλλαγές στο δίκτυο του Ethereum, υπογράφοντας τα δεδομένα της συναλλαγής με το ιδιωτικό τους κλειδί, χρησιμοποιώντας την κρυπτογραφία ελλειπτικών καμπυλών (ECDSA – Elliptic

Curve Digital Signature Algorithm). Μία συναλλαγή είναι έγκυρη μόνο εάν είναι υπογεγραμμένη από τον αποστολέα της (από το ιδιωτικό του κλειδί). Σαν αποτέλεσμα, το δίκτυο είναι σίγουρο ότι ο αποστολέας της συναλλαγής είναι αυτός που ισχυρίζεται και όχι κάποιος κακόβουλος χρήστης.

Για κάθε συναλλαγή θα πρέπει να πληρωθεί ένα μικρό τέλος στο δίκτυο (gas). Αυτό προστατεύει το δίκτυο, αφού κάνει ασύμφορες τις επιπόλαιες εντολές υπολογισμού καθώς και κακόβουλες επιθέσεις, όπως τις επιθέσεις άρνησης υπηρεσίας (DDoS). Η πληρωμή γίνεται για τον υπολογισμό και την μνήμη που χρησιμοποιεί η πραγματοποίηση της συναλλαγής και είναι ανάλογη αυτών. Το τέλος αυτό (gas) πληρώνεται στο κρυπτονόμισμα του Ethereum, το ether.

Τα ανωτέρω τέλη εισπράττονται από τους κόμβους που επικυρώνουν το δίκτυο, τους miners. Οι miners είναι κόμβοι του δικτύου που λαμβάνουν, διαδίδουν, επικυρώνουν και εκτελούν συναλλαγές. Συγκεντρώνουν ορισμένες συναλλαγές κάθε φορά σε μπλοκ και στη συνέχεια ανταγωνίζονται τους υπόλοιπους miners ώστε αυτοί να το εισάγουν στο blockchain. Κάθε φορά που ένας miner εισάγει ένα νέο μπλοκ στο blockchain λαμβάνει τα ether που αντιστοιχούν στις συναλλαγές που περιέχει το μπλοκ. Αυτό το ποσόν είναι που δίνει κίνητρο στους miners να εκτελούν την εργασία αυτή.



Εικόνα 2-5 Δομή των μπλοκ [19]

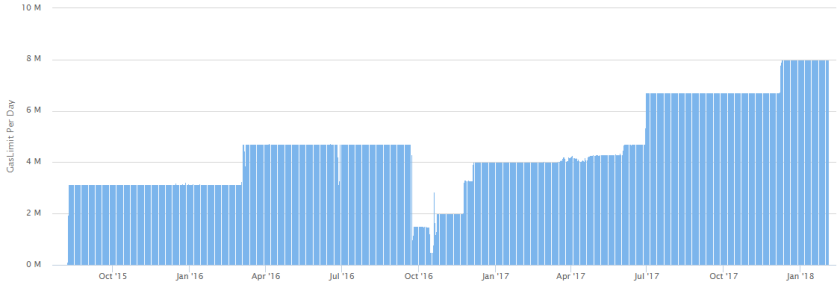
Για να εισαχθεί ένα μπλοκ στο blockchain θα πρέπει να συνοδεύεται από την απόδειξη εργασίας (nonce). Αυτό αποτελεί την λύση ενός δύσκολου μαθηματικού προβλήματος, όπως γίνεται και στο Bitcoin. Η διαφοροποίηση του Ethereum έγκειται στο γεγονός ότι το πρόβλημα αυτό έχει μεγάλες απαιτήσεις μνήμης, έτσι για το «mining» είναι απαραίτητα και η CPU και η μνήμη, ενώ στο Bitcoin αυτό δεν γίνεται (απαραίτητη μόνο κάρτα γραφικών). Η απόφαση αυτή των σχεδιαστών του Ethereum οδηγεί σε ένα πιο αποκεντρωμένο δίκτυο, αφού αποθαρρύνει την χρήση ειδικού hardware (πχ. ASICs), όπως έγινε με το Bitcoin. Το αποτέλεσμα λοιπόν, είναι ότι η απόδειξη εργασίας (PoW – Proof of Work) του Ethereum είναι πιο ανθεκτική απέναντι σε ASICs, οδηγώντας σε μεγαλύτερη αποκεντροποίηση του δικτύου, δηλαδή σε μεγαλύτερη ασφάλεια· πολλοί και μικροί miners, παρά λίγοι και ισχυροί [23].

Η επικοινωνία μεταξύ των ομότιμων κόμβων που «τρέχουν» τον Ethereum client γίνεται σύμφωνα με το πρωτόκολλο ĐΞVp2p Wire Protocol [24], [25]. Δεν θα αναλυθεί ο τρόπος λειτουργίας του πρωτοκόλλου, διότι αφορά πολύ εξειδικευμένα ζητήματα τα οποία δεν εμπίπτουν στο εύρος της διπλωματικής εργασίας.

Το κρυπτονόμισμα Ether του Ethereum έχει σαν ελάχιστη υποδιαίρεση το Wei. Ένα ether ισούται με 10^{18} wei.

Ο μηχανισμός παραγωγής των μπλοκ

Στο Ethereum blockchain τα μπλοκς δεν έχουν σταθερό μέγεθος ούτε δημιουργούνται ανά τακτά χρονικά διαστήματα. Έχουν όμως όριο στο gas κάθε μπλοκ. Το gas limit περιορίζει και το μέγεθος του μπλοκ αλλά και την υπολογιστική ισχύ που απαιτείται για την δημιουργία του κάθε μπλοκ. Το όριο του gas για κάθε μπλοκ προκύπτει μετά από ψηφοφορία μεταξύ των miners, δηλαδή μπορεί να αλλάζει με την πάροδο του χρόνου, έτσι μπορεί αλλάζει και το μέγεθος του μπλοκ. Το gas είναι η μονάδα που χρησιμοποιεί το Ethereum για την μέτρηση της υπολογιστικής προσπάθειας. Για παράδειγμα, η πρόσθεση δύο αριθμών κοστίζει 3 gas, ο υπολογισμός της τιμής κατακερματισμού κοστίζει 30 gas κτλ. [26].



Εικόνα 2-6 Gas limit μπλοκ (Πηγή: etherscan.io)

Εικόνα 2-7 Μέγεθος μπλοκ (Πηγή: etherscan.io)

Το χρονικό διάστημα που μεσολαβεί μεταξύ δύο διαδοχικών μπλοκ δεν είναι σταθερό και εξαρτάται από το επίπεδο της δυσκολίας του δικτύου. Ισχύουν οι εξής σχέσεις:

$$block\_time = current\_block\_timestamp - parent\_block\_timestamp$$

$$currentBlockDifficulty = parentBlockDifficulty + \frac{parentBlockDifficulty}{2048} * \max\left[\left(1 - \frac{blockTime}{10}\right), -99\right] + \text{floor}\left(\frac{currentBlockNumber}{100000} - 2\right)$$

Όπου floor είναι ο μεγαλύτερος ακέραιος αριθμός που είναι μικρότερος από τον περιεχόμενο αριθμό. Επίσης όλες οι διαιρέσεις είναι ακέραιες.

Από την παραπάνω σχέση βλέπουμε ότι εάν ο χρόνος που μεσολαβεί της δημιουργίας δύο διαδοχικών μπλοκ είναι μικρότερος από 10 δευτερόλεπτα η δυσκολία θα αυξηθεί. Εάν αυτός ο χρόνος είναι από 10 έως 19 δευτερόλεπτα η

δυσκολία δεν θα μεταβληθεί (αν εξαιρέσουμε τον παράγοντα difficulty bomb). Ενώ εάν είναι μεγαλύτερος από 19 δευτερόλεπτα η δυσκολία θα μειωθεί.

Ο τελευταίος όρος (floor(…)) ονομάζεται difficulty bomb και η ύπαρξή του έχει σαν αποτέλεσμα την σταδιακή αύξηση της δυσκολίας παράλληλα με την αύξηση του αριθμού των μπλοκ.

Με την παραπάνω τεχνική ο χρόνος του κάθε μπλοκ σχεδόν παραμένει μέσα στο προκαθορισμένο διάστημα 10 έως 19 sec.

Εικόνα 2-8 Χρονικό διάστημα μεταξύ δύο μπλοκ (Πηγή: etherescan.io)

Εικόνα 2-9 Δυσκολία μπλοκ (Πηγή: etherscan.io)

2.4 Σχετικές εργασίες

Στην ενότητα αυτήν θα παρουσιαστούν εργασίες σχετικές με το αντικείμενο της παρούσας διπλωματικής. Έχει προηγηθεί η παρουσίαση του Ethereum blockchain, έτσι λοιπόν εδώ δεν θα παρουσιαστούν εργασίες που αφορούν τον τρόπο λειτουργίας του. Αντιθέτως, θα παρουσιαστούν εργασίες και εφαρμογές που βασίζονται στην τεχνολογία του blockchain για την δημιουργία αποκεντρωμένων εφαρμογών με στόχο τις ανταλλαγή δεδομένων και αγοραπωλησίες μεταξύ χρηστών, προστασία πνευματικής ιδιοκτησίας αλλά και ένα σύστημα διαχείρισης φήμης χρηστών. Το κοινό σε όλες τις παρακάτω εφαρμογές είναι η αποκεντρωμένη φύση τους και η παράκαμψη των ενδιάμεσων έμπιστων αρχών (TTP). Τέλος, παρουσιάζεται και η εφαρμογή ujo music που αποτελεί παραπλήσια ιδέα με την παρούσα διπλωματική.

A Decentralised Sharing App running a Smart Contract on the Ethereum Blockchain [27]

Η ερευνητική ομάδα δημιούργησε μια αποκεντρωμένη εφαρμογή για την ανταλλαγή καθημερινών αντικειμένων η οποία βασίζεται στην τεχνολογία έξυπνων συμβολαίων του Ethereum blockchain. Τα έξυπνα συμβόλαια που υλοποιούνται στο

Ethereum προσφέρουν στους χρήστες τη δυνατότητα εγγραφής και ενοικίασης πολλαπλών συσκευών χωρίς την ανάγκη μιας ενδιάμεσης έμπιστης αρχής (TTP). Επιτυγχάνεται προστασία δεδομένων και αποφυγή επαναλαμβανόμενης εγγραφής (sign up) του χρήστη για κάθε πλατφόρμα. Η ομάδα χρησιμοποίησε μια εφαρμογή Ιστού βασισμένη σε ένα έξυπνο συμβόλαιο που εκτελείται σε δίκτυο δοκιμών Ethereum (Ethereum test network) για την κοινή χρήση των αντικειμένων, ένα δημόσιο κλειδί Ethereum που προσδιορίζει κάθε χρήστη χωρίς την ανάγκη εγγραφής (sign up) και αποκάλυψης προσωπικών οικονομικών δεδομένων. Ο ιδιοκτήτης του αντικειμένου είναι υπεύθυνος για την καταχώρηση του αντικειμένου σε ένα ζευγάρι κλειδιού-τιμής (key-value pair), όπου η τιμή είναι το όνομα του αντικειμένου με τις ιδιότητές του και το κλειδί είναι ένα αριθμητικό αναγνωριστικό (ID) που περιέχεται επίσης σε έναν QR κωδικό. Ο ενοικιαστής σαρώνει τον QR κωδικό του αντικειμένου που θέλει να νοικιάσει και αποθηκεύει την τρέχουσα χρονική σήμανση (timestamp) μετά την επιβεβαίωση της συναλλαγής Ether. Μετά το πέρας του συμφωνημένου χρονικού διαστήματος, ο ιδιοκτήτης έχει τη δυνατότητα να ανακτήσει το αντικείμενο. Όλες οι παραπάνω λειτουργίες είναι δημόσια διαθέσιμες στο blockchain, αντικαθιστώντας το TTP μιας και το δίκτυο Ethereum εγγυάται ότι το συμβόλαιο εκτελείται αναλόγως.

Intellectual-Property Blockchain-based Protection Model for Microfilms [28]

Η συγκεκριμένη δημοσίευση προτείνει την ανάπτυξη ενός μοντέλου προστασίας πνευματικής ιδιοκτησίας για μικροφίλμ χρησιμοποιώντας blockchain. Τα μικροφίλμ ορίζονται ως ταινίες μικρού μήκους και χαμηλού κόστους που είναι πολύ γνωστές στην Κίνα. Πολύ συχνά, τα δικαιώματά τους παραβιάζονται καθώς πολλά από αυτά έχουν αντιγραφεί και μεταδοθεί χωρίς την απαραίτητη άδεια από τους παραγωγούς τους. Τα δικαιώματα πνευματικής ιδιοκτησίας δεν μπορούν να προστατευθούν επαρκώς από τις παραδοσιακές πλατφόρμες εγγραφής. Επομένως, το blockchain φαίνεται ιδανικό για την προστασία τους χάρη στο γεγονός ότι οι πληροφορίες που καταχωρούνται σε αυτό δεν μπορούν να τροποποιηθούν χωρίς να εντοπιστούν, πολλαπλά αντίγραφα του ίδιου περιεχομένου θα αποθηκεύονται σε πολλαπλές τοποθεσίες και αυτές θα συγχρονίζονται με την εκτέλεση ενός πρωτοκόλλου συναίνεσης (consensus protocol) ώστε να διασφαλίζεται ότι όλοι έχουν την ίδια εικόνα των δεδομένων. Σύμφωνα με την προτεινόμενη διαδικασία συναλλαγών, όταν ένας χρήστης Α θέλει να δει το μικροφίλμ ενός χρήστη Β, το σύστημα θα ελέγξει αν το υπόλοιπο του χρήστη Α είναι αρκετό για να πληρώσει τον χρήστη Β για την προβολή της ταινίας. Αν ναι, η συναλλαγή εκτελείται, ο χρήστης Β πληρώνεται πριν ο χρήστης Α προσπελάσει το περιεχόμενο και η συναλλαγή καταχωρείται στο blockchain.

Thing-to-thing electricity micro payments using blockchain technology [29]

Η ομάδα χρησιμοποίησε το Bitcoin ως την πιο εξέχουσα εφαρμογή blockchain για να παρουσιάσει μια απόδειξη (Proof of concept – PoC) [30] της εφαρμογής ενός έξυπνου καλωδίου που συνδέεται με μια έξυπνη πρίζα και χωρίς καμία ανθρώπινη παρέμβαση πληρώνει την κατανάλωση ηλεκτρικής ενέργειας. Το κόστος της ηλεκτρικής ενέργειας για οτιδήποτε συνδέεται στο καλώδιο μπορεί να καταβληθεί σε

Bitcoins από το έξυπνο καλώδιο. Κάθε καλώδιο διαθέτει το δικό του λογαριασμό Bitcoin και ο χρήστης δεν θα χρειάζεται να γνωρίζει τις πληρωμές εκτός εάν ο λογαριασμός Bitcoin εξαντληθεί. Τότε, χρειάζεται να μεταφέρει χρήματα για να συνεχιστεί η λειτουργία του συστήματος. Τα πλεονεκτήματα της χρήσης του blockchain για αυτήν την εφαρμογή είναι η δυνατότητα υποστήριξης πολυάριθμων αυτόνομων συναλλαγών, η εύκολη δημιουργία λογαριασμού για κάθε καλώδιο και το γεγονός ότι δεν υπάρχει κεντρική αρχή που να ελέγχει τους λογαριασμούς. Για να μειωθεί ο αντίκτυπος των αθροιστικά υψηλών χρεώσεων συναλλαγής (transaction fee) κατά την πραγματοποίηση μικρο-συναλλαγών στο δίκτυο Bitcoin, η ομάδα παρουσίασε ένα πρωτόκολλο μικροπληρωμών που συγκεντρώνει πολλαπλές μικρότερες πληρωμές σε μία μεγαλύτερη συναλλαγή που χρειάζεται μόνο μία χρέωση συναλλαγής.

Towards blockchain-based intelligent transportation systems [31]

Στην συγκεκριμένη δημοσίευση, παρουσιάζεται η άποψη ότι το blockchain μπορεί να φέρει την επανάσταση στα έξυπνα συστήματα μεταφορών (Inteligent Transportation Systems ITS) συμβάλλοντας στη δημιουργία αποκεντρωμένων εφαρμογών. Ακόμα διεξάγεται μια προκαταρκτική μελέτη των έξυπνων συστημάτων μεταφορών που βασίζονται σε τεχνολογία blockchain (Blockchain-based ITS - B^2ITS). Η μελέτη περίπτωσης μιας "Uber-like" [32] υπηρεσίας συνεπιβίβασης σε πραγματικό χρόνο που βασίζεται σε Blockchain παρουσιάζεται ως εξής:

Η La'zooz στοχεύει να δημιουργήσει ένα παγκόσμιο αποκεντρωμένο δίκτυο συνεπιβίβασης για να βοηθήσει τους χρήστες της να επωφεληθούν από τα αχρησιμοποίητα κενά καθίσματα και τον άδειο χώρο φορτίου των οχημάτων. Οι χρήστες που ταξιδεύουν στις ίδιες διαδρομές μπορούν να μοιράζονται το ίδιο όχημα χρησιμοποιώντας αυτό το αποκεντρωμένο δίκτυο που ανήκει στην κοινότητα και γίνεται διαχειρίσιμο από αυτή. Η αποκεντρωμένη εφαρμογή (Dapp) του La'zooz χρησιμοποιεί ως «κοινοτικούς» κόμβους υπολογιστών (αποκαλούμενους και road-miners) τα smartphones και τους φορητούς υπολογιστές των χρηστών για να λαμβάνει δεδομένα μεταφορών. Για να παροτρύνει τους χρήστες να προσφέρουν τις συσκευές τους για τη λειτουργία του συστήματος, η εφαρμογή τους επιβραβεύει με μάρκες (tokens) που μπορούν να μετατραπούν σε χρήματα.

A concept for a decentralized rights management system based on blockchain [33]

Αυτή η δημοσίευση προτείνει ένα σύστημα διαχείρισης δικαιωμάτων βασισμένο στην τεχνολογία blockchain και περιγράφει τη δοκιμαστική εφαρμογή ενός συστήματος διαχείρισης δικαιωμάτων που βασίζεται σε Blockchain (BRIGHTS) σε βίντεο, χρησιμοποιώντας bitcoin. Το blockchain θα μπορούσε να προσφέρει έναν φθηνότερο τρόπο για την προστασία των δικαιωμάτων των βίντεο από ενδεχόμενες επιθέσεις, λαμβάνοντας υπόψη ότι οι πάροχοι υπηρεσιών δεν θα είναι οι μόνοι που θα έχουν την ευθύνη να διατηρούν το σύστημα, όπως συμβαίνει στα κεντρικά συστήματα. Τα βίντεο θα συσχετιστούν στενά στο blockchain με τις πληροφορίες σχετικά με τα δικαιώματα και όταν ένας χρήστης ζητήσει άδεια για να παρακολουθήσει ένα βίντεο, οι πληροφορίες δικαιωμάτων του βίντεο θα

περιλαμβάνονται στη συναλλαγή που εκδίδεται από τον εκδότη της άδειας. Η κρυπτογράφηση διασφαλίζει ότι μόνο ο κάτοχος άδειας χρήσης μπορεί να έχει το κλειδί άδειας χρήσης, ακόμη και αν το κλειδί είναι στο δημόσιο blockchain. Ο εκδότης της άδειας μπορεί να ελέγχει και να αλλάζει την άδεια για συγκεκριμένο κάτοχο που χρησιμοποιεί το συγκεκριμένο βίντεο. Τα ζητήματα καθυστέρησης (latency) παρακάμπτονται, αλλάζοντας τον αλγόριθμο Proof-of-work - POW του bitcoin και προσαρμόζοντας το μέσο διάστημα μεταξύ της προσθήκης νέων μπλοκ σε πέντε δευτερόλεπτα αντί για δέκα λεπτά.

Decentralizing Privacy: Using Blockchain to Protect Personal Data [34]

Στην δημοσίευση αυτή, περιγράφεται ένα αποκεντρωμένο σύστημα διαχείρισης προσωπικών δεδομένων. Η εφαρμογή χρησιμοποιεί blockchain για να δημιουργήσει έναν αυτοματοποιημένο διαχειριστή ελέγχου πρόσβασης που παρακάμπτει τις ενδιάμεσες έμπιστες αρχές (TTP) και διασφαλίζει ότι οι χρήστες κατέχουν και ελέγχουν τα προσωπικά τους δεδομένα. Κατά συνέπεια, οι χρήστες διαχειρίζονται τι είδους πρόσβαση θα μπορούσε να έχει μια εφαρμογή παροχής υπηρεσιών στα δεδομένα τους. Το προτεινόμενο σύστημα έχει τρεις οντότητες: τους χρήστες (χρήστες κινητών τηλεφώνων), τις υπηρεσίες (εφαρμογές) και τους κόμβους που είναι φορείς επιφορτισμένοι με τη συντήρηση του blockchain και ενός κατανεμημένου χώρου αποθήκευσης ιδιωτικών δεδομένων τύπου κλειδιού-τιμής (key-value). Το blockchain δέχεται τη διαχείριση ελέγχου πρόσβασης (Taccess) και τις συναλλαγές αποθήκευσης και ανάκτησης δεδομένων. Ο χρήστης και η υπηρεσία διερευνούν τα δεδομένα από το blockchain (Tdata), το οποίο επαληθεύει εάν έχουν πρόσβαση σε αυτό. Οι χρήστες μπορούν να αλλάξουν ανά πάσα στιγμή τα δικαιώματα που έχουν χορηγηθεί σε μια υπηρεσία (Taccess).

Rep on the block: A next generation reputation system based on the blockchain [35]

Η δημοσίευση προτείνει ένα γενικό σύστημα φήμης που βασίζεται στο blockchain και έχει ως στόχο να αποτρέψει πιθανές επιθέσεις στα σύγχρονα συστήματα φήμης το οποίο μπορεί να εφαρμοστεί σε οποιοδήποτε δίκτυο. Η συγκεκριμένη ομάδα δημιούργησε ένα εντελώς νέο blockchain που αποθηκεύει δεδομένα φήμης από ολοκληρωμένες συναλλαγές. Το σύστημα αποθηκεύει στην ανατροφοδότηση (feedback) φήμης είτε "1" για μια συναλλαγή στην οποία ο χρήστης έλαβε το απαιτούμενο αρχείο είτε "0" για μια μη ικανοποιητική συναλλαγή. Οι miners εξασφαλίζουν ότι η φήμη που δίνεται από έναν χρήστη βασίζεται σε μια πραγματική συναλλαγή, καθώς επικοινωνούν με κάθε χρήστη που εμπλέκεται στη συναλλαγή και ζητούν υπογεγραμμένη απόδειξή της. Οι miners συγκεντρώνουν στη συνέχεια αυτές τις επαληθευμένες συναλλαγές σε ένα μπλοκ από άλλες συναλλαγές πριν τις επιβεβαιώσουν με μια παρόμοια μέθοδο όπως το Bitcoin. Προκειμένου να αποφευχθεί η άδικη απόκτηση φήμης, κάθε χρήστης λαμβάνει μια μέση βαθμολογία φήμης από όλες τις συναλλαγές που μπορεί να έχει με κάποιον άλλο χρήστη.

Ujo music [36]

Η πλατφόρμα Ujo music δημιουργήθηκε το 2015 και αποτελεί μια αποκεντρωμένη εφαρμογή που δίνει την δυνατότητα σε μουσικούς να πωλούν τα τραγούδια τους μέσω της αυτής. Βασίζει την λειτουργία της στο Ethereum blockchain και έχει ως στόχο να παρακάμψει τους μεσάζοντες μεταξύ μουσικών και ακροατών.

Bloomen [37]

Το ερευνητικό project Bloomen υλοποιείται υπό την αιγίδα της Ευρωπαϊκής Ένωσης και βασικός συντελεστής του είναι η ομάδα εργαστηρίου Κατανεμημένης Γνώσης και Συστημάτων Πολυμέσων της σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου. Πρωταρχικός του στόχος είναι η υλοποίηση ενός συστήματος που θα δίνει την δυνατότητα σε δημιουργικούς ανθρώπους να ανταμείβονται για την δουλειά τους. Η έρευνα γίνεται στην νεότευκτη τεχνολογία του blockchain η οποία χρησιμοποιείται για την δημιουργία μια αποκεντρωμένης εφαρμογής. Χάρη στο γεγονός ότι η τεχνολογία αυτή δεν είναι ακόμα σε ώριμο στάδιο, μελετώνται πολλές διαφορετικές και ανταγωνιστικές πλατφόρμες για τον εντοπισμό της καταλληλότερης, που είναι και αυτή που θα χρησιμοποιηθεί. Πιο συγκεκριμένα, ερευνώνται τρεις λύσεις που αφορούν την μουσική βιομηχανία, τα αρχεία πολυμέσων και την διαδικτυακή τηλεόραση και αναπτύσσεται λογισμικό και οι κατάλληλες πλατφόρμες. Στόχος της εφαρμογής είναι να φέρει σε άμεση επαφή τους δημιουργούς με τους καταναλωτές του περιεχομένου και να προσφέρει λύσεις που θα δίνουν την δυνατότητα στους δημιουργούς να εμπορεύονται τα αρχεία των οποίων κατέχουν τα πνευματικά δικαιώματα, χωρίς την βοήθεια μεσαζόντων. Τέλος, μελετώνται τρόποι διαχείρισης πνευματικών δικαιωμάτων αλλά και δίκαιης και ασφαλούς αμοιβής των δημιουργών.

Deploying blockchains for a new paradigm of media experience [38]

Σήμερα παρατηρείται το φαινόμενο οι χρήστες να δημοσιεύουν αρχεία πολυμέσων στα κοινωνικά δίκτυα χωρίς να έχουν αποτελεσματικό έλεγχο στο ποιος μπορεί να επαναχρησιμοποιήσει το περιεχόμενο αυτό. Προτείνεται μια υπηρεσία που χρησιμοποιεί blockchain για τον αποτελεσματικό εντοπισμό της επαναχρησιμοποίησης αρχείων, την υλοποίηση ασφαλών συναλλαγών μεταξύ χρηστών και την δυνατότητα πώλησης των αρχείων από τους δημιουργούς τους. Η υλοποίηση αυτή αναδεικνύει τα προτερήματα της τεχνολογίας blockchain ως βάση δεδομένων για διαχείριση αρχείων.

Η συγκεκριμένη δημοσίευση έχει συνταχθεί από μέλη της εργαστηριακής ομάδας Κατανεμημένης Γνώσης και Συστημάτων Πολυμέσων της σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου με συμμετοχή και του συγγραφέα της παρούσας εργασίας. Την στιγμή συγγραφής του παρόντος κειμένου, η δημοσίευση έχει κατατεθεί στο συνέδριο GECON 2018 [39] και είναι σε κατάσταση αξιολόγησης.

3

Ανάλυση Απαιτήσεων Συστήματος

Σε αυτό το κεφάλαιο θα καταγραφούν λεπτομέρειες που αφορούν τον σκοπό του συστήματος, τις κατηγορίες των χρηστών, τις παραδοχές που έγιναν, καθώς και οι λειτουργικές και μη λειτουργικές απαιτήσεις του συστήματος. Η διαδικασία αυτή αποτελεί μέρος της προεργασίας που γίνεται κάθε φορά που κάποιος οργανισμός σκοπεύει να παράξει ή να αναθέσει σε τρίτο την δημιουργία σύνθετου λογισμικού. Στο στάδιο αυτό καθορίζονται με σαφήνεια, ακρίβεια και πληρότητα οι λειτουργίες τις οποίες το λογισμικό πρέπει να υλοποιήσει ώστε να εξυπηρετήσει τους μελλοντικούς του χρήστες.

Σκοπός του συστήματος

Σκοπός του συστήματος είναι η παροχή μιας αποκεντρωμένης εφαρμογής (DApp – Decentralized Application) που θα δίνει την δυνατότητα αγοραπωλησίας αρχείων πολυμέσων μεταξύ χρηστών. Οι χρήστες μπορούν να αγοράζουν ή/και να πωλούν αρχεία των οποίων κατέχουν τα πνευματικά δικαιώματα. Κάθε χρήστης ταυτοποιείται από την διεύθυνσή του στο δίκτυο του Ethereum. Συνεπώς, δεν χρειάζεται να χρησιμοποιεί την εφαρμογή ταυτοποιούμενος με τα προσωπικά του έγγραφα (π.χ. ταυτότητα, διαβατήριο κτλ.). Πιο συγκεκριμένα, σκοπός της παρούσας εφαρμογής είναι να δοθεί η δυνατότητα στους χρήστες να καταχωρούν στο δίκτυο του Ethereum blockchain τα στοιχεία των αρχείων που διαθέτουν προς πώληση καθώς και τους υπόλοιπους κατόχους των πνευματικών δικαιωμάτων του κάθε αρχείου. Έτσι, για κάθε αρχείο θα συμφωνείται εκ των προτέρων η αμοιβή που θα λάβει ο κάθε εμπλεκόμενος στην δημιουργία του και οι αγοραστές θα πληρώνουν άμεσα τον κάθε έναν από αυτούς πριν αποκτήσουν πρόσβαση στο αρχείο.

Η εφαρμογή αυτή έχει στόχο να παρακάμψει τους μεσάζοντες (π.χ. εταιρείες διαχείρισης πνευματικών δικαιωμάτων, δισκογραφικές εταιρείες και εφαρμογές όπως iTunes, Spotify κτλ.) μεταξύ δημιουργού και αγοραστή καθώς και να παρέχει σημαντική ασφάλεια ώστε να μην χρειάζονται ενδιάμεσες έμπιστες αρχές για την διασφάλιση και την επικύρωση των συναλλαγών (π.χ. τράπεζες, PayPal κτλ.).

Η εφαρμογή είναι ιδανική για νέους και μη εμπορικούς καλλιτέχνες καθώς γι' αυτούς έχει μεγάλη σημασία να λαμβάνουν το πλήρες ποσό από την διάθεση των αρχείων τους. Ακόμα, αυτοί είναι που αντιμετωπίζουν την μεγαλύτερη δυσκολία να υπογράψουν κάποιο ικανοποιητικό συμβόλαιο με τις εταιρείες-μεσάζοντες. Παρόλα αυτά, έχει την δυναμική να χρησιμοποιηθεί και από εμπορικούς καλλιτέχνες μιας και μια σημαντική μερίδα αυτών είναι συχνά δυσαρεστημένη από τις δισκογραφικές εταιρείες [40]. Τέλος, είναι μια εφαρμογή που δίνει την δυνατότητα σε ερασιτέχνες

δημιουργούς να διαθέσουν τα αρχεία τους στους καταναλωτές με εύκολο τρόπο, καλύπτοντας ένα σημαντικό κενό από την απουσία αντίστοιχης εφαρμογής.

Κατηγορίες χρηστών

Ανώνυμοι χρήστες:

Δεν έχουν λογαριασμό στο Ethereum. Επισκέπτονται την ιστοσελίδα από οποιονδήποτε φυλλομετρητή, μπορούν να περιηγηθούν σε αυτήν και να δουν όλα τα αρχεία πολυμέσων που είναι διαθέσιμα προς πώληση.

Αγοραστές:

Έχουν κάποιον λογαριασμό (EOA) Ethereum. Επισκέπτονται την ιστοσελίδα μέσω κάποιου φυλλομετρητή ο οποίος τους δίνει πρόσβαση στις κατανεμημένες εφαρμογές του Ethereum (πχ. Metamask plugin). Έχουν πρόσβαση σε ό,τι και οι ανώνυμοι χρήστες και επιπλέον:

- Έχουν την δυνατότητα να αγοράζουν αρχεία πολυμέσων που είναι διαθέσιμα πληρώνοντας απευθείας τους κατόχους πνευματικών δικαιωμάτων των αρχείων είτε σε ethers είτε στο κρυπτονόμισμα NtuaToken που δημιουργήθηκε ειδικά για χρήση στην εφαρμογή.

Πωλητές:

Έχουν κάποιον λογαριασμό (EOA) Ethereum. Όπως και οι αγοραστές επισκέπτονται την ιστοσελίδα με κατάλληλο φυλλομετρητή. Έχουν πρόσβαση σε ό,τι και οι ανώνυμοι χρήστες και επιπλέον:

- Έχουν την δυνατότητα να καταχωρούν νέα αρχεία προς πώληση στο σύστημα.
- Έχουν την δυνατότητα να ανεβάζουν τα αρχεία προς πώληση στην βάση δεδομένων και να παίρνουν το url στο οποίο θα είναι διαθέσιμο το αρχείο.
- Οφείλουν να καταχωρούν και τους υπόλοιπους κατόχους πνευματικών δικαιωμάτων του αρχείου που ανεβάζουν καθώς και το ποσοστό της αμοιβής που λαμβάνει ο καθένας από την κάθε πώληση του αρχείου.
- Έχουν την δυνατότητα να επεξεργάζονται χαρακτηριστικά των αρχείων που έχουν καταχωρίσει στο σύστημα (π.χ. αλλαγή της τιμής πώλησης, αλλαγή του ύψους της αποζημίωσης κάθε κατόχου πνευματικών δικαιωμάτων, αλλαγή του url στο οποίο θα είναι διαθέσιμο το αρχείο).

Παραδοχές:

Η εφαρμογή θα έχει εκπαιδευτικό χαρακτήρα και όχι εμπορικό. Συνεπώς, έχουν γίνει κάποιες παραδοχές που στόχο έχουν να μειώσουν την πολυπλοκότητα της υλοποίησης, ώστε να φανούν καλύτερα τα χαρακτηριστικά των τεχνολογιών που χρησιμοποιήθηκαν. Οι παραδοχές που έγιναν και αφορούν στην σχεδίαση και στην λειτουργία του συστήματος είναι οι εξής:

- Το Ethereum blockchain προσφέρει ασφάλεια στις συναλλαγές. Η παραδοχή αυτή υιοθετείται από το σύνολο της κοινότητας, αφού οι αλγόριθμοι κρυπτογράφησης που χρησιμοποιούνται θεωρούνται ασφαλείς. Έτσι δεν χρειάζεται μέριμνα από μέρους της εφαρμογής για επιπρόσθετη ασφάλεια.
- Οι πωλητές θα καταχωρούν αρχεία των οποίων κατέχουν τα πνευματικά δικαιώματα και δεν λαμβάνουμε υπόψιν την πιθανότητα κάποιος χρήστης να καταχωρεί αρχείο που δεν του ανήκει.
- Δεν υπάρχουν κακόβουλοι δράστες που υποδύονται τους πωλητές αρχείων πολυμέσων που δεν υπάρχουν. Η επίλυση αυτού του προβλήματος θα εμπεριείχε διαδικασίες πιστοποίησης των πωλητών, κάτι που ούτως ή άλλως δεν αποτελεί χαρακτηριστικό των αποκεντρωμένων εφαρμογών.
- Στην εφαρμογή μας έχουμε εισάγει την δυνατότητα οι κάτοχοι των πνευματικών δικαιωμάτων κάθε αρχείου να είναι δύο. Αυτό σημαίνει ότι για την αγορά κάθε αρχείου μπορούν να πληρωθούν μέχρι δύο ιδιοκτήτες. Αυτό έγινε για λόγους απλούστερης υλοποίησης και είναι κλιμακώσιμο σε όσους ιδιοκτήτες επιθυμούμε.
- Επιλέξαμε η ισοτιμία του NtuaToken με το Ether να είναι ένα προς ένα. Εύκολα μπορούμε να την αλλάξουμε μέσα από το πρόγραμμά μας.
- Το URL της βάσης δεδομένων στο οποίο είναι διαθέσιμο το αρχείο θα αποθηκεύεται στο blockchain. Γνωρίζουμε ότι, χάρη στην διαφάνεια που προσφέρει το δίκτυο του blockchain, το url είναι δυνατόν να διαβαστεί και από χρήστες που δεν έχουν προβεί σε αγορά του αρχείου. Σε αντίθετη περίπτωση έπρεπε να εισάγουμε μεγαλύτερη πολυπλοκότητα στην βάση δεδομένων και να ζητάμε από αυτή το url κάθε φορά που γίνεται κάποια αγορά. Η μέθοδος αυτή μειώνει την αποκεντρωμένη φύση της εφαρμογής και θα εξεταστεί μελλοντικά ως επέκταση της παρούσας εφαρμογής. Για τον λόγο αυτό όμως, κάθε φορά που γίνεται κάποια συναλλαγή, η οποία απαιτεί την διαπίστευση του χρήστη, ζητάμε από τον αποστολέα να υπογράψει την συναλλαγή μέσω του Metamask (συνάρτηση web3.eth.sign). Έτσι, μετά μπορούμε να επιβεβαιώσουμε ότι ο αγοραστής είναι και αυτός που αποκτά πρόσβαση στο αρχείο.

3.1 Λειτουργικές απαιτήσεις

Οι λειτουργικές απαιτήσεις εξασφαλίζουν ότι η εφαρμογή θα έχει τα ζητούμενα χαρακτηριστικά και πως θα παρέχει στους χρήστες της τις κατάλληλες λειτουργίες. Οι λειτουργικές απαιτήσεις λοιπόν της εφαρμογής είναι οι εξής:

- Η εφαρμογή να επικοινωνεί με το Ethereum blockchain.
- Να μην υπάρχει συμβατικό login, αλλά να γίνεται διαπίστευση του κάθε χρήστη μόνο μέσω των εργαλείων που προσφέρει το Ethereum blockchain.
- Να μην ζητείται ποτέ και από κανέναν χρήστη το ιδιωτικό κλειδί του Ethereum λογαριασμού του.
- Να δίνεται στους πωλητές/ιδιοκτήτες η δυνατότητα καταχώρησης νέου αρχείου.

- Να δίνεται στους πωλητές/ιδιοκτήτες αρχείων η δυνατότητα επεξεργασίας στοιχείων όπως η τιμή πώλησης, το url στην βάση καθώς και η διαγραφή κάποιου καταχωρημένου αρχείου. Το σύστημα να επιβεβαιώνει ότι μόνο κάποιος από τους ιδιοκτήτες του αρχείου μπορεί να προβεί στις παραπάνω αλλαγές.
- Οι αγοραστές να μπορούν να επιλέξουν ποιο αρχείο θα αγοράσουν από το ID του αρχείου στο blockchain που θα είναι μοναδικό.
- Όλα τα διαθέσιμα αρχεία να παρουσιάζονται στους χρήστες, με μέρος των πληροφοριών τους που είναι αποθηκευμένες στο δίκτυο του blockchain (ID αρχείου, τίτλος, όνομα δημιουργού, τελική τιμή, χρονολογία δημιουργίας).

Ακολουθούν οι λειτουργικές απαιτήσεις των έξυπνων συμβολαίων, τα οποία και θα αποτελέσουν τον κορμό της αποκεντρωμένης εφαρμογής.

Οι λειτουργικές απαιτήσεις **του έξυπνου συμβολαίου NtuaToken**, το οποίο θα υλοποιεί τις λειτουργίες του κρυπτονομίσματος NtuaToken:

- Να είναι διαθέσιμο το υπόλοιπο της κάθε διεύθυνσης (χρήστη) σε NTUA Tokens.
- Να παρέχεται εξαρχής ένα δεδομένο ποσό NtuaTokens στους χρήστες το οποίο θεωρητικά έχει προαγοραστεί από αυτούς πληρώνοντας σε Ethers.
- Να παρέχει την δυνατότητα μεταφοράς NTUA Tokens από μία διεύθυνση (χρήστη) σε άλλη διεύθυνση (χρήστη).
- Να παρέχει την δυνατότητα να υπολογίζουμε κάθε στιγμή το υπόλοιπο σε NtuaTokens συγκεκριμένου λογαριασμού.

Οι λειτουργικές απαιτήσεις του **έξυπνου συμβολαίου Uploader**, το οποίο θα υλοποιεί τις λειτουργίες καταχώρησης των αρχείων, αλλαγής των δεδομένων τους, διαγραφής τους αλλά και τη λειτουργία αγοράς και πώλησης των αρχείων:

- Να δίνει την δυνατότητα στους πωλητές να καταχωρίσουν το αρχείο που διαθέτουν μαζί με όλα τα απαραίτητα στοιχεία που αφορούν τις συναλλαγές (υπόλοιποι συνιδιοκτήτες, ποσό αποζημίωσης καθενός, τελική τιμή) αλλά και στοιχεία που θα προσδιορίζουν το αρχείο (τίτλος, καλλιτέχνης, έτος δημιουργίας)
- Να δίνει την δυνατότητα στους πωλητές να αλλάξουν κάποια στοιχεία από ήδη καταχωρημένο αρχείο του οποίου διαθέτουν τα πνευματικά δικαιώματα.
- Να δίνει την δυνατότητα στους πωλητές να διαγράψουν κάποιο ήδη καταχωρημένο αρχείο του οποίου διαθέτουν τα πνευματικά δικαιώματα.
- Να γίνεται έλεγχος αν οι όποιες αλλαγές στο καταχωρημένο αρχείο ή η διαγραφή του ζητείται από κάποιον από τους ιδιοκτήτες και μόνο τότε να ολοκληρώνεται.
- Να αποθηκεύονται στο Ethereum blockchain τα στοιχεία των αρχείων όπως τα καταχώρησε ο χρήστης.

- Να δίνει την δυνατότητα στους αγοραστές να αγοράσουν κάποιο από τα διαθέσιμα αρχεία και αφού γίνει έλεγχος για την επιτυχή συναλλαγή να τους παρέχει το url στο οποίο είναι διαθέσιμο το αρχείο.

3.2 Μη λειτουργικές απαιτήσεις

Ακολουθούν οι μη λειτουργικές απαιτήσεις που αφορούν ποιοτικά χαρακτηριστικά της εφαρμογής.

Ασφάλεια – Ακεραιότητα:

- Δεδομένου ότι στην εφαρμογή αυτή θα πραγματοποιούνται οικονομικές συναλλαγές θα πρέπει να υπάρχει υψηλό επίπεδο ασφάλειας.
- Τα χρήματα – κρυπτονομίσματα των χρηστών πρέπει να είναι ασφαλή απέναντι σε επιθέσεις.
- Οι συναλλαγές μεταξύ των χρηστών πρέπει και αυτές να είναι ασφαλείς και να διασφαλίζεται η ακεραιότητά τους.
- Τα έξυπνα συμβόλαια θα πρέπει να είναι απολύτως ασφαλή. Δεν επιτρέπεται να υπάρξει κανένα κενό ασφαλείας (bug) στον κώδικά τους, μιας και αυτός δεν θα μπορέσει να αλλάξει, αλλά οποιαδήποτε αλλαγή επιβάλλει την υλοποίηση του συμβολαίου εκ νέου.

Ευελιξία:

- Η εφαρμογή να μπορεί εύκολα να επεκταθεί, χωρίς να απαιτούνται πολλοί επιπλέον πόροι. (scalability)
- Επίσης η υλοποίηση να είναι ευέλικτη, δηλαδή να «γενικεύσει» την έννοια του αρχείου, έτσι ώστε με κάποια πιθανή επιπλέον εργασία να μπορούμε να μετατρέψουμε την εφαρμογή για πώληση π.χ. δεδομένα πραγματικού χρόνου (δεδομένα αισθητήρων, IoT συσκευών κτλ.).
- Τα έξυπνα συμβόλαια θα πρέπει να έχουν βελτιστοποιημένο κώδικα, ώστε να πραγματοποιούνται οι συναλλαγές γρήγορα και με μικρό κόστος gas (προμήθεια του δικτύου).

Απόδοση – Αποκρισιμότητα:

- Υπάρχουν περιπτώσεις που τα στοιχεία που έχουν σταλεί από το blockchain προς τις ιστοσελίδες δεν έχουν αλλάξει ή απλά έχουν προστεθεί νέα στοιχεία στο τέλος της λίστας των διαθέσιμων αρχείων μετά από κάποια εκτέλεση της εντολής upload. Τότε το σύστημα δεν χρειάζεται να ζητήσει εκ νέου τα καταχωρημένα δεδομένα για να τα προβάλει στις ιστοσελίδες. Συνεπώς, σε αυτές τις περιπτώσεις αποθηκεύουμε κάποια στοιχεία σε πίνακες Javascript στον κώδικα των ιστοσελίδων ώστε να επιτυγχάνουμε καλύτερη απόδοση και αποκρισιμότητα του συστήματος αλλά και για να αποφεύγουμε κόστος άσκοπης μεταφοράς δεδομένων.
- Αντίθετα όταν εκτελείται συνάρτηση updateURL, updatePrice ή delete, πρέπει να γίνει αλλαγή στα στοιχεία που προβάλλονται στις ιστοσελίδες και μόνο τότε το σύστημα χρειάζεται να ζητήσει εκ νέου την μεταφορά των στοιχείων από τα καταχωρημένα αρχεία στο blockchain.

4

Εργαλεία και τεχνολογίες

Στο κεφάλαιο αυτό παρουσιάζονται τα κυριότερα εργαλεία και τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής της παρούσας διπλωματικής εργασίας. Πιο συγκεκριμένα, παρουσιάζονται ορισμένα εργαλεία που προσφέρει το Ethereum για την ανάπτυξη, λειτουργία και πρόσβαση στις αποκεντρωμένες εφαρμογές (DApps) Αυτά είναι τα Geth, Ganache CLI, Ethereumjs, Web3, Solidity, Metamask. Στην συνέχεια, παρουσιάζεται το πρόγραμμα διαχείρισης πακέτων NPM. Ακολούθως, παρουσιάζεται η server-side πλατφόρμα Node.js, η οποία χρησιμοποιήθηκε για την επικοινωνία με την βάση δεδομένων MongoDB. Τέλος, παρουσιάζονται εργαλεία για την ανάπτυξη ιστοσελίδων και το σύστημα βάσεων δεδομένων MongoDB με την μέθοδο αποθήκευσης GridFS.

4.1 Περιγραφή του Ethereum Blockchain

Το οικοσύστημα του Ethereum παρέχει στους προγραμματιστές πολλά εργαλεία για την ανάπτυξη αποκεντρωμένων εφαρμογών (DApps), στην συνέχεια παρουσιάζονται μερικά από τα βασικότερα εργαλεία που χρησιμοποιήθηκαν για την εκπόνηση της παρούσας διπλωματικής εργασίας.

4.1.1 Διεπαφή για εκτέλεση Ethereum κόμβου: Geth

Το geth [41] είναι μία command line διεπαφή, υλοποιημένη στην γλώσσα προγραμματισμού Go, για την εκτέλεση του πλήρους Ethereum κόμβου. Με το geth μπορεί κάποιος να γίνει μέλος του Ethereum δικτύου και μεταξύ άλλων να:

- Εξορύξει (mine) ether.
- Μεταφέρει ether μεταξύ διευθύνσεων.
- Να δημιουργήσει συμβόλαια και να στείλει λοιπές συναλλαγές.
- Εξερευνήσει το blockchain του Ethereum.
- Δημιουργήσει ένα ιδιωτικό Ethereum blockchain.
- Πραγματοποιήσει πολλές άλλες λειτουργίες.

4.1.2 Εξομοιωτής δικτύου Blockchain: Ganache CLI

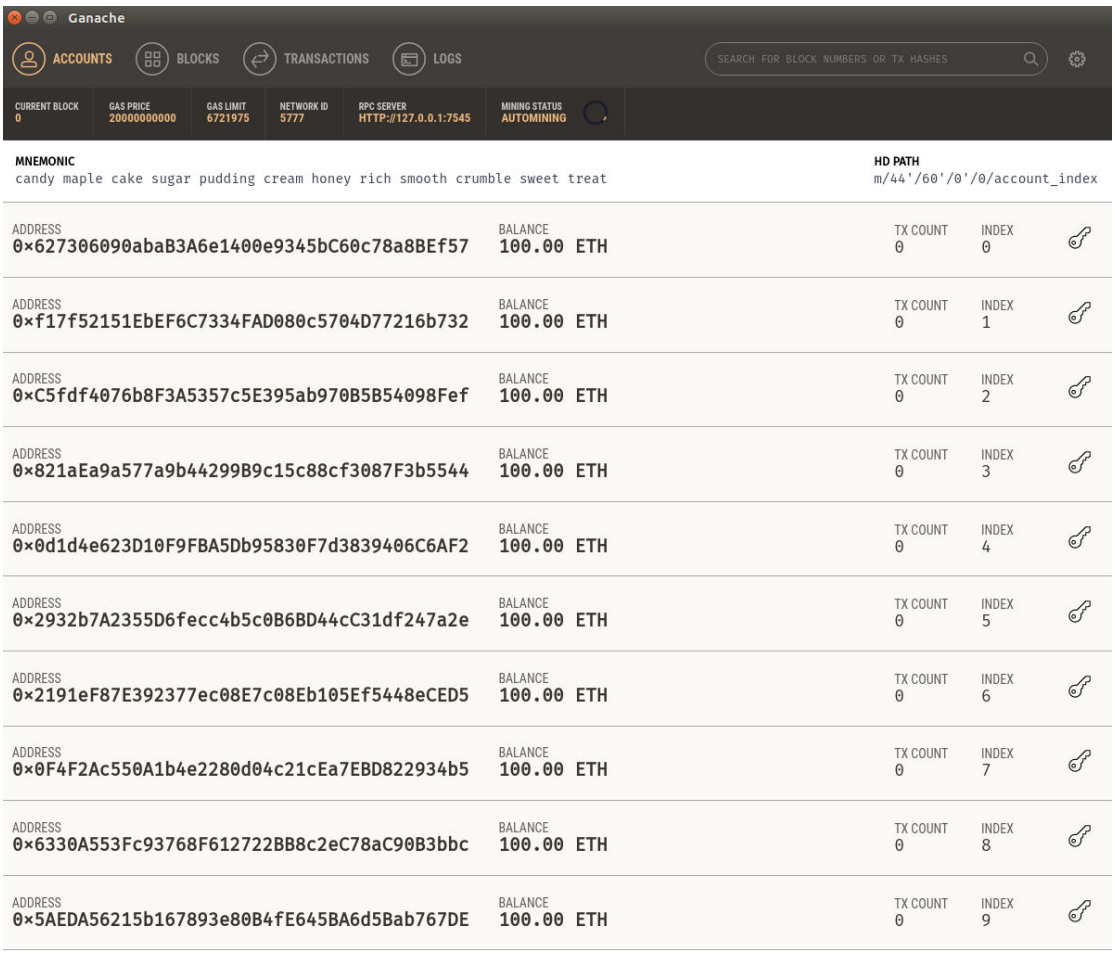
Έκδοση που χρησιμοποιήθηκε: 6.0.1

Το Ganache CLI [42] είναι ένα npm πακέτο. Αποτελεί έναν Ethereum client ειδικό για δοκιμή και ανάπτυξη (test&dev), ο οποίος βασίζεται στο Node.js.

Χρησιμοποιεί την συλλογή βιβλιοθηκών ethereumjs για να προσομοιώσει πλήρως έναν πραγματικό Ethereum κόμβο, χωρίς όμως την επιβάρυνση που συνεπάγεται η εκτέλεση του πραγματικού Ethereum κόμβου. Δημιουργεί δηλαδή ένα ιδιωτικό Ethereum blockchain για γρήγορη δοκιμή και ανάπτυξη έξυπνων συμβολαίων και αποκεντρωμένων εφαρμογών DApps. Επίσης, περιλαμβάνει όλες τις δημοφιλείς RPC (Remote Procedure Call) συναρτήσεις και δυνατότητες (π.χ. events) και μπορεί να εκτελεστεί αιτιοκρατικά, καθιστώντας έτσι την διαδικασία ανάπτυξης πολύ γρηγορότερη. Συνεπώς, το Ganache CLI είναι ένα εργαλείο που προσφέρει μεγάλη ευκολία και πολλές δυνατότητες στον προγραμματιστή αποκεντρωμένων εφαρμογών.

Από τα παραπάνω, φαίνεται η αναγκαιότητα ενός τέτοιου εργαλείου, ειδάλλως θα έπρεπε όλη η ανάπτυξη και οι δοκιμές να γίνουν στο πραγματικό Ethereum δίκτυο ή σε κάποιο από τα test networks. Η διαδικασία αυτή θα είχε ως συνέπεια τα εξής:

- Αποθήκευση περίσσειας πληροφορίας στο δημόσιο Ethereum δίκτυο
- Επιβάρυνση του δικτύου με μη παραγωγική κίνηση
- Μεγάλα διαστήματα αναμονής για τον προγραμματιστή
- Πολύ μεγαλύτερο κόστος ανάπτυξης, καθώς θα απαιτούσε την αγορά και χρήση πραγματικών ethers



Εικόνα 4-1 Εκτέλεση του ganache-cli σε λειτουργικό σύστημα Ubuntu

Όπως παρατηρούμε στην εικόνα, η εκτέλεση της εντολής ganache-cli δημιουργεί τοπικά ένα Ethereum blockchain, με 10 Ethereum λογαριασμούς/διευθύνσεις, που ακούει στην τοπική πόρτα 7545 (Listening on

localhost:7545). Σε αντίθεση με την προηγούμενη έκδοση του Ganache που ονομαζόταν testrpc, το Ganache δημιουργεί κάθε φορά τους ίδιους Ethereum λογαριασμούς/διευθύνσεις.

4.1.3 Javascript κοινότητα του Ethereum: EthereumJS

Το EthereumJS [43] είναι η JavaScript κοινότητα για το Ethereum. Αποτελεί την μεγαλύτερη ως τώρα συλλογή βιβλιοθηκών και εργαλείων για την αλληλεπίδραση με το δίκτυο Ethereum μέσω της γλώσσας JavaScript. Περιλαμβάνει μία μεγάλη λίστα από modules χρήσιμα στην ανάπτυξη εφαρμογών για το Ethereum.

4.1.4 Συλλογή βιβλιοθηκών Web3.js

Έκδοση που χρησιμοποίησα: 0.19.1

Ένα από τα πιο σημαντικά npm πακέτα που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής. Το πακέτο web3.js [44] είναι μία συλλογή βιβλιοθηκών που επιτρέπουν την αλληλεπίδραση με ένα τοπικό ή απομακρυσμένο κόμβο του Ethereum blockchain, χρησιμοποιώντας HTTP ή IPC σύνδεση. Με άλλα λόγια, είναι μία διεπαφή επικοινωνίας μεταξύ της JavaScript και του Ethereum blockchain («Ethereum compatible JavaScript API which implements the Generic JSON RPC specification»). Είναι δηλαδή ο πιο διαδεδομένος τρόπος για να αναφέρεται η JavaScript (server-side ή/και front-end) σε αντικείμενα που «ζουν» μέσα στο blockchain, όπως τα έξυπνα συμβόλαια (solidity smart contracts), τα δεδομένα τους, τις συναρτήσεις τους, τις διευθύνσεις και τα υπόλοιπα λογαριασμών, όπως και πολλά άλλα.

4.1.5 Γλώσσα προγραμματισμού έξυπνων συμβολαίων Solidity

Έκδοση που χρησιμοποιήθηκε: 0.4.18

Η Solidity [45] είναι μία «συμβολαιοστρεφής», υψηλού επιπέδου γλώσσα προγραμματισμού που εκτελείται στο Ethereum Virtual Machine (EVM). Χρησιμοποιείται κυρίως για την δημιουργία έξυπνων συμβολαίων (smart contracts) για το Ethereum blockchain. Ο ίδιος ο πηγαίος κώδικας της γλώσσας Ethereum είναι γραμμένος σε Solidity. Έχει παρόμοιο συντακτικό με αυτό της JavaScript, οπότε είναι εύκολα κατανοήσιμη από έναν πολύ μεγάλο αριθμό προγραμματιστών. Είναι μία στατικού τύπου γλώσσα. Υποστηρίζει την κληρονομικότητα με παρόμοιο τρόπο με άλλες γλώσσες προγραμματισμού (πχ. C++).

4.1.6 Διαδικτυακή πλατφόρμα έξυπνων συμβολαίων Remix

Το Remix [46], [47] είναι μία σουίτα εργαλείων για την αλληλεπίδραση με το Ethereum blockchain. Το Remix IDE είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης, προσβάσιμο από φυλλομετρητή για την ανάπτυξη έξυπνων συμβολαίων (Solidity smart contracts), την μεταγλώττιση (compile) και κατόπιν την δημιουργία (deploy) και εκτέλεσή (run) τους στο Ethereum blockchain. Πιο συγκεκριμένα μέσω του Remix πραγματοποιήθηκε το deployment των συμβολαίων στο ιδιωτικό Ethereum δίκτυο του Ganache test network.

Solc solidity compiler

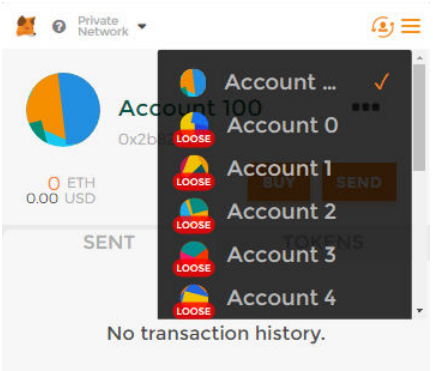
Τέλος, για να γίνει η ανάπτυξη των έξυπνων συμβολαίων γρηγορότερη έγινε χρήση του npm πακέτου solc. Αυτό περιλαμβάνει την προγραμματιστική γλώσσα στην οποία γράφονται όλα τα έξυπνα του Ethereum, Solidity με τον μεταγλωττιστή της. Είναι μία ανεξάρτητη βιβλιοθήκη της JavaScript, μέσω της οποίας μπορούμε και να μεταγλωττίσουμε κώδικα Solidity.

4.1.7 Επέκταση φυλλομετρητή MetaMask

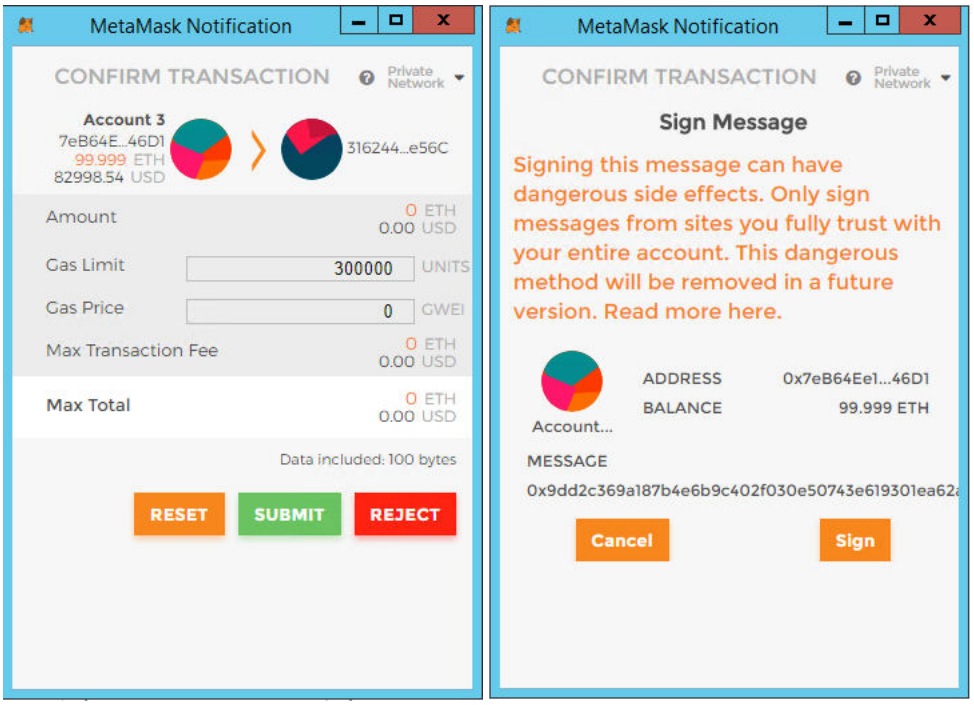
Ιστοσελίδα: https://MetaMask.io/

Το MetaMask [48] είναι ένα plugin διαθέσιμο για τους φυλλομετρητές Google Chrome, Mozilla Firefox, Brave και Opera. Το MetaMask είναι στην ουσία μία γέφυρα η οποία δίνει στον browser πρόσβαση στις κατανεμημένες εφαρμογές (DApps), που για την λειτουργία τους βασίζονται στο δίκτυο Ethereum. Το μεγάλο πλεονέκτημα που προσφέρει είναι το γεγονός ότι με την χρήση του, η πρόσβαση στις εφαρμογές αυτές, δεν απαιτεί την εκτέλεση του πλήρους Ethereum κόμβου στο μηχάνημα του χρήστη, όπως για παράδειγμα απαιτεί ο ειδικός για το Ethereum φυλλομετρητής Mist.

Συμπεριλαμβάνει μία ασφαλή κρύπτη και παρέχει στον χρήστη μία διεπαφή, μέσω της οποίας αυτός μπορεί να διαχειρίζεται τους Ethereum λογαριασμούς/διευθύνσεις του, ώστε να αλληλεπιδρά με τις ιστοσελίδες, να στέλνει ή και να υπογράφει συναλλαγές.



Εικόνα 4-2 Διαχείριση λογαριασμών στο MetaMask



Εικόνα 4-3 Αποστολή συναλλαγής στο MetaMask
Εικόνα 4-4 Υπογραφή δεδομένων στο MetaMask

Σύμφωνα με τους δημιουργούς του, ο σκοπός του είναι να κάνει το Ethereum προσβάσιμο σε όσο το δυνατόν περισσότερο κόσμο.

Είναι πολύ γρήγορη η εγκατάστασή του και η χρήση του ακόμα και από χρήστες χωρίς εμπειρία σε DApps. Από την άλλη η διαδικασία εγκατάστασης και εκτέλεσης του πλήρους Ethereum κόμβου απαιτεί τεχνικές γνώσεις, αρκετούς πόρους συστήματος (επεξεργαστή και μνήμη) και πολύ χρόνο.

4.2 Πρόγραμμα διαχείρισης πακέτων της Javascript: Node Package Manager

Το npm – Node Package Manager [49] είναι ένα πρόγραμμα διαχείρισης πακέτων της γλώσσας JavaScript και το προκαθορισμένο πρόγραμμα διαχείρισης πακέτων του προγραμματιστικού περιβάλλοντος Node.js. Εκτός από πακέτα, στο αρχείο του npm υπάρχουν και node modules τα οποία χρησιμοποιούνται στο server-side προγραμματισμό. Το πακέτο (package) είναι ένα αρχείο ή ένα directory το οποίο περιγράφεται από ένα package.json, ενώ το module είναι ένα αρχείο ή ένα directory το οποίο φορτώνεται από το Node.js μέσω του require().

Αποτελείται από τρία διακριτά κομμάτια:

- Την ιστοσελίδα (https://www.npmjs.com/).
- Ένα αρχείο (registry), στο οποίο είναι αποθηκευμένα τα JavaScript πακέτα και τα Node modules.
- Τον command line client (Command Line Interface) (που βρίσκεται στον υπολογιστή του χρήστη), μέσω του οποίου είτε λαμβάνονται τα διαθέσιμα πακέτα είτε δημοσιεύονται στο αρχείο πακέτα που ο χρήστης έχει δημιουργήσει.

Το NPM είναι πολύ χρήσιμο στην κοινότητα ανάπτυξης λογισμικού, διότι αφενός επιτρέπει την ανταλλαγή πακέτων, δηλαδή κώδικα που λύνει πολύ καλά ένα συγκεκριμένο πρόβλημα και αφετέρου έχει θεσπίσει προδιαγραφές τις οποίες ακολουθούν όλα τα πακέτα που δημοσιεύονται στο αρχείο, έτσι ώστε να υπάρχει ένας ενιαίος τρόπος χρησιμοποίησής τους από τους προγραμματιστές.

4.3 Πλατφόρμα ανάπτυξης λογισμικού Node.js

Έκδοση που χρησιμοποιήθηκε: v8.9.3

Το Node.js [50] είναι μία server-side πλατφόρμα, η οποία αναπτύχθηκε πάνω στο Google Chrome JavaScript Engine (V8 Engine) το 2009 και από τότε συνεχώς βελτιώνεται. Είναι ένα ανοιχτού κώδικα (open source), cross-platform περιβάλλον ανάπτυξης και εκτέλεσης της γλώσσας προγραμματισμού JavaScript, κατάλληλο για εύκολη και γρήγορη ανάπτυξη κλιμακώσιμων διαδικτυακών εφαρμογών. Χρησιμοποιεί ένα event-driven, non-blocking, I/O μοντέλο και πετυχαίνει υψηλή απόδοση χρησιμοποιώντας λίγους φυσικούς πόρους. Θεωρείται ιδανικό για εφαρμογές πραγματικού χρόνου, υπολογιστικά έντονες που τρέχουν σε κατανεμημένα περιβάλλοντα. Επίσης, το Node.js παρέχει μία μεγάλη βιβλιοθήκη από JavaScript modules, γεγονός που απλοποιεί ουσιωδώς την διαδικασία ανάπτυξης διαδικτυακών εφαρμογών. Δηλαδή το Node.js είναι και περιβάλλον εκτέλεσης, αλλά και βιβλιοθήκη της JavaScript.

Χαρακτηριστικά του Node.js [51]:

- Ασύγχρονο και οδηγούμενο από γεγονότα (Asynchronous and Event Driven) – Όλες οι προγραμματιστικές διεπαφές (API application programming interface) της Node.js βιβλιοθήκης είναι ασύγχρονες κάτι που σημαίνει ότι ο κώδικας δεν κολλάει ποτέ σε ένα σημείο περιμένοντας από μία σύνθετη (εξωτερική) διεργασία (API) να επιστρέψει δεδομένα, παρά προχωράει η εκτέλεση, και όταν η κληθείσα διεργασία ολοκληρωθεί, ο server, μέσω ενός ειδικού μηχανισμού ενημέρωσης συμβάντων (notification mechanism of Events) λαμβάνει την απάντηση/δεδομένα από αυτήν.
- Πολύ γρήγορο – Επειδή βασίζεται πάνω στην ιδεατή μηχανή Google Chrome V8 JavaScript Engine η βιβλιοθήκη Node.js είναι πολύ γρήγορη στην εκτέλεση κώδικα.
- Μονού νήματος, αλλά πολύ κλιμακώσιμο – Το Node.js χρησιμοποιεί μοντέλο μονού νήματος εκτέλεσης με event looping. Ο μηχανισμός αυτός των συμβάντων βοηθάει τον εξυπηρετητή να απαντάει με non-blocking τρόπο κάτι που του δίνει την ιδιότητα της κλιμακωσιμότητας, σε αντίθεση με άλλους παραδοσιακούς εξυπηρετητές που χρησιμοποιούν περιορισμένο αριθμό νημάτων για την εξυπηρέτηση αιτήσεων. Το πρόγραμμα ενός νήματος του Node.js μπορεί να εξυπηρετήσει πολύ μεγαλύτερο αριθμό αιτήσεων από παραδοσιακούς εξυπηρετητές, όπως είναι ο Apache HTTP Server.
- Δεν υπάρχει προσωρινή αποθήκευση δεδομένων – Οι εφαρμογές του Node.js δεν κάνουν προσωρινή αποθήκευση δεδομένων, αλλά στέλνουν τα δεδομένα σε μικρά κομμάτια.

Google V8 engine

Η ιδεατή μηχανή V8 της Google είναι μία ανοικτού κώδικα υψηλής απόδοσης μηχανή JavaScript [52]. Είναι γραμμένη σε C++ και JavaScript χρησιμοποιείται από τον ανοικτού κώδικα browser Google Chrome, το Node.js και σε πολλές άλλες εφαρμογές. Είναι υπεύθυνη για την εκτέλεση κώδικα που γραμμένος στην γλώσσα JavaScript. Υλοποιεί το ECMAScript και τρέχει σε Windows 7 ή μεταγενέστερα, macOS 10.5+, και Linux συστήματα που χρησιμοποιούν IA-32, ARM, ή MIPS επεξεργαστές.

4.4 Ανάπτυξη ιστοσελίδων

Σε αυτήν την ενότητα θα παρουσιαστούν οι τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη των ιστοσελίδων, μέσω των οποίων οι χρήστες χρησιμοποιούν την εφαρμογή. Φυσικά χρησιμοποιήθηκε η γλώσσα HTML (Hypertext Markup Language), η οποία είναι η γλώσσα που χρησιμοποιείται σε ολόκληρο τον κόσμο για την δημιουργία ιστοσελίδων και διαδικτυακών εφαρμογών. Η HTML μαζί με τα Cascading Style Sheets (CSS) και την JavaScript είναι ίσως οι τρεις πιο σημαντικές τεχνολογίες για τον παγκόσμιο ιστό.

4.4.1 Εργαλείο ανάπτυξης ιστοσελίδων Bootstrap

Το Bootstrap [53] είναι ένα ελεύθερο, ανοιχτού κώδικα πλαίσιο ανάπτυξης ιστοσελίδων και διαδικτυακών εφαρμογών (front-end web framework), δηλαδή μία συλλογή εργαλείων. Παράλληλα, είναι ένας συνδυασμός από HTML, CSS, και JavaScript κώδικα, σχεδιασμένος να συμβάλλει στην αποτελεσματικότερη δημιουργία διεπαφών χρήστη. Κάνει την διαδικασία σχεδίασης και ανάπτυξης ιστοσελίδων πιο εύκολη και πιο γρήγορη, βοηθώντας παράλληλα τον προγραμματιστή να πετύχει καλύτερο αποτέλεσμα, με πιο όμορφες και χρηστικές διεπαφές. Επίσης, με τα CSS που παρέχει, συμβάλλει στην δημιουργία προσαρμοστικών ιστοσελίδων· δηλαδή ιστοσελίδων οι οποίες δυναμικά αλλάζουν τον τρόπο παρουσίασής περιεχομένου, ανάλογα το μέγεθος της οθόνης στην οποία προβάλλονται, πράγμα απαραίτητο, αφού όλο και περισσότεροι χρήστες χρησιμοποιούν κινητά και tablet και όχι μόνο υπολογιστές για την περιήγησή τους στο διαδίκτυο.

Το Bootstrap είναι ένα από τα πιο διαδεδομένα front-end frameworks. Ορισμένοι λόγοι γι' αυτό είναι [54]:

- Ευκολία στην χρήση. Οποιοσδήποτε με στοιχειώδεις γνώσεις HTML και CSS μπορεί να το χρησιμοποιήσει.
- Προσαρμοστικότητα στην παρουσίαση. Τα CSS του Bootstrap προσαρμόζονται σε κινητά, tablets, και υπολογιστές.
- Συμβατότητα με φυλλομετρητές. Υποστηρίζει όλους τους ευρέως χρησιμοποιούμενους φυλλομετρητές, όπως Google Chrome, Firefox, Microsoft Edge, Internet Explorer, Opera, Safari και άλλους.
- Πολύ καλό σύστημα πλέγματος. Όταν χρησιμοποιείται το Bootstrap τα πάντα μπορούν να οργανωθούν σε στήλες.

- Παρέχεται βασικό στυλ για τα περισσότερα HTML στοιχεία (Typography, Code, Tables, Forms, Buttons, Images, Icons).
- JavaScript components με την μορφή jQuery plugins, τα οποία παρέχουν επιπλέον στοιχεία για την διεπαφή του χρήστη, όπως dialog boxes, tooltips, carousels. Επίσης προσθέτουν επιπλέον λειτουργικότητα σε ήδη υπάρχοντα στοιχεία. Ενδεικτικά μερικά JavaScript plugins που υποστηρίζονται: Dropdown, Scrollspy, Tab, Tooltip, Popover, Alert, Button, Collapse, Carousel και Typehead.

4.4.2 Γλώσσα προγραμματισμού JavaScript

JavaScript [55] χρησιμοποιήθηκε και για το front-end (κώδικας που εκτελείται στον φυλλομετρητή του πελάτη – client-side JavaScript). Η JavaScript είναι διερμηνευμένη γλώσσα προγραμματισμού και μία από τις πιο διαδεδομένες, αφού χρησιμοποιείται κατά κόρον σε διαδικτυακές εφαρμογές (και όχι μόνο) και υποστηρίζεται από όλους τους μοντέρνους φυλλομετρητές. Είναι ο καλύτερος τρόπος να δώσουμε δυναμικό περιεχόμενο σε μία ιστοσελίδα κάνοντάς την να επιτελεί σύνθετες λειτουργίες. Μπορεί επίσης να χρησιμοποιηθεί και για την ανάπτυξη προγραμμάτων που δεν εκτελούνται σε φυλλομετρητές, όπως για παράδειγμα σε εξυπηρετητές, βάσεις δεδομένων, επεξεργαστές PDF εγγράφων, Desktop εφαρμογές, αλλά και εφαρμογές κινητών.

Για την εκτέλεσή της είναι απαραίτητη κάποια μηχανή JavaScript, όπως είναι για παράδειγμα η Google V8. Η μηχανή JavaScript είναι στην ουσία ένα πρόγραμμα ή διερμηνέας που εκτελεί κώδικα γραμμένο στην γλώσσα JavaScript. Συναντάται κυρίως στους φυλλομετρητές, όμως χρησιμοποιείται και από άλλες πλατφόρμες, όπως είναι το Node.js.

Η JavaScript είναι μία υψηλού επιπέδου, δυναμική, ασθενούς τύπου, prototype-based, πολυ-παραδειγματική, διερμηνευμένη (interpreted) γλώσσα προγραμματισμού. Ως πολυ-παραδειγματική γλώσσα, η JavaScript υποστηρίζει τα event-driven, συναρτησιακά και προστακτικά (συμπεριλαμβανομένων των αντικειμενοστρεφών prototype-based) προγραμματιστικά στυλ. Έχει έτοιμες προγραμματιστικές διεπαφές (APIs) για την επεξεργασία κειμένου, πινάκων, ημερομηνιών, καθώς και τον χειρισμό DOM (Document Object Model), όμως δεν υποστηρίζει λειτουργίες εισόδου εξόδου (I/O), όπως δικτύωση, αποθήκευση, γραφικά, παρά βασίζει την υλοποίηση των λειτουργιών αυτών στο περιβάλλον εκτέλεσής της.

Μερικά από τα πλεονεκτήματα της χρήσης JavaScript στις διαδικτυακές εφαρμογές είναι [56]:

- Λιγότερη αλληλεπίδραση μεταξύ πελάτη – εξυπηρετητή. Προσφέρεται η δυνατότητα προεπεξεργασίας των δεδομένων τοπικά, στον φυλλομετρητή του πελάτη πριν αυτά σταλούν στον εξυπηρετητή. Μειώνεται έτσι ο φόρτος του εξυπηρετητή, αλλά και ολόκληρου του δικτύου.
- Άμεση ανατροφοδότηση στον χρήστη. Υπάρχει η δυνατότητα εμφάνισης μηνύματα στον χρήστη που τον ενημερώνουν σχετικά με την κατάσταση διεργασιών που απαιτούν πολύ χρόνο κτλ.
- Αυξημένη διαδραστικότητα ιστοσελίδας. Μπορούν να δημιουργηθούν διεπαφές, οι οποίες ανταποκρίνονται στις κινήσεις του χρήστη, χωρίς να χρειάζεται να ανανεωθεί ολόκληρη η ιστοσελίδα.

- Πλουσιότερες διεπαφές. Η JavaScript εμπλουτίζει την HTML και τα CSS, κάνοντας έτσι διαθέσιμα στον χρήστη πολλά νέα στοιχεία.

4.4.3 Βιβλιοθήκη jQuery της Javascript

Η jQuery [57] είναι μία γρήγορη, συμπαγής, με πλούσια χαρακτηριστικά JavaScript βιβλιοθήκη, σχεδιασμένη να απλοποιήσει την συγγραφή JavaScript κώδικα για το front-end. Είναι δωρεάν και ανοιχτού κώδικα και αποτελεί την πιο ευρέως χρησιμοποιούμενη JavaScript βιβλιοθήκη στο διαδίκτυο. Η αρθρωτή φύση της καθώς και η εύκολη στην χρήση προγραμματιστική διεπαφή την καθιστούν ιδανική για την δημιουργία πλούσιων ιστοσελίδων και διαδικτυακών εφαρμογών.

Πιο συγκεκριμένα παρέχονται τα παρακάτω χαρακτηριστικά [58]:

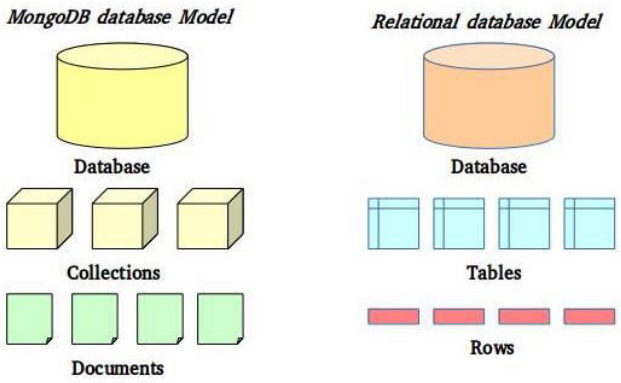
- Εύκολος χειρισμός HTML/DOM, CSS και συμβάντων (events)
- Εφέ και κινούμενα σχέδια
- Απλοποίηση των AJAX (Asynchronous JavaScript And XML) κλήσεων
- Συναρτήσεις για την διευκόλυνση και τυποποίηση της υλοποίηση συνηθισμένων διεργασιών

Επιπλέον λειτουργικότητες προσφέρονται μέσω των jQuery plugins. Καλύπτεται έτσι από την jQuery ένα πολύ μεγάλο φάσμα λειτουργιών. Τέτοιου είδους πακέτα μπορούν να βρεθούν έτοιμα, από την κοινότητα ελεύθερου λογισμικού ή και να δημιουργηθούν από οποιονδήποτε και να μοιραστούν και σε άλλους χρήστες.

4.5 Βάσεις Δεδομένων

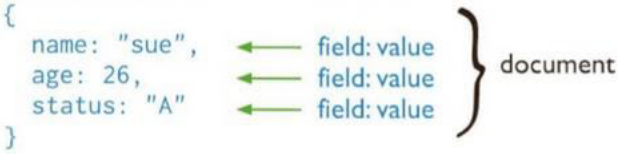
4.5.1 Βάση Δεδομένων MongoDB

Για την εφαρμογή χρησιμοποιήθηκε η βάση δεδομένων MongoDB [59]. Η MongoDB αποτελεί μια ανοιχτού κώδικα εγγραφο-κεντρική (document-based) βάση δεδομένων που χρησιμοποιείται σε διαφορετικά λειτουργικά συστήματα και ανήκει στην κατηγορία των Μη-σχεσιακών [60] βάσεων δεδομένων (No-SQL), όπως είδαμε προηγουμένως. Αποθηκεύει τα δεδομένα σε binary μορφή JSON εγγράφων (JavaScript Object Notation documents) που ονομάζονται BSON (Binary JSON). Συνήθως τα έγγραφα με παρόμοια μορφή οργανώνονται σε συλλογές (collections). Αν θέλαμε να κάνουμε μια αντιστοίχιση με τις σχεσιακές βάσεις θα λέγαμε ότι οι συλλογές είναι οι πίνακες (tables) και τα έγγραφα είναι οι εγγραφές (records) που στην περίπτωση των RDBMS αυτές οι εγγραφές βρίσκονται διασκορπισμένες σε διαφορετικούς πίνακες, ενώ εδώ τα έγγραφα είναι οργανωμένα σε μία συλλογή.



Εικόνα 4-5 MongoDB και RDBMS Models

Ας δούμε τώρα ένα έγγραφο BSON της MongoDB:



Εικόνα 4-6 BSON MongoDB document

Παρατηρούμε ότι η MongoDB δομεί τα δεδομένα της σε ζεύγη τύπου "πεδίο:τιμή" (field:value) και ομαδοποιώντας τέτοια ζεύγη ορίζει ένα BSON document. Τα έγγραφα στη MongoDB αποθηκεύονται σε συλλογές (collections), όπως είδαμε, και προσφέρουν κι άλλες χρήσιμες ιδιότητες. Οι τιμές των πεδίων των BSON αντικειμένων μπορούν να πάρουν ως τιμές άλλα έγγραφα, πίνακες ακόμα και πίνακες από έγγραφα και το δυναμικό τους σχήμα καθιστά δυνατό τον πολυμορφισμό (documents με διαφορετικές δομές στην ίδια συλλογή). Παρακάτω παρουσιάζονται τα βασικά χαρακτηριστικά που προσφέρει η βάση MongoDB.

Σημαντικά χαρακτηριστικά της βάσης MongoDB:

Η MongoDB προσφέρει τα πλεονεκτήματα των No-SQL βάσεων μαζί με αυτά των document-based No-SQL βάσεων. Κάποιες βασικές ιδιότητες και δυνατότητες της αρχιτεκτονικής της είναι η οριζόντια κλιμάκωση (horizontal scaling), η υψηλή διαθεσιμότητα (high availability) και η υψηλή διεκπεραιωτική ικανότητα (throughput), τα οποία προσφέρονται αντίστοιχα από τον θρυμματισμό (sharding), τα αντίγραφα που δημιουργεί (replica sets) αλλά και την κατανομή του φόρτου εργασίας σε όλο το σύμπλεγμα (cluster load balancing). Παρακάτω αναφέρονται τα τρία διαφορετικά είδη κόμβων της MongoDB, αφού προηγηθεί η εξήγηση του sharding και των replica sets:

Sharding: Με τη μέθοδο αυτή, η MongoDB χωρίζει (θρυμματίζει) τα δεδομένα σε επίπεδο συλλογής χρησιμοποιώντας ένα κλειδί ονομαζόμενο "shard key". Τα κομμάτια στα οποία η MongoDB χωρίζει τα δεδομένα με βάση το shard key λέγονται "chunks". Μετά τον διαχωρισμό, τα διανέμει ανάμεσα στους "shard" κόμβους του cluster με τη βοήθεια του balancer. Ο balancer ελέγχει αν τηρείται εξισορρόπηση των chunks μιας συλλογής ανάμεσα στους shard κόμβους, και σε διαφορετική περίπτωση

μετακινεί ένα chunk κάθε φορά ανάμεσα στους κατάλληλους shard κόμβους ώστε να πετύχει εξισορρόπηση. Ακόμη, είναι σημαντικό να αναφέρουμε τους τρόπους και τους περιορισμούς επιλογής shard key. Το shard key μπορεί να αποτελείται από ένα πεδίο (single index: "{name:1}") ή περισσότερα πεδία (compound index: "{name:1, price:1}"). Κάθε έγγραφο (document) της συλλογής που θρυμματίζεται (sharded collection) πρέπει να περιέχει το shard key. Οι τιμές του shard key διανέμονται στα κομμάτια (chunks) με μεθόδους όπως "Hashed sharding" (χρησιμοποιώντας hashed index σε ένα πεδίο του εγγράφου ως shard key), "Ranged sharding" (διαιρώντας με βάση το εύρος τιμών) ή "Tag Aware sharding" (ονομάζοντας το κάθε shard με ένα tag και διαχειρίζοντας τα shards πλέον μέσω των tags). Είναι φανερό ότι ανάλογα με το πρόβλημα που έχουμε κάθε φορά, επιλέγουμε και την μέθοδο που μας εξυπηρετεί καλύτερα.

Replica Sets (διατήρηση εφεδρικών αντιγράφων): Με τη μέθοδο αυτή, η MongoDB προσφέρει υψηλή διαθεσιμότητα (high availability) και ανοχή σφαλμάτων (fault tolerance) καθώς διατηρεί αντίγραφα της ίδιας πληροφορίας σε διαφορετικούς κόμβους. Τα replica sets είναι ομάδες από mongod δαίμονες (διεργασίες) που αποτελούνται από έναν πρωτεύων κόμβο (primary node), ο οποίος λαμβάνει όλες τις write ενέργειες, και από έναν ή περισσότερους δευτερεύοντες κόμβους (secondary nodes), που ο καθένας τους αναπαράγει με την ίδια σειρά τις ενέργειες του πρωτεύοντος για να διατηρεί ταυτόσημα δεδομένα με αυτόν, αλλά χωρίς να επηρεάζει την λειτουργία του. Κάθε εφαρμογή μπορεί να γράφει μόνο στον πρωτεύοντα, αλλά έχει την επιλογή να διαβάζει είτε από αυτόν είτε από κάποιον δευτερεύοντα. Ακόμη, σε περίπτωση που ο πρωτεύων γίνει για οποιοδήποτε λόγο μη διαθέσιμος, γίνεται εκλογή νέου πρωτεύοντα από τους δευτερεύοντες. Όταν ο προηγούμενος πρωτεύων γίνει και πάλι διαθέσιμος, συγχρονίζεται με τον νέο πρωτεύοντα και συνεχίζει τη λειτουργία του πια ως δευτερεύων.

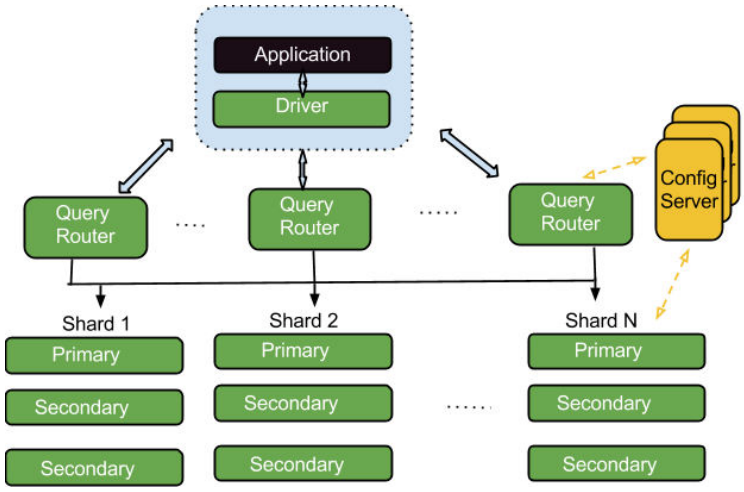
Παρακάτω συνεχίζουμε με την ανάλυση των κόμβων ως προς το είδος τους. Τα είδη είναι τρία:

Shard: ένας shard κόμβος αποθηκεύει ένα κλάσμα των δεδομένων του συμπλέγματος (cluster). Η πληροφορία όλων των shards αποτελεί το σύνολο των δεδομένων. Όπως φαίνεται στην παρακάτω εικόνα, ένα shard αντιστοιχεί σε ένα replica set που διατηρεί αντίγραφα των δεδομένων του shard.

Configuration Servers: αποθηκεύουν τα metadata ενός sharded cluster. Αυτά τα metadata περιλαμβάνουν την κατάσταση και την οργάνωση όλων των δεδομένων του cluster. Για παράδειγμα, τη λίστα των chunks που ανήκουν σε κάθε shard, τα εύρη τιμών που ορίζουν το κάθε chunk, κ.ά. Κάθε sharded cluster έχει τους δικούς του config servers και δεν τους μοιράζεται με άλλα clusters. Η βάση γράφει στους config servers όταν τα metadata αλλάζουν, για παράδειγμα, μετά από chunk migration ή chunk split, και διαβάζει από τους config servers όταν ξεκινάει κάποιο νέο mongos στιγμιότυπο ή γίνει αλλαγή στα metadata όπως αναφέραμε προηγουμένως.

Query Router: Σε ένα sharded cluster, είναι τα λεγόμενα "mongos" στιγμιότυπα (instances) που δρομολογούν τα queries και writes της εκάστοτε

εφαρμογής στα shards. Για λόγους κυρίως ασφάλειας και απόδοσης, δεν επιτρέπεται στις εφαρμογές να έχουν άμεση επικοινωνία με τα shards. Ο κάθε query router ή mongos μαθαίνει ποια δεδομένα βρίσκονται σε ποια shards φέρνοντας στην cache τα metadata των configuration servers. Με αυτά τα metadata, δρομολογεί τα αιτήματα (queries, writes) των εφαρμογών στα κατάλληλα mongod στιγμιότυπα (shards). Με περισσότερους query routers έχουμε και καλύτερη ανταπόκριση του cluster στα αιτήματα της εφαρμογής.



Εικόνα 4-7 Αρχιτεκτονική MongoDB: οι query routers δρομολογούν τα αιτήματα στα κατάλληλα shards, κάθε shard είναι ένα replica set με primary και secondary κόμβους, κάθε κόμβος "τρέχει" έναν mongod δαίμονα

4.5.2 Προδιαγραφή GridFS της MongoDB

Το GridFS [61] είναι ένα specification της MongoDB για την αποθήκευση και την ανάκτηση μεγάλων αρχείων όπως εικόνες, αρχεία ήχου, αρχεία βίντεο κ.λπ. Είναι ένα είδος συστήματος αρχείων για την αποθήκευση αρχείων, αλλά τα δεδομένα αποθηκεύονται σε συλλογές (collections) της MongoDB. Το GridFS έχει τη δυνατότητα να αποθηκεύει αρχεία ακόμη μεγαλύτερα από το όριο μεγέθους εγγράφων που ανεβαίνουν στην MongoDB και είναι 16MB.

Το GridFS διαιρεί ένα αρχείο σε κομμάτια και αποθηκεύει κάθε κομμάτι δεδομένων σε ξεχωριστό έγγραφο. Το κάθε κομμάτι έχει μέγιστο μέγεθος 255kB.

Το GridFS χρησιμοποιεί από προεπιλογή δύο συλλογές, τις fs.files και fs.chunks για να αποθηκεύσει τα μεταδεδομένα του αρχείου και τα κομμάτια. Κάθε κομμάτι αναγνωρίζεται από το μοναδικό πεδίο _id ObjectId. Τα fs.files χρησιμεύουν ως γονικό έγγραφο. Το πεδίο files_id στο έγγραφο fs.chunks συνδέει το κάθε κομμάτι με τον γονέα του.

Χάρη στο GridFS, πετυχαίνουμε υψηλές ταχύτητες στο ανέβασμα και κατέβασμα των αρχείων που θα ανεβάσουμε στην βάση MongoDB, ανεξαρτήτως του μεγέθους τους.

5

Σχεδιασμός και υλοποίηση Συστήματος

Στο κεφάλαιο αυτό θα παρουσιαστούν λεπτομέρειες που αφορούν τον σχεδιασμό και την υλοποίηση της αποκεντρωμένης εφαρμογής που αναπτύχθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας.

5.1 Έξυπνα συμβόλαια

Τα έξυπνα συμβόλαια αποτελούν το κυριότερο συστατικό στοιχείο κάθε αποκεντρωμένης εφαρμογής. Περιέχουν τον κώδικα που εκτελείται στο Ethereum VM και συνεπώς, αποτελούν τον βασικότερο ιστό της εφαρμογής.

Μία ιδιαιτερότητά τους είναι πως ο κώδικάς τους είναι οριστικός και δεν μπορεί να αλλάξει. Άπαξ και υλοποιηθεί κάποιο συμβόλαιο στο blockchain, τότε δεν υπάρχει τρόπος να αλλάξουμε την συμπεριφορά του, ούτε να διορθώσουμε κάποιο πιθανό σφάλμα, μιας και, όπως έχει ήδη αναφερθεί, τα περιεχόμενα του blockchain δεν μπορούν να μεταβληθούν. Στην περίπτωση που, κατά την διάρκεια λειτουργίας της εφαρμογής διαπιστώσουμε κάποιο κρίσιμο σφάλμα στα συμβόλαια αυτής, ο μόνος τρόπος διόρθωσής του θα ήταν η υλοποίηση εξαρχής κάποιου νέου συμβολαίου. Δόθηκε λοιπόν εξαιρετική προσοχή στο στάδιο ανάπτυξης των συμβολαίων λόγω της σπουδαιότητάς τους για την ορθή λειτουργία της εφαρμογής.

Για την εφαρμογή δημιουργήθηκαν δύο έξυπνα συμβόλαια το NtuaToken και το Uploader. Αυτά υλοποιούν όλες τις λειτουργικές απαιτήσεις που τέθηκαν στο προηγούμενο κεφάλαιο.

Το έξυπνο συμβόλαιο NtuaToken δημιουργεί το κρυπτονόμισμα NtuaToken (National Technical University of Athens Token). Το νόμισμα αυτό χρησιμοποιείται από την εφαρμογή σαν μια εναλλακτική των Ethers (κρυπτονόμισμα του Ethereum) για τις οικονομικές συναλλαγές μεταξύ των χρηστών που πωλούν τα αρχεία πολυμέσων και αυτών που τα αγοράζουν.

Το έξυπνο συμβόλαιο Uploader υλοποιεί όλη την λειτουργικότητα που αφορά την λειτουργία της εφαρμογής.

5.1.1 Αναλυτική παρουσίαση συμβολαίου NtuaToken

Στην συνέχεια γίνεται **αναλυτική παρουσίαση** κάθε σημείου του συμβολαίου **NtuaToken**.

```
pragma solidity ^0.4.18;
```

Η έκδοση της Solidity που χρησιμοποιείται.

```
event Transfer(address indexed _from, address indexed _to, uint256 _value);
```

Ένα event που ονομάζεται Transfer. Χρησιμοποιείται για την ενημέρωση των κόμβων του δικτύου για την πραγματοποίηση της μεταφοράς value NTUA Tokens από την διεύθυνση from προς την διεύθυνση to.

```
//This creates an array with all balances
mapping(address => uint256) public balanceOf;
```

Δημιουργία μιας δομής mapping που προσομοιάζει πίνακα στην Solidity και χρησιμοποιήθηκε για να αποθηκεύσουμε τα υπόλοιπα (balances) των Ethereum λογαριασμών.

```
//Public variables of the NtuaToken
string public name;
string public symbol;
uint8 public decimals;
uint256 public totalSupply;
```

Ορισμός των public μεταβλητών του συμβολαίου.

```
//constructor function
//Initializes contract with initial Supply tokens to each account (this
balance of NtuaTokens will have already been bought from the users)

function NtuaToken() public{
    balanceOf[0x627306090abaB3A6e1400e9345bC60c78a8BEf57] = 1000 * 1 ether;
    balanceOf[0xf17f52151EbEF6C7334FAD080c5704D77216b732] = 1000 * 1 ether;
    balanceOf[0xC5fdf4076b8F3A5357c5E395ab970B5B54098Fef] = 1000 * 1 ether;
    balanceOf[0x821aEa9a577a9b44299B9c15c88cf3087F3b5544] = 1000 * 1 ether;
    balanceOf[0x0d1d4e623D10F9FBA5Db95830F7d3839406C6AF2] = 1000 * 1 ether;
    balanceOf[0x2932b7A2355D6fecc4b5c0B6BD44cC31df247a2e] = 1000 * 1 ether;
    balanceOf[0x2191eF87E392377ec08E7c08Eb105Ef5448eCED5] = 1000 * 1 ether;
    balanceOf[0x0F4F2Ac550A1b4e2280d04c21cEa7EBD822934b5] = 1000 * 1 ether;
    balanceOf[0x6330A553Fc93768F612722BB8c2eC78aC90B3bbc] = 1000 * 1 ether;
    balanceOf[0x5AEDA56215b167893e80B4fE645BA6d5Bab767DE] = 1000 * 1 ether;
    name = "NtuaToken";
    symbol = "NtuaTok";
    decimals = 18;
}
```

Ο κατασκευαστής (constructor) του συμβολαίου NtuaToken εκτελείται αυτόματα μόλις το συμβόλαιο ενεργοποιηθεί στο blockchain. Μεταφέρει σε όλους τους λογαριασμούς 1000 NtuaTokens, τα οποία θα έχουν αγοραστεί προηγουμένως από τους 10 χρήστες της εφαρμογής. Το Ganache χρησιμοποιεί σε κάθε ενεργοποίησή του τις ίδιες 10 διευθύνσεις. Επομένως, αυτό μας επιτρέπει να τις χρησιμοποιήσουμε στην αρχικοποίηση των υπολοίπων σε NtuaTokens.

```
function balanceof(address _owner) external constant returns (uint256
balance){
     return balanceOf[_owner];
}
```

Συνάρτηση για την εμφάνιση των υπολοίπων σε NtuaToken όλων των λογαριασμών.

```
function _transfer(address _from, address _to, uint _value) internal {
     require(_to != 0x0);    //Prevent transfer to address 0x0
     require(balanceOf[_from] >= _value && balanceOf[_to] + _value >=
balanceOf[_to]);
     //Check if the sender has enough tokens - Check for overflow
     balanceOf[_from] -= _value; //Subtract from the sender
     balanceOf[_to] += _value; //Add the same to the recipient
     emit Transfer(_from,_to,_value);
}
```

Η συνάρτηση αυτή είναι internal, που σημαίνει ότι μπορεί να κληθεί μόνο εσωτερικά του συμβολαίου, δηλαδή μόνο από κάποια άλλη συνάρτηση του ίδιου συμβολαίου. Χρησιμοποιείται για να γίνει η μεταφορά NtuaTokens από την διεύθυνση from προς την διεύθυνση to εάν φυσικά το υπόλοιπο του αποστολέα επαρκεί. Κάθε φορά που η συναλλαγή επιτυγχάνει εκπέμπεται ένα event Transfer.

```
function transfer(address _to, uint256 _value) external{
     _transfer(msg.sender,_to,_value); //Contract calls the internal method
}
```

Η εξωτερική (μπορεί να κληθεί και από άλλα έξυπνα συμβόλαια ή μέσω web3 από Javascript κώδικα) συνάρτηση για την μεταφορά NtuaTokens μεταξύ λογαριασμών. Η συνάρτηση αυτή καλεί την εσωτερική συνάρτηση _transfer με παραμέτρους τα msg.sender (διεύθυνση του λογαριασμού που ζήτησε την μεταφορά), _to (λογαριασμός παραλήπτη), _value (ποσό NtuaTokens).

```
function transferFrom(address _from, address _to, uint256 _value) external
returns (bool success){
     _transfer(_from,_to,_value);
     return true;
}
```

Η εξωτερική (μπορεί να κληθεί και από άλλα έξυπνα συμβόλαια ή μέσω web3 από Javascript κώδικα) συνάρτηση για την μεταφορά NtuaTokens μεταξύ λογαριασμών. Αντίθετα με την προηγούμενη συνάρτηση, σε αυτή προσδιορίζουμε κατά την κλήση της, πέρα από τον παραλήπτη και το ποσό NtuaTokens, και τον αποστολέα.

5.1.2 Αναλυτική παρουσίαση συμβολαίου Uploader

Στην συνέχεια γίνεται **αναλυτική παρουσίαση** κάθε συνάρτησης/πεδίου του συμβολαίου **Uploader**.

```
pragma solidity ^0.4.18;
```

Η έκδοση της Solidity που χρησιμοποιείται.

```
struct media_ownership {
       address owner1;
       uint percentage1;
       address owner2;
       uint percentage2;
}
```

```
struct media_content {
     string mediaTitle;
     string artist;
     uint year;
     string url;
     uint price;
}
```

Στο συμβόλαιο αυτό, δημιουργήσαμε δύο structs για την αποθήκευση των δεδομένων από τα αρχεία πολυμέσων στο δίκτυο του blockchain.

Στο struct media_ownership αποθηκεύουμε τις Ethereum διευθύνσεις των κατόχων πνευματικών δικαιωμάτων του εκάστοτε αρχείου μαζί με την αποζημίωση που αιτούνται από κάθε πώληση του αρχείου.

Όσον αφορά το struct media_content, εκεί αποθηκεύουμε τα στοιχεία που αφορούν στο ίδιο το αρχείο, όπως είναι ο τίτλος του, το όνομα του καλλιτέχνη-δημιουργού, το έτος δημιουργίας του, η διεύθυνση στην οποία είναι αποθηκευμένο στη βάση δεδομένων και την τελική τιμή στην οποία διατίθεται το αρχείο για πώληση.

```
//mapping for all media (mediaID => media_content)
mapping(uint => media_ownership[]) mediaOwnershipTable;
//mediaOwnershipTable[int] array
```

```
//mapping for all media (mediaID => media_content)
mapping(uint => media_content[]) mediaContentTable;
//mediaContentTable[int] array
```

Δημιουργήσαμε δύο δομές mapping, μία για κάθε struct με στόχο να αποθηκεύσουμε ξεχωριστά τα στοιχεία που αφορούν τους ιδιοκτήτες των αρχείων με αυτά που αφορούν τα ίδια τα αρχεία πολυμέσων. Ως index έχουμε βάλει τον αύξοντα αριθμό id που είναι μοναδικός για κάθε αρχείο.

```
//constructor function
function uploader() public {
	idStable = 1;
	//initialization of auto-incremented id used for uploading media
files
}
```

Ο κατασκευαστής (constructor) του συμβολαίου Uploader εκτελείται αυτόματα μόλις το συμβόλαιο ενεργοποιηθεί στο blockchain. Η λειτουργία που κάνει είναι να αρχικοποιεί την μεταβλητή idStable, η οποία αποτελεί τον αύξοντα αριθμό που δίνεται σε κάθε νέο αρχείο το οποίο καταχωρείται στο δίκτυο του Blockchain. Ο αριθμός αυτός είναι μοναδικός για κάθε αρχείο, πράγμα που επιτυγχάνουμε αυξάνοντάς τον κάθε φορά που ολοκληρώνεται μια καταχώρηση αρχείου στο δίκτυο (στο τέλος της συνάρτησης createMedia).

```
//Function for the creator to upload the media content in the Ethereum
Blockchain.
function createMedia(address o1, uint p1, address o2, uint p2, string url,
uint price, string mediaTitle, string artist, uint year) external {
	mediaOwnershipTable[idStable].push(media_ownership(o1, p1, o2, p2));
	mediaContentTable[idStable].push(media_content(mediaTitle, artist,
year, url, price));
	//use of push function to insert new rows into mappings
	emit MediaCreated(mediaTitle, artist, year, price, idStable);
	idStable++;
	//increment id after the insertion of a media file
}
```

Η συνάρτηση createMedia λαμβάνει ως ορίσματα τα δεδομένα που εισήγαγε ο χρήστης – ιδιοκτήτης στην ιστοσελίδα Upload Media File. Αυτά είναι οι διευθύνσεις των χρηστών που κατέχουν τα πνευματικά δικαιώματα του αρχείου, τα ποσά που ζητούν να λάβουν για κάθε πώληση του αρχείου, η τελική τιμή του αρχείου, η διεύθυνση στην οποία είναι αποθηκευμένο το αρχείο, ο τίτλος του αρχείου, ο καλλιτέχνης και το έτος δημιουργίας. Έπειτα, δημιουργεί νέες γραμμές στους πίνακες mediaOwnershipTable και mediaContentTable με index το μοναδικό idStable. Στην συνέχεια, εκπέμπει ένα event MediaCreated ώστε να ενημερωθούν οι κόμβοι του δικτύου και οι ιστοσελίδες για την επιτυχή καταχώρηση των δεδομένων στο blockchain και να στείλει κάποια δεδομένα. Τέλος, αυξάνει κατά ένα τον μετρητή idStable που θα χρησιμοποιηθεί στην εισαγωγή του επόμενου αρχείου.

```
function deleteMedia(address request_er, uint32 id) external{
	require((request_er == mediaOwnershipTable[id][0].owner1) ||
(request_er == mediaOwnershipTable[id][0].owner2));
	//check if the account owner who requested the deletion of this
specific media file is one of its owners
	mediaOwnershipTable[id][0].owner1 = 0x0;
	mediaOwnershipTable[id][0].owner2 = 0x0;
	mediaOwnershipTable[id][0].percentage1 = 0;
	mediaOwnershipTable[id][0].percentage2 = 0;
	mediaContentTable[id][0].mediaTitle = "";
	mediaContentTable[id][0].artist = "";
	mediaContentTable[id][0].year = 0;
	mediaContentTable[id][0].url = "";
	mediaContentTable[id][0].price = 0;
	//replace blockchain info of deleted file with dummy info
	emit MediaDeleted(id);
}
```

Η συνάρτηση deleteMedia λαμβάνει ως ορίσματα την διεύθυνση του χρήστη που ζήτησε να διαγραφεί κάποιο αρχείο από το blockchain καθώς και το id του αρχείου αυτού. Αρχικά, ελέγχει ότι ο εντολέας είναι ένας από τους καταχωρημένους κατόχους πνευματικών δικαιωμάτων του αρχείου. Αν είναι ένας από αυτούς η συνάρτηση συνεχίζει αλλιώς σταματά η εκτέλεσή της. Αυτό επιτυγχάνεται από την εντολή require που ελέγχει αν ισχύει μια συνθήκη και σταματά την συνάρτηση αν αυτή δεν ισχύει. [62] Εν συνεχεία, αλλάζει τα στοιχεία στους δύο πίνακες που αφορούν το συγκεκριμένο αρχείο με dummy δεδομένα. Η επιλογή αυτή έγινε για εποπτικούς λόγους και στην συνέχεια η αναζήτηση θα αγνοεί τις γραμμές που έχουν μηδενική διεύθυνση στον owner1. Τέλος, εκπέμπεται ένα event MediaDeleted με παράμετρο το id του αρχείου που διαγράφηκε από το blockchain ώστε να ενημερωθούν οι κόμβοι του δικτύου και οι ιστοσελίδες και να γίνει ορατή η διαγραφή του αρχείου στους χρήστες.

```
function updatePrice(address request_er, uint32 id, uint p1, uint p2, uint
price) external{
	require((request_er == mediaOwnershipTable[id][0].owner1) ||
(request_er == mediaOwnershipTable[id][0].owner2));
	//check if the account owner who requested the price change of this
specific media file is one of its owners
	mediaOwnershipTable[id][0].percentage1 = p1;
	mediaOwnershipTable[id][0].percentage2 = p2;
```

```
        mediaContentTable[id][0].price = price;
        emit MediaChangedPrice(id, mediaOwnershipTable[id][0].percentage1,
mediaOwnershipTable[id][0].percentage2, mediaContentTable[id][0].price);
}
```

Η συνάρτηση updatePrice λαμβάνει ως ορίσματα την διεύθυνση του χρήστη που ζήτησε να διαγραφεί κάποιο αρχείο από το blockchain, το id του αρχείου καθώς και τα νέα ποσά που ζητούν οι κάτοχοι των πνευματικών δικαιωμάτων μαζί με την τελική τιμή που θα πωλείται το αρχείο. Αρχικά, ελέγχει ότι ο εντολέας είναι ένας από τους καταχωρημένους κατόχους πνευματικών δικαιωμάτων του αρχείου. Αν είναι ένας από αυτούς, η συνάρτηση συνεχίζει αλλιώς σταματά η εκτέλεσή της. Αυτό επιτυγχάνεται ξανά από την εντολή require. Εν συνεχεία, αλλάζει τα στοιχεία percentage1 και percentage2 από τον πίνακα mediaOwnershipTable καθώς και το price από τον πίνακα mediaContentTable. Τέλος, εκπέμπεται ένα event MediaChangedPrice με παράμετρο το id του αρχείου που διαγράφηκε από το blockchain και τις τρεις νέες τιμές ώστε να ενημερωθούν οι κόμβοι του δικτύου και οι ιστοσελίδες και να γίνει ορατή η αλλαγή στους χρήστες.

```
function updateURL(address request_er, uint32 id, string url) external{
        require((request_er == mediaOwnershipTable[id][0].owner1) ||
(request_er == mediaOwnershipTable[id][0].owner2));
        //check if the account owner who requested the URL change of this
specific media file is one of its owners
        mediaContentTable[id][0].url = url;
        emit MediaChangedURL(id, mediaContentTable[id][0].url);
}
```

Η συνάρτηση updateURL λαμβάνει ως ορίσματα την διεύθυνση του χρήστη που ζήτησε να διαγραφεί κάποιο αρχείο από το blockchain, το id του αρχείου καθώς και την νέα διεύθυνση URL στην οποία θα είναι διαθέσιμο το αρχείο. Αρχικά, ελέγχει ότι ο εντολέας είναι ένας από τους καταχωρημένους κατόχους πνευματικών δικαιωμάτων του αρχείου. Αν είναι ένας από αυτούς η συνάρτηση συνεχίζει αλλιώς σταματά η εκτέλεσή της. Αυτό επιτυγχάνεται και εδώ από την εντολή require. Εν συνεχεία, αλλάζει τα την εγγραφή url από τον πίνακα mediaContentTable. Τέλος, εκπέμπεται ένα event MediaChangedURL με παράμετρο το id του αρχείου που διαγράφηκε από το blockchain και την νέα διεύθυνση URL ώστε να ενημερωθούν οι κόμβοι του δικτύου και οι ιστοσελίδες και να γίνει ορατή η αλλαγή στους χρήστες.

```
function buyMedia(uint32 chosenMediaID, address buyerAddress) external{
        address owner1 = mediaOwnershipTable[chosenMediaID][0].owner1;
        //retrieve owner1 address from mapping
        address owner2 = mediaOwnershipTable[chosenMediaID][0].owner2;
        //retrieve owner2 address from mapping
        uint percentage1 =mediaOwnershipTable[chosenMediaID][0].percentage1;
        //retrieve percentage1 value from mapping
        uint percentage2 =mediaOwnershipTable[chosenMediaID][0].percentage2;
        //retrieve percentage2 value from mapping
        uint price = mediaContentTable[chosenMediaID][0].price;
        //retrieve price value from mapping
        string url = mediaContentTable[chosenMediaID][0].url;
        //retrieve url string from mapping
        if (x == 1){
```

```
            emit PaymentInfo(buyerAddress, owner1, owner2, percentage1,
percentage2, chosenMediaID, url);
      }
       else{
            emit PaymentInfoNtuaToken(buyerAddress, owner1, owner2,
percentage1, percentage2, chosenMediaID, url);
      }
}
```

Η συνάρτηση buyMedia λαμβάνει ως ορίσματα το id του αρχείου που πρόκειται να αγοραστεί από τον χρήστη, την Ethereum διεύθυνση του χρήστη που επιθυμεί να πραγματοποιήσει την αγορά καθώς και μια ακέραια μεταβλητή που θα είναι είτε 1 είτε 2. Αρχικά ανακτά από τον πίνακα mediaOwnershipTable τις διευθύνσεις των ιδιοκτητών του αρχείου μαζί με τις απαιτήσεις του καθενός από την πώλησή του και από τον πίνακα mediaContentTable την τιμή πώλησης του αρχείου και την διεύθυνση URL στην οποία είναι αποθηκευμένο. Για να τα βρει αυτά χρησιμοποιεί το chosenMediaID που είναι το μοναδικό id του αρχείου και ανατρέχει στην αντίστοιχη εγγραφή του κάθε πίνακα. Αφού τα βρει, ελέγχει αν η ακέραια μεταβλητή έχει την τιμή 1 και στην περίπτωση αυτή εκπέμπεται ένα event PaymentInfo ώστε να σταλθεί η συγκεκριμένη πληροφορία στην ιστοσελίδα και να πραγματοποιηθούν οι συναλλαγές με χρήση Ethers. Αλλιώς, εκπέμπεται ένα event PaymentInfoNtuaToken ώστε να σταλθεί η συγκεκριμένη πληροφορία στην ιστοσελίδα και να πραγματοποιηθούν οι συναλλαγές με χρήση NtuaTokens. Η τιμή που θα έχει η μεταβλητή καθορίζεται από το αν ο αγοραστής επέλεξε να αγοράσει το αρχείο πληρώνοντας σε Ethers ή σε NtuaTokens.

```
//EVENTS
//emit these events to send information to the other nodes and the webpages
event MediaCreated(string mediaTitle, string artist, uint year, uint price,
uint id);
event MediaDeleted(uint32 id);
event MediaChangedPrice(uint32 id, uint p1, uint p2, uint price);
event MediaChangedURL(uint32 id, string url);
event PaymentInfo(address buyerAddress, address owner1, address owner2,
uint percentage1, uint percentage2, uint32 id, string url);
event PaymentInfoNtuaToken(address buyerAddress, address owner1, address
owner2, uint percentage1, uint percentage2, uint32 id, string url);
```

Τα παραπάνω events χρησιμοποιούνται για την ενημέρωση των κόμβων του δικτύου και των ιστοσελίδων για πιθανές αλλαγές αλλά και για την μεταφορά δεδομένων. Πιο συγκεκριμένα:

- Το event MediaCreated ενημερώνει για την επιτυχή καταχώριση ενός νέου αρχείου στο δίκτυο του blockchain.
- Το event mediaDeleted ενημερώνει για την επιτυχή διαγραφή ενός αρχείου που ήταν καταχωρημένο στο δίκτυο του blockchain.
- Το event mediaChangedPrice ενημερώνει για την επιτυχή αλλαγή τις τιμής πώλησης και των αποδόσεων για κάθε κάτοχο των πνευματικών δικαιωμάτων ενός αρχείου που ήταν καταχωρημένο στο δίκτυο του blockchain.

- Το event mediaChangedURL ενημερώνει για την επιτυχή αλλαγή της διεύθυνσης στην οποία είναι αποθηκευμένο ένα αρχείο που ήταν καταχωρημένο στο δίκτυο του blockchain
- Το event PaymentInfo στέλνει τα στοιχεία που απαιτούνται για την πώληση του συγκεκριμένου αρχείου στις ιστοσελίδες ώστε να εκτελεστούν οι μεταφορές με χρήση Ethers.
- Το event PaymentInfoNtuaToken στέλνει τα στοιχεία που απαιτούνται για την πώληση του συγκεκριμένου αρχείου στις ιστοσελίδες ώστε να εκτελεστούν οι μεταφορές με χρήση NtuaTokens.

```
//initialize variables
uint32 j;
uint32 idStable;
```

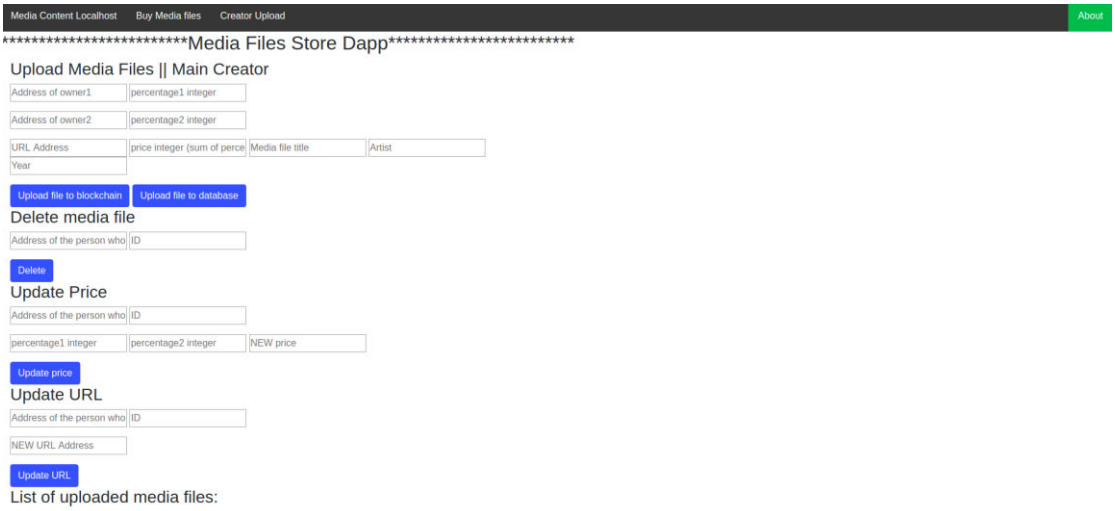
Ορισμός των μεταβλητών που θα χρησιμοποιηθούν στο συμβόλαιο. Το idStable είναι μία μεταβλητή η οποία αρχικοποιείται στο 1 κατά την δημιουργία του συμβολαίου και αυξάνεται κατά ένα κάθε φορά που καταχωρείται ένα νέο αρχείο. Η μεταβλητή αυτή δίνει έναν μοναδικό κωδικό/αριθμό σε κάθε αρχείο που καταχωρείται στο blockchain μέσω του συμβολαίου Uploader.

5.1.3 Περιγραφή διαδικασιών

Στο κεφάλαιο αυτό θα περιγραφούν αναλυτικά κάποιες από τις βασικές διαδικασίες της εφαρμογής. Θα γίνει αναφορά στην ακολουθία από κλήσεις συναρτήσεων που εκτελούν τις βασικές λειτουργίες της εφαρμογής.

Διαδικασία Upload

Ο χρήστης που επιθυμεί να καταχωρήσει κάποιο αρχείο προς πώληση αρχικά ανεβάζει το αρχείο στην βάση δεδομένων πατώντας το πλήκτρο Upload file to database από την ιστοσελίδα Creator upload. Ακολούθως, παίρνει το url στο οποίο ανέβηκε το αρχείο στην βάση δεδομένων και συμπληρώνει τα πεδία που βρίσκονται στην ιστοσελίδα Creator upload. Τα πεδία αυτά είναι οι κάτοχοι των πνευματικών δικαιωμάτων του αρχείου, το ποσοστό της αμοιβής που πρέπει να λάβει ο κάθε ένας από αυτούς, την συνολική αξία του αρχείου που πρέπει να πληρωθεί από τον αγοραστή, λεπτομέρειες όπως το όνομα του άλμπουμ, τον καλλιτέχνη, την χρονολογία έκδοσης καθώς και την διεύθυνση στην οποία είναι ανεβασμένο το περιεχόμενο στην βάση δεδομένων. Μόλις τα συμπληρώσει πατάει το πλήκτρο Upload file to blockchain.



Εικόνα 5-1 Ιστοσελίδα Creator Upload

Η html ιστοσελίδα Creator upload (indexcreator.html) με το πάτημα του πλήκτρου Upload file to blockchain ξεκινά την εκτέλεση της συνάρτησης Upload() που βρίσκεται στον Javascript κώδικα εντός του αρχείου indexcreator.html. Η Upload() θα πάρει τα στοιχεία που εισήχθησαν και θα καλέσει την συνάρτηση createMedia του έξυπνου συμβολαίου Uploader. Με την κλήση αυτή θα στείλει τα ορίσματα που εισήγαγε ο χρήστης για να καταχωρηθούν στο blockchain. Το {gas: 500000} είναι το ποσό που επιλέγεται να διατεθεί από τον εντολέα λογαριασμό για την εκτέλεση της συναλλαγής (εδώ: την κλήση της συνάρτησης).

```
//Upload Button
function Upload() {
      var owner1address = document.getElementById("owner1").value;
      var percentage1number = document.getElementById("percentage1").value;
      var owner2address = document.getElementById("owner2").value;
      var percentage2number = document.getElementById("percentage2").value;
      var url = document.getElementById("url").value;
      var price = parseInt(document.getElementById("price").value, 10);
      var mediaTitle = document.getElementById("mediaTitle").value;
      var artist = document.getElementById("artist").value;
      var year = document.getElementById("year").value;

      CONTRACTCREATOR.createMedia(owner1address, percentage1number,
owner2address, percentage2number, url, price, mediaTitle, artist,
year,{gas: 500000});
      alert("Upload done.");
}
```

Συνάρτηση Upload() του indexcreator.html

Εν συνεχεία, όπως περιγράψαμε και στο κεφάλαιο 5.1.2, εκτελείται η συνάρτηση createMedia του έξυπνου συμβολαίου Uploader, καταχωρείται το νέο αρχείο στους πίνακες mediaOwnershipTable και mediaContentTable και εκπέμπεται ένα γεγονός mediaCreated για την ενημέρωση των κόμβων του δικτύου και των ιστοσελίδων.

```
function createMedia(address o1, uint p1, address o2, uint p2, string url,
uint price, string mediaTitle, string artist, uint year) external {
```

```
        mediaOwnershipTable[idStable].push(media_ownership(o1, p1, o2, p2));
        mediaContentTable[idStable].push(media_content(mediaTitle, artist,
year, url, price));
        //use of push function to insert new rows into mappings
        emit MediaCreated(mediaTitle, artist, year, price, idStable);
        idStable++;
        //increment id after the insertion of a media file
}
```

Συνάρτηση createMedia του smart contract Uploader

Όσον αφορά τις τελευταίες, τόσο στην ιστοσελίδα Creator upload όσο και στην Buy media files, αναπτύχθηκε ένα κομμάτι κώδικα για να «πιάνει» τα γεγονότα αυτά και να αποθηκεύει σε πίνακα τα στοιχεία των αρχείων που θα προβάλλονται στις ιστοσελίδες. Αυτά είναι το ID, ο τίτλος του αρχείου, το όνομα του καλλιτέχνη, το έτος δημιουργίας του και η τελική τιμή πώλησης του.

```
var display = [];
var text = "";
//catch event MediaCreated
var uploadEvent = CONTRACTCREATOR.MediaCreated();
uploadEvent.watch(function(error, result){
      if(!error){
        var i = display.length; //keep starting position
        display[display.length] = result.args.id;
        display[display.length] = result.args.mediaTitle;
        display[display.length] = result.args.artist;
        display[display.length] = result.args.year;
        display[display.length] = result.args.price;

        text += ("ID: " + display[i] + ',  Title: ' + display[i+1] + ',
Artist: ' + display[i+2] + ', Year: ' + display[i+3] + ', Price: ' +
display[i+4]+ "<br>");
        document.getElementById("listOfMedia").innerHTML = text;
      }
      else{
        console.log(error);
      }
});
```

Κώδικας για αποθήκευση των στοιχείων που θα προβάλλονται στις ιστοσελίδες σε μορφή πίνακα. Ο κώδικας αυτός υπάρχει τόσο στην ιστοσελίδα Creator upload όσο και στην Buy media files

Διαδικασία Delete

Η διαδικασία της διαγραφής κάποιου αρχείου ξεκινά από την ιστοσελίδα Creator Upload στην οποία ο χρήστης στον οποίο ανήκουν τα πνευματικά δικαιώματα ενός αρχείου επιλέγει να το διαγράψει. Τότε εισάγει την Ethereum διεύθυνσή του και το μοναδικό id του αρχείου που επιθυμεί να διαγράψει και πατάει το πλήκτρο Delete. Η ιστοσελίδα Creator upload με το πάτημα του πλήκτρου Delete ξεκινά την εκτέλεση της συνάρτησης Delete() που βρίσκεται στον Javascript κώδικα εντός του αρχείου indexcreator.html. Η Delete() θα πάρει τα στοιχεία που εισήχθησαν και θα καλέσει την συνάρτηση deleteMedia του έξυπνου συμβολαίου Uploader. Με την κλήση θα στείλει τα ορίσματα που εισήγαγε ο χρήστης για να μπορέσει να γίνει ο έλεγχος αν ο

χρήστης είναι όντως καταχωρημένος κάτοχος πνευματικών δικαιωμάτων του συγκεκριμένου αρχείου και αν ναι, να διαγραφεί το εν λόγω αρχείο. Ξανά, το {gas: 500000} είναι το ποσό που επιλέγεται να διατεθεί από τον εντολέα λογαριασμό για την εκτέλεση της συναλλαγής (εδώ: την κλήση της συνάρτησης).

```
//Delete Button
function Delete(){
      var request_er = document.getElementById("request_er").value;
      var ID = document.getElementById("ID1").value;
      CONTRACTCREATOR.deleteMedia(request_er, ID, {gas: 500000});
      alert("Media file with ID: "+ID+" was deleted.");
}
```

Συνάρτηση Delete() του indexcreator.html

Μετά τον έλεγχο και τις αλλαγές που θα γίνουν από την συνάρτηση deleteMedia του έξυπνου συμβολαίου, όπως περιγράψαμε και στην προηγούμενη ενότητα, θα γίνει emit ένα γεγονός mediaDeleted το οποίο θα στείλει σαν παράμετρο το id του αρχείου που διαγράφηκε. Αντίστοιχα με την διαδικασία upload, στις ιστοσελίδες Creator upload και Buy media files υπάρχει ο παρακάτω κώδικας που «πιάνει» το συγκεκριμένο event:

```
var flagDeleted = 0;
//catch event MediaDeleted
var deleteEvent = CONTRACTCREATOR.MediaDeleted();
deleteEvent.watch(function(error, result){
      if(!error){
          var m = display.length;
          deletedID = result.args.id;
          for(n=0; n<m; n=n+5){
          //check if owner1 is zero. If yes, it is a deleted file so skip it
             var a = display[n];
             var b = deletedID;
             if(a==0){
                continue;
             }
             var y = a.equals(b);
             if(y){
                 flagDeleted = 1;
                 display[n]=0;display[n+1]="-";
                 display[n+2]="-";
                 display[n+3]=0;
                 display[n+4]=0;
                 break;
             }
             else{
                continue;
             }
      }
      if(flagDeleted == 1){
         displayMediaFromScratch();
         flagDeleted = 0;
      }
  }
  else {
      console.log(error);
```

```
  }
});
```

Κώδικας για αλλαγή των στοιχείων που θα εμφανίζονται στις ιστοσελίδες από τα αρχεία που διαγράφηκαν. Ο κώδικας αυτός υπάρχει τόσο στην ιστοσελίδα Creator upload όσο και στην Buy media files

Διαδικασία Update price

Η διαδικασία της αλλαγής της τιμής κάποιου αρχείου ξεκινά από την ιστοσελίδα Creator Upload στην οποία ο χρήστης στον οποίο ανήκουν τα πνευματικά δικαιώματα ενός αρχείου προτίθεται να εκτελέσει την αλλαγή. Τότε εισάγει την Ethereum διεύθυνσή του, το μοναδικό id του αρχείου που επιθυμεί να αλλάξει την τιμή, τις νέες αμοιβές για κάθε κάτοχο πνευματικών δικαιωμάτων καθώς και την νέα τιμή πώλησης του αρχείου και πατάει το πλήκτρο Update price. Η ιστοσελίδα Creator upload με το πάτημα του πλήκτρου ξεκινά την εκτέλεση της συνάρτησης UpdatePrice() που βρίσκεται στον Javascript κώδικα εντός του αρχείου indexcreator.html. Η UpdatePrice() θα πάρει τα στοιχεία που εισήχθησαν και θα καλέσει την συνάρτηση updatePrice του έξυπνου συμβολαίου Uploader. Με την κλήση θα στείλει τα ορίσματα που εισήγαγε ο χρήστης για να μπορέσει να γίνει ο έλεγχος αν ο χρήστης είναι όντως καταχωρημένος κάτοχος πνευματικών δικαιωμάτων του συγκεκριμένου αρχείου και αν ναι, να πραγματοποιηθούν οι αλλαγές στο εν λόγω αρχείο. Ξανά, το {gas: 500000} είναι το ποσό που επιλέγεται να διατεθεί από τον εντολέα λογαριασμό για την εκτέλεση της συναλλαγής (εδώ: την κλήση της συνάρτησης).

```
//Update Price Button
function UpdatePrice() {
	var request_er2 = document.getElementById("request_er2").value;
	var ID2 = document.getElementById("ID2").value;
	var percentage1number =
document.getElementById("percentage1.2").value;
	var percentage2number =
document.getElementById("percentage2.2").value;
	var price = parseInt(document.getElementById("price2").value, 10);
	CONTRACTCREATOR.updatePrice(request_er2, ID2, percentage1number,
percentage2number, price, {gas: 500000});
	alert("Price updated in media file with ID: "+ ID2);
}
```

Συνάρτηση UpdatePrice() του indexcreator.html

Μετά τον έλεγχο και τις αλλαγές που θα γίνουν από την συνάρτηση uploadPrice του έξυπνου συμβολαίου, όπως περιγράψαμε και στην προηγούμενη ενότητα, θα γίνει emit ένα γεγονός MediaChangedPrice το οποίο θα στείλει σαν παράμετρο το id του αρχείου και τις τρεις νέες τιμές. Οι νέες τιμές στέλνονται για να μπορέσει να τις πάρει και η ιστοσελίδα Buy media files. Αντίστοιχα με τις προηγούμενες διαδικασίες, στις ιστοσελίδες Creator upload και Buy media files υπάρχει ο παρακάτω κώδικας που «πιάνει» το συγκεκριμένο event:

```
var  flagPriceUpdated = 0;
//catch event MediaChangedPrice
var updatePriceEvent = CONTRACTCREATOR.MediaChangedPrice();
```

```
updatePriceEvent.watch(function(error, result){
     if(!error){
       var g = display.length; //kratao to megethos tou pinaka gia tin for
       var tempID = result.args.id;
       var tempP1 = result.args.p1;
       var tempP2 = result.args.p2;
       var tempPrice = result.args.price;
       for(s=0; s<g; s = s + 5){
          var a = display[s];
          if(a==0){
             continue;
          }
          if(display[s].equals(tempID)){
             flagPriceUpdated = 1;
             display[s+4] = tempPrice;
             break;
          }
          else{
             continue;
          }
       }
       if(flagPriceUpdated == 1){
           displayMediaFromScratch();
           flagPriceUpdated = 0;
       }
     }
     else {
       console.log(error);
     }
});
```

Κώδικας για αλλαγή των στοιχείων που θα εμφανίζονται στις ιστοσελίδες από τα αρχεία που άλλαξε η τιμή τους. Ο κώδικας αυτός υπάρχει τόσο στην ιστοσελίδα Creator upload όσο και στην Buy media files

Διαδικασία Update URL

Η διαδικασία της αλλαγής της διεύθυνσης url στην οποία είναι αποθηκευμένο κάποιο αρχείο ξεκινά από την ιστοσελίδα Creator Upload στην οποία ο χρήστης στον οποίο ανήκουν τα πνευματικά δικαιώματα ενός αρχείου προτίθεται να εκτελέσει την αλλαγή. Τότε εισάγει την Ethereum διεύθυνσή του, το μοναδικό id του αρχείου που επιθυμεί να αλλάξει την διεύθυνση και την νέα αυτή διεύθυνση και πατάει το πλήκτρο Update URL. Η ιστοσελίδα Creator upload με το πάτημα του πλήκτρου ξεκινά την εκτέλεση της συνάρτησης UpdateURL() που βρίσκεται στον Javascript κώδικα εντός του αρχείου indexcreator.html. Η UpdateURL() θα πάρει τα στοιχεία που εισήχθησαν και θα καλέσει την συνάρτηση updateURL του έξυπνου συμβολαίου Uploader. Με την κλήση θα στείλει τα ορίσματα που εισήγαγε ο χρήστης για να μπορέσει να γίνει ο έλεγχος αν ο χρήστης είναι όντως καταχωρημένος κάτοχος πνευματικών δικαιωμάτων του συγκεκριμένου αρχείου και αν ναι, να πραγματοποιηθούν οι αλλαγές στο εν λόγω αρχείο. Ξανά, το `{gas: 500000}` είναι το ποσό που επιλέγεται να διατεθεί από τον εντολέα λογαριασμό για την εκτέλεση της συναλλαγής (εδώ: την κλήση της συνάρτησης).

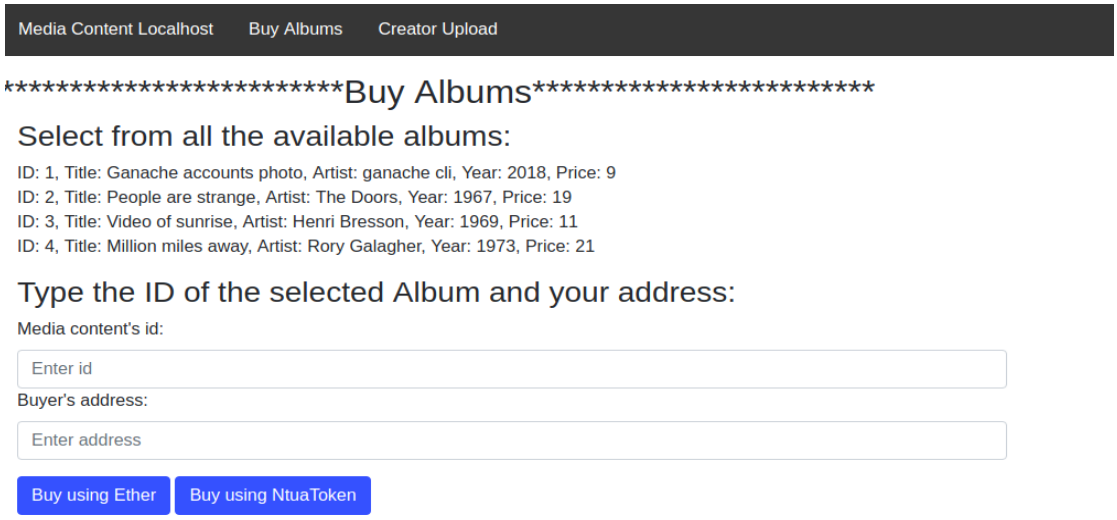
```
//Update URL Button
function UpdateURL() {
      var request_er3 = document.getElementById("request_er3").value;
      var ID3 = document.getElementById("ID3").value;
      var url3 = document.getElementById("url3").value;
      CONTRACTCREATOR.updateURL(request_er3, ID3, url3, {gas: 500000});
      alert("URL updated in media file with ID: "+ ID3);
}
```

Συνάρτηση UpdateURL() του indexcreator.html

Μετά τον έλεγχο και τις αλλαγές που θα γίνουν από την συνάρτηση uploadURL του έξυπνου συμβολαίου, όπως περιγράψαμε και στην προηγούμενη ενότητα, θα γίνει emit ένα γεγονός MediaChangedURL το οποίο θα στείλει σαν παράμετρο το id του αρχείου και την νέα διεύθυνση URL. Η διεύθυνση URL στέλνεται για να μπορέσει να την πάρει και η ιστοσελίδα Buy media files ώστε να την δώσει στον αγοραστή όταν ολοκληρωθεί επιτυχώς η αγορά.

Διαδικασία Buy

Η λειτουργία της αγοράς κάποιου αρχείου πολυμέσων ξεκινά από την ιστοσελίδα Buy Media Content ως εξής: ο χρήστης που ενδιαφέρεται να αγοράσει κάποιο αρχείο πηγαίνει στην σελίδα και βλέπει την λίστα με τα διαθέσιμα αρχεία προς πώληση. Για την αγορά πρέπει να συμπληρώσει την Ethereum διεύθυνσή του και το ID του αρχείου που επιθυμεί να αγοράσει. Ακολούθως, διαλέγει αν θέλει να πληρώσει με Ethers ή με το κρυπτονόμισμα που αναπτύχθηκε για τις ανάγκες της εφαρμογής και πατάει το κατάλληλο πλήκτρο από τα Buy using Ether και Buy using NtuaToken.



Εικόνα 5-2 Ιστοσελίδα Buy media files

```
//buy using ether
function Buy(){
      var ID = document.getElementById("ID").value;
```

```
        var buyer = document.getElementById("buyer").value;
       //from the smart contract I want to retrieve the payment info, not to
do the transaction
        CONTRACTCREATOR.buyMedia(ID, buyer, 1, {gas: 50000});
}
```

```
//buy using NtuaToken
//παραδοχή ότι το NtuaToken θα έχει σταθερή ισοτιμία με το Ether (1:1)
function BuyNtuaToken(){
        var ID = document.getElementById("ID").value;
        var buyer = document.getElementById("buyer").value;
        //from the smart contract I want to retrieve the payment info, not
to do the transaction
        CONTRACTCREATOR.buyMedia(ID, buyer, 2, {gas: 50000});
}
```

Με το πάτημα του πλήκτρου Buy using Ether, η σελίδα Buy media files εκκινεί την συνάρτηση Buy() που βρίσκεται στον Javascript κώδικα εντός του αρχείου indexbuy.html. Με την σειρά της η Buy() καλεί την συνάρτηση buyMedia του έξυπνου συμβολαίου με ορίσματα το ID του αρχείου που πρόκειται να αγοραστεί, την Ethereum διεύθυνση του αγοραστή και τον αριθμό 1.

Αντίστοιχα, με το πάτημα του πλήκτρου Buy using NtuaToken, η σελίδα Buy media files εκκινεί την συνάρτηση BuyNtuaToken() που βρίσκεται στον Javascript κώδικα εντός του αρχείου indexbuy.html. Με την σειρά της η BuyNtuaToken() καλεί την συνάρτηση buyMedia του έξυπνου συμβολαίου με ορίσματα το ID του αρχείου που πρόκειται να αγοραστεί, την Ethereum διεύθυνση του αγοραστή και τον αριθμό 2.

Η διαφορά αυτή στο τρίτο όρισμα με το οποίο καλείται η συνάρτηση buyMedia του έξυπνου συμβολαίου Uploader, μας δίνει την δυνατότητα να εκπέμψουμε διαφορετικό γεγονός κατά την διάρκεια εκτέλεσης της συνάρτησης buyMedia του Uploader. Πιο συγκεκριμένα, όπως φαίνεται και στον παρακάτω κώδικα, η συνάρτηση buyMedia δεν εκτελεί τις συναλλαγές αλλά βρίσκει τα απαραίτητα στοιχεία που χρειάζονται για την εκτέλεση των συναλλαγών και τα στέλνει στις ιστοσελίδες. Ο τρόπος αποστολής τους είναι με την εκπομπή του γεγονότος PaymentInfo ή του γεγονότος PaymentInfoNtuaToken για την εκτέλεση των συναλλαγών από τον αγοραστή προς τους ιδιοκτήτες με μεταφορά Ethers ή NtuaTokens αντίστοιχα.

```
function buyMedia(uint32 chosenMediaID, address buyerAddress) external{
        address owner1 = mediaOwnershipTable[chosenMediaID][0].owner1;
        //retrieve owner1 address from mapping
        address owner2 = mediaOwnershipTable[chosenMediaID][0].owner2;
        //retrieve owner2 address from mapping
        uint percentage1 =mediaOwnershipTable[chosenMediaID][0].percentage1;
        //retrieve percentage1 value from mapping
        uint percentage2 =mediaOwnershipTable[chosenMediaID][0].percentage2;
        //retrieve percentage2 value from mapping
        uint price = mediaContentTable[chosenMediaID][0].price;
        //retrieve price value from mapping
        string url = mediaContentTable[chosenMediaID][0].url;
        //retrieve url string from mapping
        if (x == 1){
```

```
            emit PaymentInfo(buyerAddress, owner1, owner2, percentage1,
percentage2, chosenMediaID, url);
      }
      else{
            emit PaymentInfoNtuaToken(buyerAddress, owner1, owner2,
percentage1, percentage2, chosenMediaID, url);
      }
}
```

Συνάρτηση buyMedia του έξυπνου συμβολαίου Uploader

Όπως και στις προηγούμενες διαδικασίες δημιούργησα κατάλληλο κώδικα για να «πιάνει» τα γεγονότα PaymentInfo και PaymentInfoNtuaTokens. Ο κώδικας αυτός είναι εντός της ιστοσελίδας Buy media files και παρουσιάζεται ακολούθως:

```
var paymentInfoEvent = CONTRACTCREATOR.PaymentInfo();
paymentInfoEvent.watch(function(error, result){
     if(!error){
      var tempBuyerAddress = result.args.buyerAddress;
      var tempowner1 = result.args.owner1;
      var tempowner2 = result.args.owner2;
      var tempP1 = result.args.percentage1;
      var tempP2 = result.args.percentage2;
      var tempid = result.args.id;
      var tempurl = result.args.url;
      var amountToSend1 =web3.toWei(tempP1, "ether");
      var amountToSend2 =web3.toWei(tempP2, "ether");
      var balance = web3.eth.getBalance(tempBuyerAddress);
      if(balance - amountToSend1 - amountToSend2 >= 0){
      //check if buyer's balance has enough Ethers, if yes continue
         web3.eth.sendTransaction({from:tempBuyerAddress, to:tempowner1,
value:amountToSend1});
         web3.eth.sendTransaction({from:tempBuyerAddress, to:tempowner2,
value:amountToSend2});
         //use web3 to do the transactions
         alert("Please access the file that you have bought from the
following url: "+ tempurl);
      }
      else{
          alert("Account "+ tempBuyerAddress+" does not have enough Ether
for this Media item.");
      }
     }
     else {
      console.log(error);
     }
});
```

Γίνεται έλεγχος αν το υπόλοιπο σε Ethers του λογαριασμού του αγοραστή επαρκεί για την αγορά του αρχείου πολυμέσων. Στην περίπτωση που δεν επαρκεί, εμφανίζεται αναδυόμενο παράθυρο στον χρήστη πως δεν επαρκούν τα Ethers του λογαριασμού του για την αγορά του συγκεκριμένου αρχείου. Στην αντίθετη περίπτωση, γίνονται δύο άμεσες συναλλαγές μεταξύ του αγοραστή και των δύο ιδιοκτητών του αρχείου και έπειτα εμφανίζεται σε αναδυόμενο παράθυρο η διεύθυνση url στην οποία είναι αποθηκευμένο το αρχείο στον χρήστη ώστε να αποκτήσει πρόσβαση σε αυτό ή να το κατεβάσει. Τόσο για την πραγματοποίηση των

συναλλαγών όσο και για τον έλεγχο του διαθέσιμου υπολοίπου για τον λογαριασμό του αγοραστή, χρησιμοποιήθηκαν εντολές του web3.

```
var paymentInfoEventNtuaToken = CONTRACTCREATOR.PaymentInfoNtuaToken();
paymentInfoEventNtuaToken.watch(function(error, result){
     if(!error){
       var tempBuyerAddress = result.args.buyerAddress;
       var tempowner1 = result.args.owner1;
       var tempowner2 = result.args.owner2;
       var tempP1 = result.args.percentage1;
       var tempP2 = result.args.percentage2;
       var tempid = result.args.id;
       var tempurl = result.args.url;
       var butAmountTokens1 = parseInt(tempP1, 10);
       var butAmountTokens2 = parseInt(tempP2, 10);

       CONTRACTNTUA.transferFrom(tempBuyerAddress, tempowner1,
web3.toWei(butAmountTokens1, "ether"));
       CONTRACTNTUA.transferFrom(tempBuyerAddress, tempowner2,
web3.toWei(butAmountTokens2, "ether"));
       //the check if buyer has enough NtuaTokens is done in the smart
contract NtuaToken
       alert("Please access the file that you have bought from the
following url: "+ tempurl);
     }
     else {
       console.log(error);
     }
});
```

Στην περίπτωση που γίνει “catch” κάποιο event paymentInfoEventNtuaToken, δεν θα χρησιμοποιηθεί web3 για τον έλεγχο της διαθεσιμότητας NtuaTokens από τον αγοραστή ούτε για τις συναλλαγές. Αντιθέτως, θα κληθεί δύο φορές η εξωτερική συνάρτηση transferFrom του έξυπνου συμβολαίου NtuaToken δίνοντας ως ορίσματα την διεύθυνση του αποστολέα NtuaTokens, την διεύθυνση του παραλήπτη καθώς και το ποσό μεταφοράς για κάθε μια από τις δύο συναλλαγές. Ομοίως με προηγουμένως, μόλις ολοκληρωθούν με επιτυχία οι μεταφορές, εμφανίζεται σε αναδυόμενο παράθυρο η διεύθυνση url στην οποία είναι αποθηκευμένο το αρχείο στον χρήστη ώστε να αποκτήσει πρόσβαση σε αυτό ή να το κατεβάσει.

5.1.4 Παρατηρήσεις

Παρατήρηση: Έχει γίνει προσπάθεια βελτιστοποίησης του κώδικα των συμβολαίων ώστε οι συναρτήσεις τους να έχουν μικρή απαίτηση σε gas, στοιχείο πολύ σημαντικό, αφού έτσι μειώνεται σε μεγάλο βαθμό το επιπλέον κόστος των προμηθειών που δίνονται για την ολοκλήρωση των συναλλαγών. Μικρή απαίτηση σε gas μεταφράζεται σε μικρότερο κόστος για την συναλλαγή.

Παρατήρηση: Στα συμβόλαια NtuaToken και Uploader έχουν χρησιμοποιηθεί κατά βάση external συναρτήσεις αντί για public. Αυτό γίνεται για να μειωθεί το κόστος (gas) που απαιτείται για την εκτέλεση της κάθε συνάρτησης. Όταν μία συνάρτηση είναι external και καλείται έξω από το συμβόλαιο με πολλά δεδομένα τότε συνήθως

απαιτεί λιγότερους πόρους (gas) για την εκτέλεση της συγκριτικά με την περίπτωση να ήταν public.

Καθορισμός της διεύθυνσης ενός έξυπνου συμβολαίου.

Η διεύθυνση του κάθε συμβολαίου υπολογίζεται ντετερμινιστικά στο Ethereum blockchain. Εξαρτάται αποκλειστικά από την διεύθυνση του δημιουργού του και από τον αριθμό των συναλλαγών που αυτός έχει στείλει στο blockchain (nonce). Αυτά τα δύο κρυπτογραφούνται με τον αλγόριθμο RLP [63] και το αποτέλεσμα περνάει στην συνέχεια μέσα από την συνάρτηση κατακερματισμού Keccak-256 [64].

5.2 Βάση δεδομένων

Η εφαρμογή για να είναι πλήρως αποκεντρωμένη θα έπρεπε να λειτουργεί χωρίς την χρήση βάσης δεδομένων. Η ουσία της αποκεντρωμένης εφαρμογής (DApp) είναι κανένα φυσικό πρόσωπο ή οργανισμός (π.χ. εταιρεία, κυβέρνηση κτλ) να μην μπορεί να ελέγχει μονόπλευρα την λειτουργία της. Στην υλοποίησή μας ανεβάζουμε όλα τα στοιχεία που αφορούν τα αρχεία που διατίθενται προς πώληση στο δίκτυο του Blockchain, εκτός από τα ίδια τα αρχεία τα οποία ανεβάζουμε σε μια βάση δεδομένων και παίρνουμε την διεύθυνση url στην οποία είναι αποθηκευμένα ώστε να υπάρχει πρόσβαση σε αυτά από τους αγοραστές.

Θεωρητικά είναι εφικτό να ανεβάσουμε τα ίδια τα αρχεία στο blockchain συμπιέζοντάς τα και ανεβάζοντάς τα σε δεκαεξαδική μορφή αλλά αυτό απέχει πολύ από το να κάνει την εφαρμογή αποδοτική και εύχρηστη, πράγμα που θα εξηγηθεί στην συνέχεια.

Παρόλο που η λειτουργία χωρίς βάσεις δεδομένων είναι θεωρητικά εφικτή, στην πραγματικότητα μία τέτοια προσέγγιση θα δυσχέραινε τη χρήση της εφαρμογής και θα περιόριζε την αποδοτικότητά της. Ο λόγος δεν είναι άλλος από την διεκπεραιωτική ικανότητα (throughput) και την χρονοκαθυστέρηση (latency), εγγενή χαρακτηριστικά του blockchain.

Χρησιμοποίησα την μη-σχεσιακή βάση δεδομένων MongoDB και το εργαλείο GridFS για το σπάσιμο των αρχείων σε μικρά κομμάτια και κατά συνέπεια το γρηγορότερο ανέβασμα και κατέβασμά τους. Στόχος ήταν η βάση δεδομένων να εκτελεί όσο το δυνατόν λιγότερες λειτουργίες για να έχει η εφαρμογή μας όσο το δυνατόν περισσότερο αποκεντρωμένο χαρακτήρα. Συνεπώς, στην βάση δεδομένων ανεβάζουμε μόνο τα αρχεία και παίρνουμε την διεύθυνση url στην οποία είναι αποθηκευμένα. Επειδή η εφαρμογή προορίζεται κατά κόρον για αρχεία πολυμέσων, έχει δημιουργηθεί ένας διαχωρισμός για αρχεία εικόνας, βίντεο και ήχου όπως παρουσιάζεται στο παρακάτω κομμάτι κώδικα.

```
app.get('/', (req, res) => {
  gfs.files.find().toArray((err, files) => {
    // Check if files
    if (!files || files.length === 0) {
      res.render('index', { files: false });
    } else {
      files.map(file => {
        if (
```

```
            file.contentType === 'image/jpeg' ||
            file.contentType === 'image/png'
          ) {
            file.isImage = true;
          } else {
            file.isImage = false;
          }
        });
        files.map(file => {
          if (file.contentType === 'audio/mp3')
          {
            file.isAudio = true;
          } else {
            file.isAudio = false;
          }
        });
        files.map(file => {
          if (file.contentType === 'video/mp4')
          {
            file.isVideo = true;
          } else {
            file.isVideo = false;
          }
        });
        res.render('index', { files: files });
      }
    });
  });
```

Έτσι τα αρχεία ήχου θα βρίσκονται στην σελίδα localhost:8081/audio/, τα αρχεία εικόνας localhost:8081/image/ και τα αρχεία βίντεο localhost:8081/video/.

Αν δεν γινόταν χρήση της βάσης δεδομένων θα υπήρχε πρόβλημα όταν θα ήθελα να ανεβάζω μεγάλα αρχεία από άποψη χρόνου, μιας και θα ήταν πολύ χρονοβόρα η μετατροπή τους σε δεκαεξαδική μορφή αλλά και το ανέβασμα τους στο δίκτυο του blockchain. Θα υπήρχε όμως και περιορισμός από άποψη κόστους μιας και θα χρειαζόταν μεγάλη ποσότητα gas για το ανέβασμα μεγάλων αρχείων στο δίκτυο [65].

5.3 Προσωρινή αποθήκευση δεδομένων στις ιστοσελίδες

Ακολούθως, περιγράφεται το σενάριο λειτουργίας χωρίς την προσωρινή αποθήκευση σε ιστοσελίδα των δεδομένων των αρχείων που είναι απαραίτητα να παρουσιάζονται στους πιθανούς αγοραστές, ώστε να φανεί η χρησιμότητα της συγκεκριμένης υλοποίησης.

Όταν κάποιος χρήστης επισκέπτεται την ιστοσελίδα Buy media files, πρέπει να βλέπει τα αρχεία που διατίθενται προς πώληση. Εφόσον η πληροφορία αυτή δεν είναι κάπου αποθηκευμένη, θα πρέπει να ανακτάται στον φυλλομετρητή του πελάτη από το blockchain κάθε φορά που φορτώνεται μία ιστοσελίδα που την χρειάζεται. Το παραπάνω όμως έχει επίπτωση στην απόδοση της εφαρμογής μιας και θα εισάγει

συνεχώς μεγάλη καθυστέρηση, ειδικά όταν τα διαθέσιμα αρχεία είναι πολλά και πρέπει να σταλθούν πολλά events. Για παράδειγμα, κάθε φορά που κάποιος χρήστης θα ήθελε να δει τα διαθέσιμα αρχεία, θα έπρεπε να διαβάσει όλα τα μπλοκ[2] από την ενεργοποίηση των συμβολαίων έως το τελευταίο, ώστε να βρει όλα τα events mediaCreated που έχουν παραχθεί και σηματοδοτούν την καταχώριση αρχείων· έπειτα, θα πρέπει να βρει και όλα τα events MediaDeleted, MediaChangedPrice, και MediaChangedURL ώστε να βεβαιωθεί ότι βλέπει τα επικαιροποιημένα διαθέσιμα αρχεία κάθε στιγμή. Συνέπεια των παραπάνω θα ήταν αυξημένη καθυστέρηση στην φόρτωση κάθε ιστοσελίδας. Έτσι λοιπόν προσπαθήσαμε να βελτιώσουμε αυτή τη λύση για να αποφευχθούν κάποιες περιττές καθυστερήσεις.

Κατά συνέπεια, επιλέχθηκε η υλοποίηση κατά την οποία οι ιστοσελίδες Creator upload και Buy media files «ακούν» συνεχώς για πιθανή εκπομπή γεγονότων από τα έξυπνα συμβόλαια και κρατούν-ενημερώνουν κατάλληλα τα δεδομένα που είναι απαραίτητα προς παρουσίαση. Έτσι δημιουργήθηκε ένας πίνακας στον κώδικα Javascript στον οποίο θα προστίθενται γραμμές με τις απαιτούμενες προς επίδειξη πληροφορίες κάθε φορά.

Το παραπάνω μας εξοικονομεί κάποια άσκοπα διαβάσματα όλης της καταχωρημένης πληροφορίας από το blockchain καθώς δεν χρειάζεται να διαβαστούν όλα τα διαθέσιμα αρχεία από την αρχή όταν δεν έχει μεσολαβήσει ενέργεια delete, updatePrice ή updateURL.

5.4 Μετρικές απόδοσης

Οι αναγνώσεις από το blockchain είναι δωρεάν, ενώ οι εγγραφές και οι υπολογισμοί κοστίζουν.

Ο χρόνος αναμονής κάθε συναλλαγής δεν εξαρτάται μόνο από τον κώδικα που πρέπει να εκτελεστεί, αλλά και από άλλες παραμέτρους. Μερικές από αυτές τις παραμέτρους είναι η τιμή του gas που θέτει ο χρήστης, η τιμή του gas που δέχονται οι miners, τον φόρτος του δικτύου κτλ.

Στο ιδιωτικό Ethereum blockchain (χρήση Ganache test network) ο χρόνος αναμονής της κάθε συναλλαγής ήταν αμελητέος.

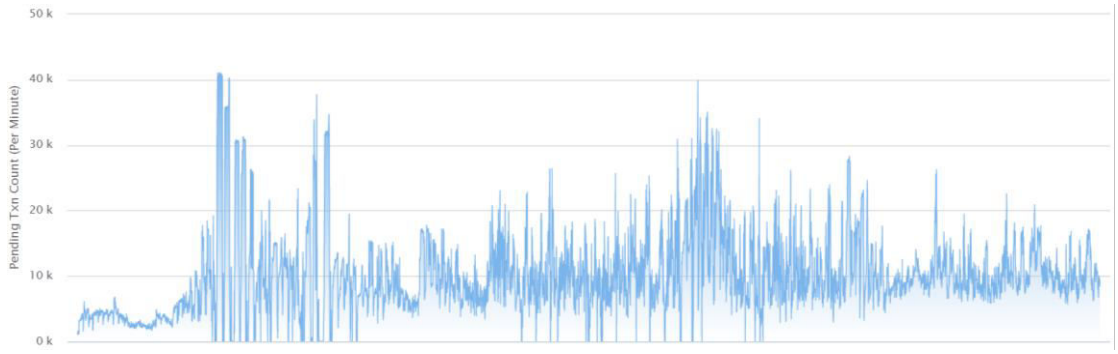
Στο Ethereum δίκτυο (και στο κυρίως και στα Test Nets) ο χρόνος αναμονής της κάθε συναλλαγής μπορεί να ποικίλλει από μερικά δευτερόλεπτα έως μερικές ώρες, αναλόγως της απαίτησης σε gas της συναλλαγής και της τιμής του gas που προσφέρεται στους miners.

Σε κάθε περίπτωση οι συναρτήσεις πρέπει να έχουν την ελάχιστη απαίτηση σε gas, διότι έτσι οι συναλλαγές θα πραγματοποιούνται γρηγορότερα και με μικρότερη προμήθεια προς τους miners. $Προμήθεια = Gas * Gas\ Price$ βλέπουμε λοιπόν ότι με μικρότερη απαίτηση σε gas μπορούμε να προσφέρουμε μεγαλύτερο gas price και έτσι η συναλλαγή μας να πραγματοποιηθεί γρηγορότερα, αυξάνοντας την απόδοση όλης της εφαρμογής.

Ακολουθούν διαγράμματα που δείχνουν την μεταβολή της τιμής του gas και του μεγέθους της ουράς αναμονής των συναλλαγών προς επικύρωση με την πάροδο του χρόνου στο Ethereum Main Net.

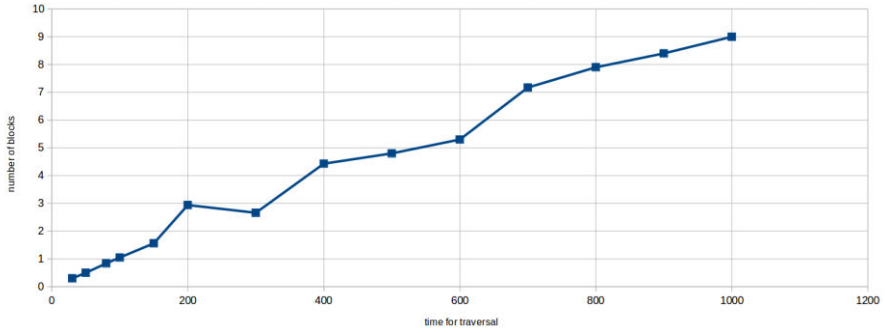
[2] Η διαδικασία αυτή θα γινόταν με την χρήση client-side JavaScript.

Εικόνα 5-3 Τιμή του gas (Πηγή: etherscan.io)

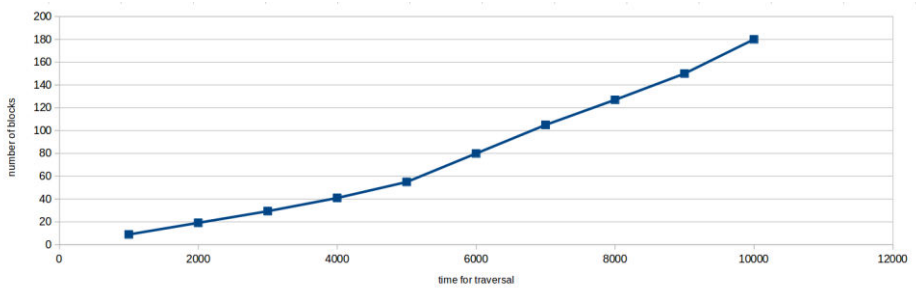


Εικόνα 5-4 Μέγεθος ουράς αναμονής (Πηγή: etherscan.io)

Επίσης, μετρήθηκε ο χρόνος που χρειάζεται για να διασχιστούν όλα τα blocks του blockchain προς αναζήτηση κάποιου συγκεκριμένου Event. Πιο συγκεκριμένα, εκτέλεσα μεγάλο αριθμό συναλλαγών (μέχρι 10000) και κάθε μια από αυτές εξέπεμπε ένα συγκεκριμένο event με κάποια πληροφορία. Το πείραμα που έγινε λοιπόν, ήταν η διάσχιση όλων των blocks προς αναζήτηση της συγκεκριμένης πληροφορίας. Αξίζει να σημειώσουμε ότι στο Ganache κάθε νέα συναλλαγή καταγράφεται σε νέο block. Το συμπέρασμα που προέκυψε είναι, όπως αναμέναμε, ότι ο χρόνος διάσχισης κάθε μπλοκ είναι σε γενικές γραμμές ίδιος και εξαρτάται σε μεγάλο βαθμό και από τα στοιχεία του ηλεκτρονικού υπολογιστή που γίνεται το πείραμα αλλά και από τη χρήση της μνήμης και του επεξεργαστή του υπολογιστή από άλλες διεργασίες την ίδια στιγμή. Τα αποτελέσματα των μετρήσεων παρουσιάζονται στα παρακάτω διαγράμματα:



Εικόνα 5-5 Χρόνος διάσχισης blocks για αριθμό blocks μέχρι 1000



Εικόνα 5-6 Χρόνος διάσχισης blocks για αριθμό blocks από 1000 μέχρι 10000

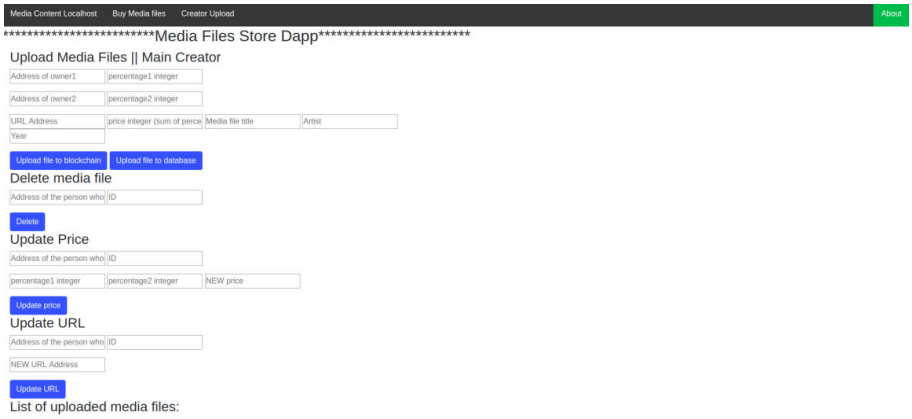
6

Επίδειξη λειτουργικότητας εφαρμογής

Στο κεφάλαιο αυτό θα γίνει παρουσίαση της εφαρμογής και του τρόπου λειτουργίας της. Η παρουσίαση του κεφαλαίου αυτού αφορά το κομμάτι της εφαρμογής που αφορά τον χρήστη, ενώ στο προηγούμενο κεφάλαιο γίνεται παρουσίαση του τρόπου υλοποίησής της.

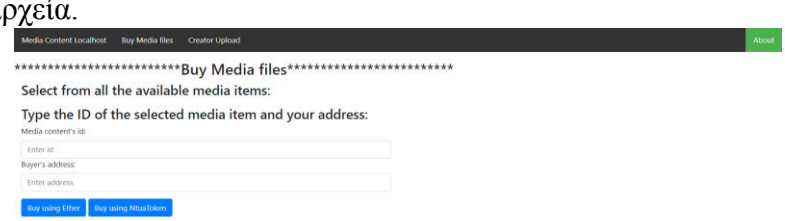
Η εφαρμογή αποτελείται από 4 βασικές ιστοσελίδες: την σελίδα Creator Upload, την Buy Media Files, την Media Content Localhost και την ιστοσελίδα διαχείρισης της βάσης δεδομένων.

Η σελίδα Creator Upload είναι αυτή στην οποία οι χρήστες μπορούν να καταχωρήσουν προς πώληση τα αρχεία των οποίων κατέχουν μέρος των πνευματικών δικαιωμάτων.



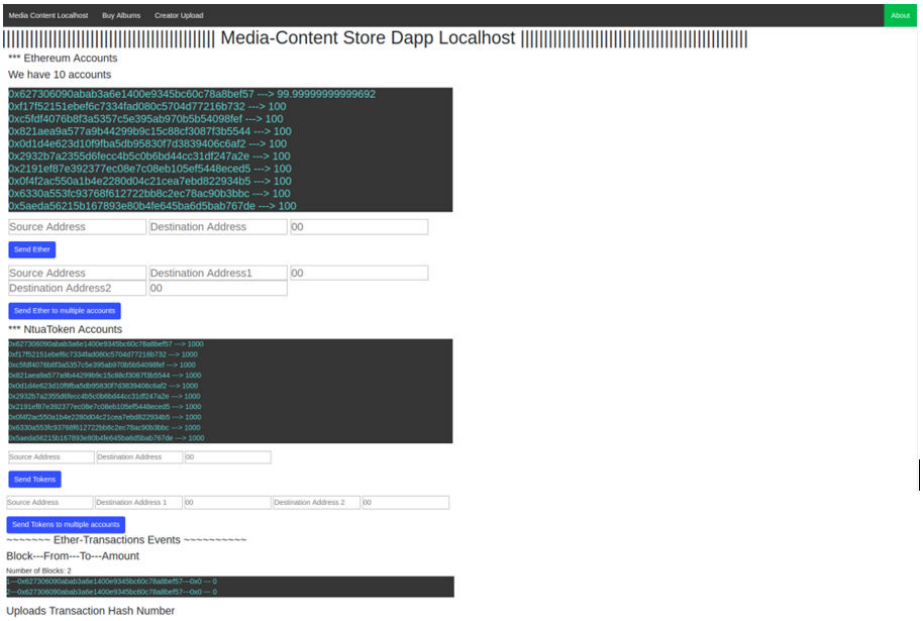
Εικόνα 6-1 Creator Upload, καταχώρηση αρχείων

Στην σελίδα Buy Media Files οι χρήστες μπορούν να αγοράσουν τα διαθέσιμα προς πώληση αρχεία.



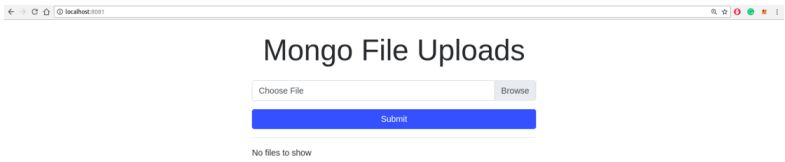
Εικόνα 6-2 Buy Albums, αγορά αρχείων

Στην σελίδα Media Content Locahost μπορούμε να παρακολουθήσουμε το υπόλοιπο κάθε λογαριασμού σε Ethers και NtuaTokens καθώς και τις συναλλαγές που καταγράφονται στο δίκτυο του blockchain.



Εικόνα 6-3 Media Content Localhost, παρακολούθηση πορτοφολιών και συναλλαγών στο δίκτυο του blockchain

Η ιστοσελίδα της βάσης δεδομένων σκόπιμα έχει πολύ λίγες δυνατότητες, μιας και επιθυμούμε η εφαρμογή να είναι όσο το δυνατότερο αποκεντροποιημένη. Συνεπώς, δίνει την δυνατότητα για ανέβασμα και διαγραφή αρχείων.



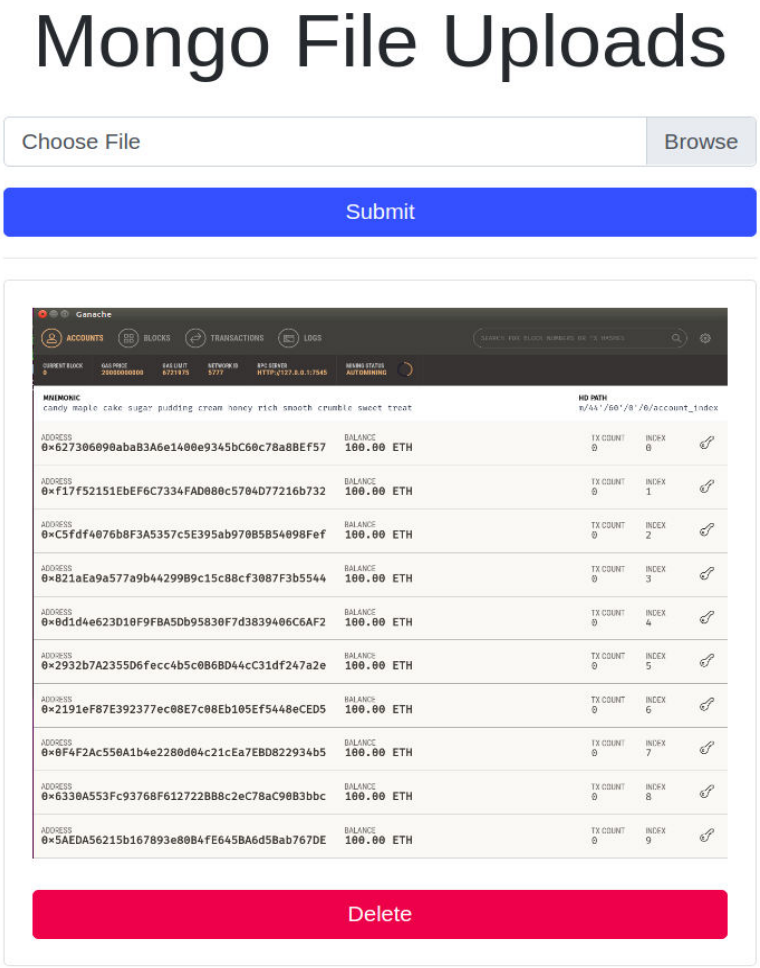
Εικόνα 6-4 Ιστοσελίδα βάσης δεδομένων όταν δεν περιέχει κανένα αρχείο

Περιγραφή των λειτουργιών που δίνονται από τις ιστοσελίδες:

Λειτουργία Upload από ιστοσελίδα Creator Upload:

Παρατηρούμε ότι ο χρήστης με ρόλο ιδιοκτήτη μπορεί αρχικά να ανεβάσει το αρχείο του στην βάση δεδομένων πατώντας το κουμπί Upload file to database. Τότε θα του ανοίξει μια νέα καρτέλα που θα τον οδηγήσει στην ιστοσελίδα της βάσης δεδομένων.

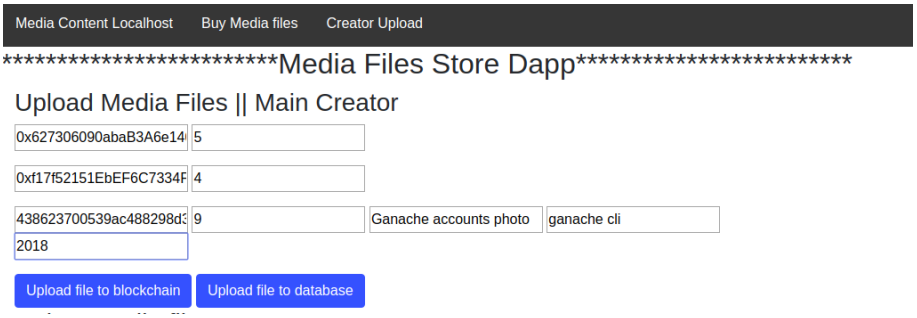
Εκεί ο χρήστης πατάει το πλήκτρο Browse για να βρει από το σύστημα αρχείων το αρχείο που επιθυμεί να ανεβάσει στην βάση δεδομένων. Μόλις το επιλέξει, πατάει το πλήκτρο Submit και αυτό ανεβαίνει στην βάση. Ακολούθως, λαμβάνει το νέο όνομα με το οποίο ανέβηκε το αρχείο στην βάση δεδομένων. Το νέο όνομα αποτελείται από έναν δεκαεξαψήφιο κωδικό ακολουθούμενο από την κατάληξη του αρχείου. Ο δεκαεξαψήφιος κωδικός προέκυψε μέσω της συνάρτησης crypto.randomBytes(size[, callback]) [66].



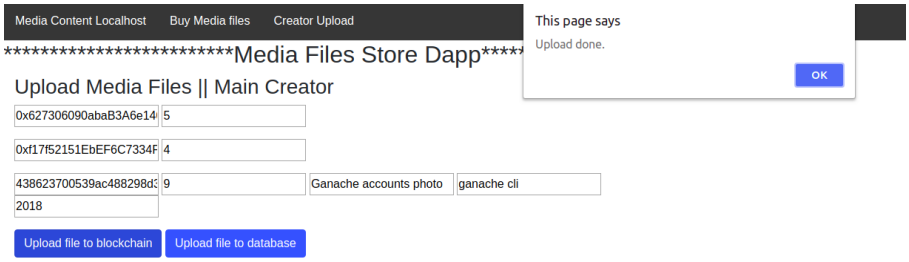
Εικόνα 6-5 Βάση δεδομένων μετά το ανέβασμα αρχείου εικόνας

Στην συνέχεια, ο χρήστης παίρνει το παραχθέν όνομα αρχείου και δημιουργεί τη διεύθυνση url στην οποία το αρχείο είναι διαθέσιμο (localhost:8081/[item type]/[item name]) και επανέρχεται στην ιστοσελίδα Creator Upload (Εικόνα 6-1).

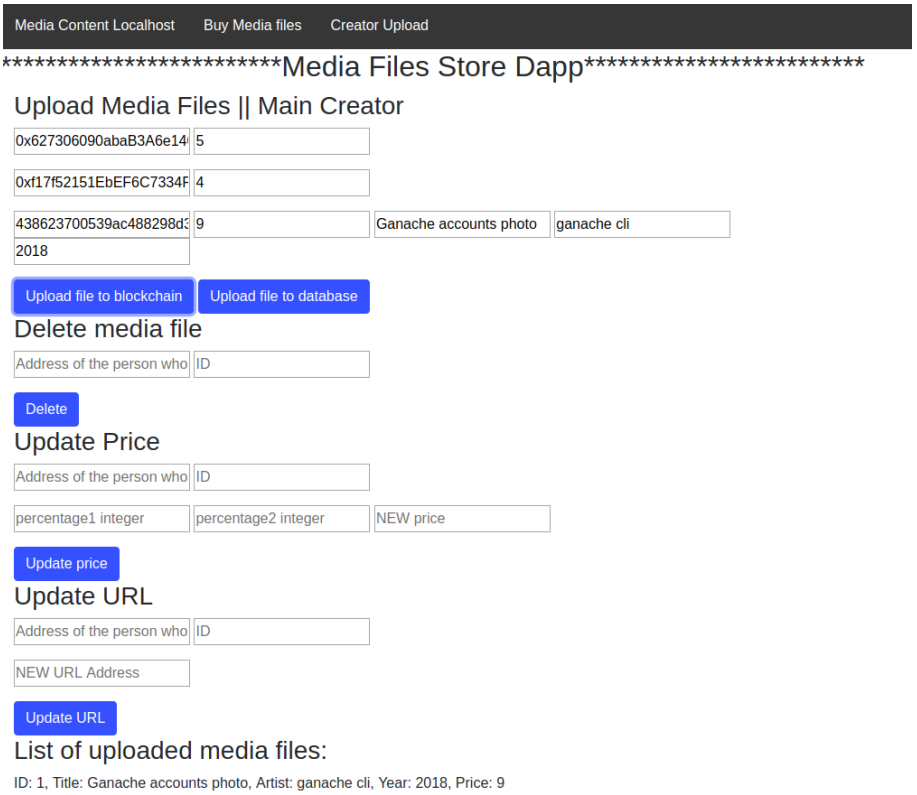
Εκεί, συμπληρώνει τα στοιχεία του αρχείου όπως θέλει αυτά να καταχωρηθούν στο ιδιωτικό δίκτυο του Blockchain. Πιο συγκεκριμένα, συμπληρώνει τις Ethereum διευθύνσεις του κάθε ένα από τους δύο κατόχους πνευματικών δικαιωμάτων του κάθε αρχείου καθώς και το ποσό που πρέπει να λάβει καθένας εξ αυτών από την κάθε αγορά του αρχείου. Ακόμα, συμπληρώνει στοιχεία που θα παρουσιαστούν στους πιθανούς αγοραστές όπως το όνομα του αρχείου, το όνομα του καλλιτέχνη, το έτος δημιουργίας του και την τελική τιμή στην οποία θα πωληθεί. Τέλος, συμπληρώνει το url στο οποίο θα είναι διαθέσιμο το αρχείο και πατάει το πλήκτρο Upload file to blockchain.



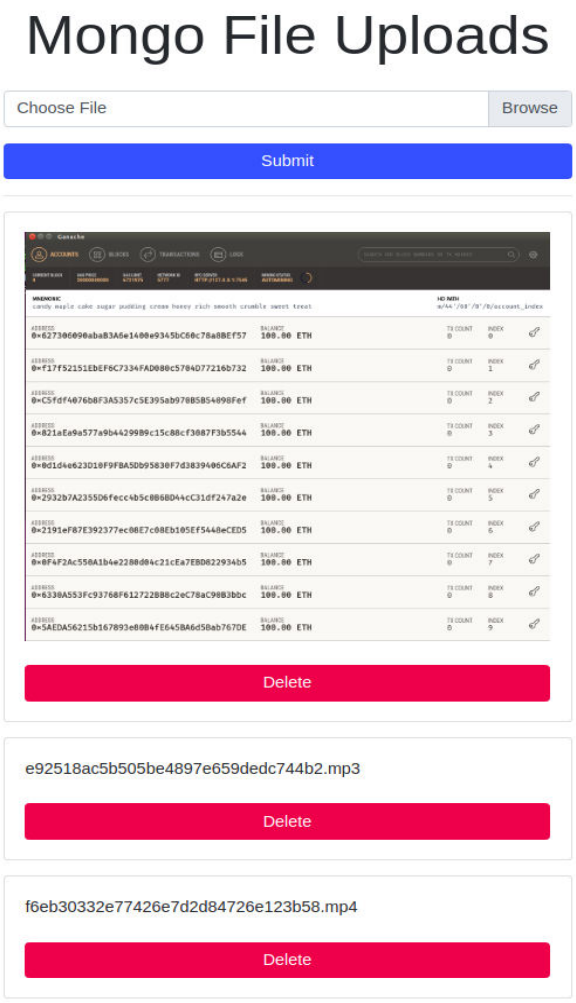
Εικόνα 6-6 Upload media file, μόλις έχουν συμπληρωθεί τα απαιτούμενα δεδομένα



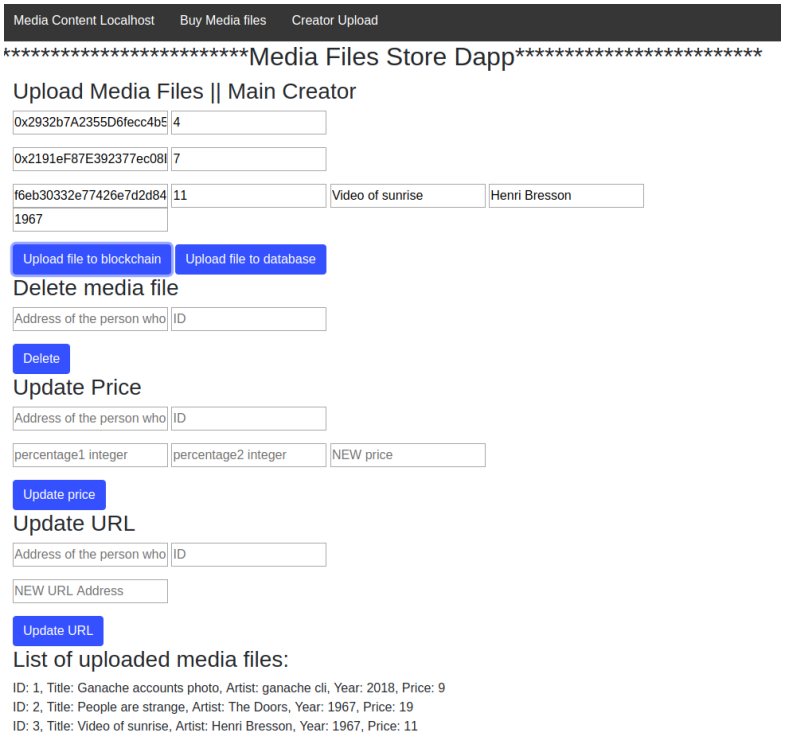
Εικόνα 6-7 Upload media file, μόλις πατηθεί το πλήκτρο Upload file to blockchain



Εικόνα 6-8 media file, μόλις αφού έχει ανεβεί το πρώτο αρχείο στο blockchain



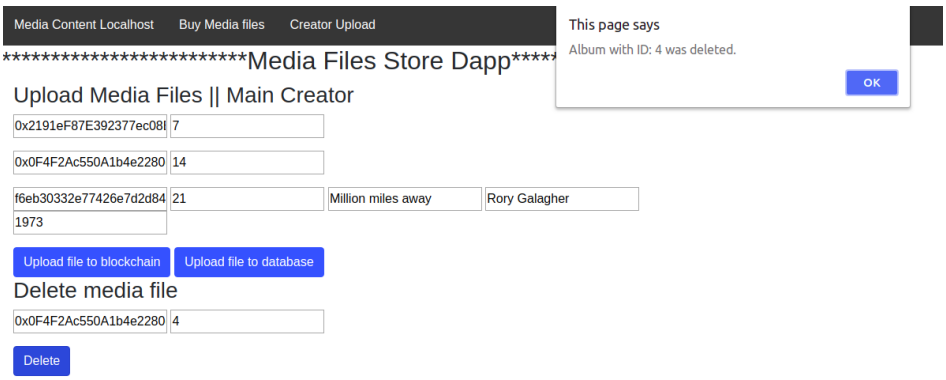
Εικόνα 6-9 Βάση δεδομένων μετά το ανέβασμα αρχείων εικόνας, ήχου και βίντεο



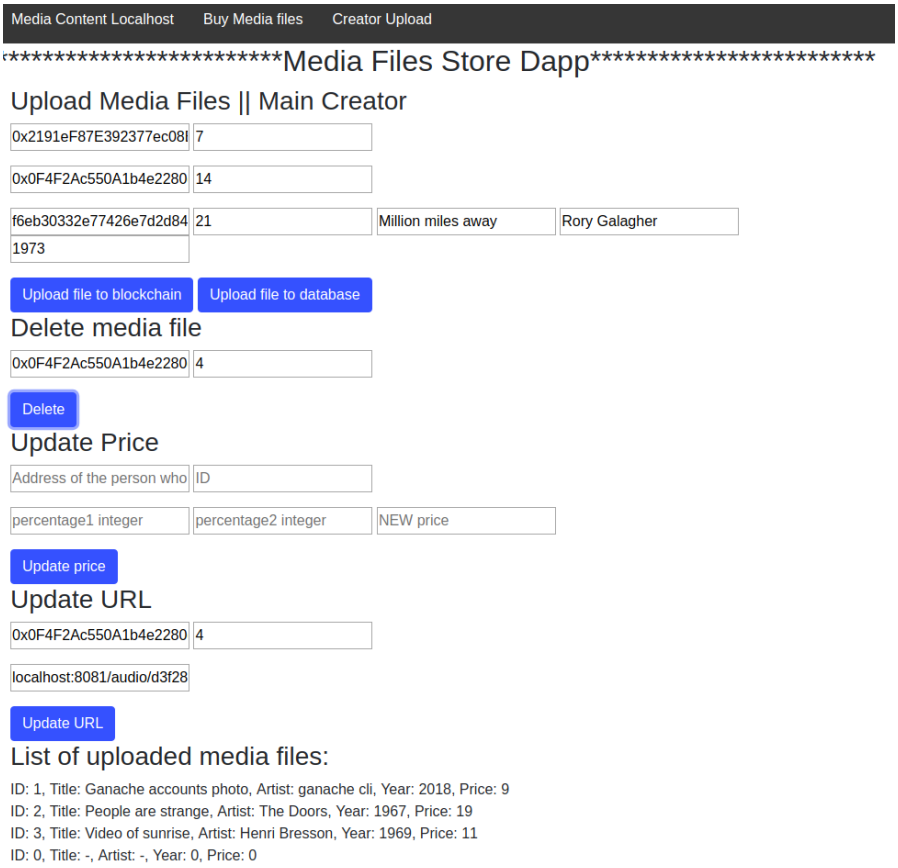
Εικόνα 6-10 Upload media file, μόλις αφού έχουν ανεβεί αρχεία εικόνας, ήχου και βίντεο στο blockchain

Λειτουργία Delete:

Η σελίδα Creator Upload προσφέρει περαιτέρω δυνατότητες στον χρήστη-ιδιοκτήτη των αρχείων. Μια από αυτές είναι η δυνατότητα να διαγράψει κάποιο αρχείο, του οποίου είναι κάτοχος πνευματικών δικαιωμάτων, από το δίκτυο του blockchain. Ο χρήστης συμπληρώνει την Ethereum διεύθυνση του και το ID του αρχείου που επιδιώκει να διαγράψει ώστε να γίνεται έλεγχος αν είναι ένας από τους κατόχους πνευματικών δικαιωμάτων του αρχείου και συνεπώς, έχει την δικαιοδοσία να προβεί στην διαγραφή.



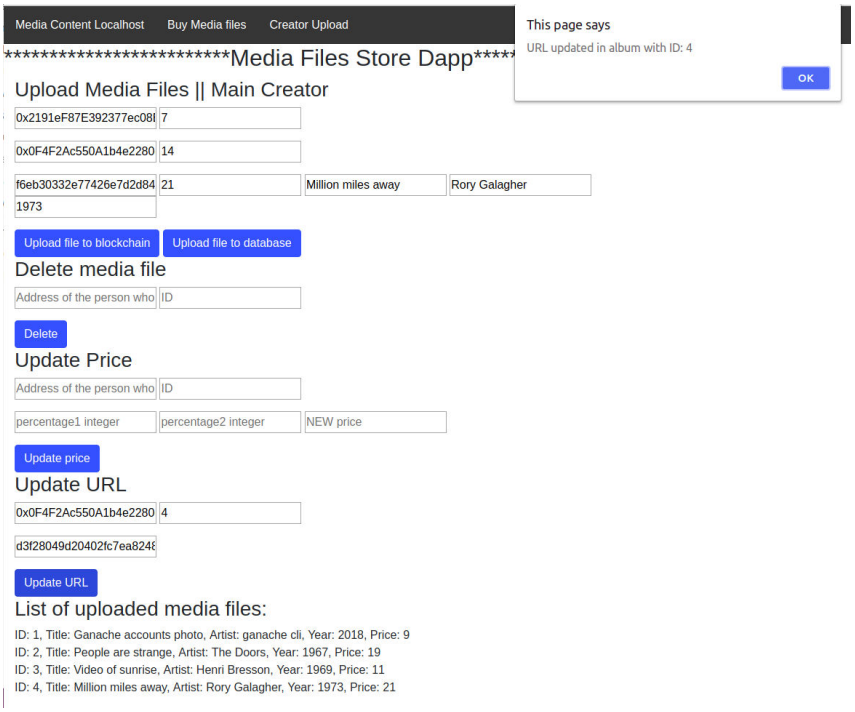
Εικόνα 6-11 Delete media file, μετά την επιτυχή διαγραφή του αρχείου



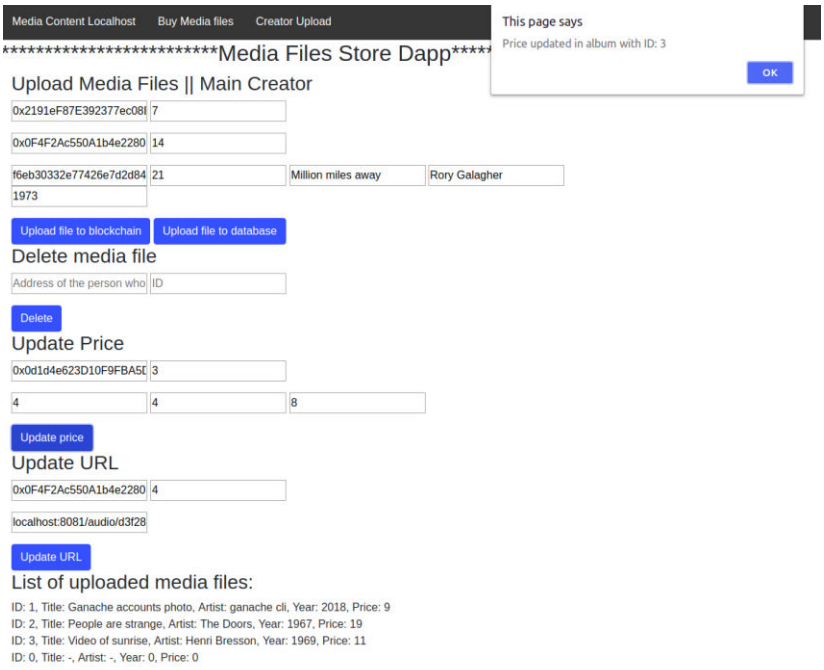
Εικόνα 6-12 List of uploaded media files after the deletion of file with ID: 4

Λειτουργίες Update URL και Update price:

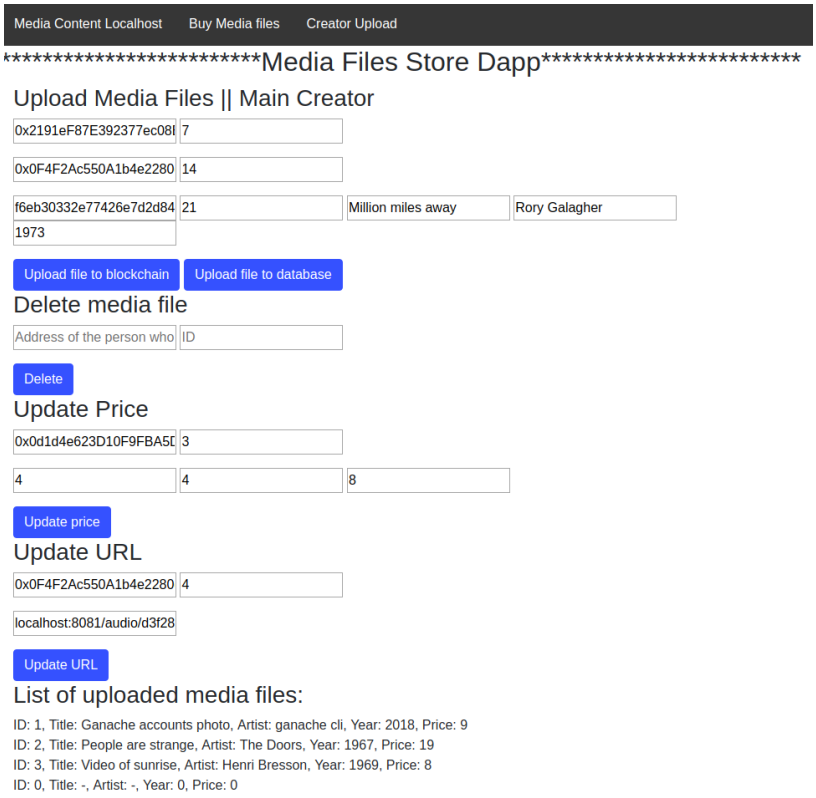
Μέσα από την σελίδα Creator Upload, οι κάτοχοι πνευματικών δικαιωμάτων έχουν την δυνατότητα να αλλάξουν το καταχωρημένο url στο οποίο είναι αποθηκευμένο το αρχείο στην βάση δεδομένων αλλά και την τελική τιμή πώλησης του αρχείου μαζί με τις επιμέρους αποδόσεις που θα λάβει ο κάθε ένας από αυτούς. Για να επιτύχουν τα παραπάνω, οι χρήστες αρχικά συμπληρώνουν τον Ethereum λογαριασμό τους και το ID του αρχείου για να επιβεβαιωθεί ότι έχουν την δικαιοδοσία να αλλάξουν δεδομένα σε αυτό. Έπειτα, συμπληρώνουν την νέα διεύθυνση url ή τις τιμές που πρέπει να πληρωθούν σε κάθε έναν από τους ιδιοκτήτες και την τελική τιμή που θα διατίθεται το αρχείο, αντίστοιχα.



Εικόνα 6-13 Update url



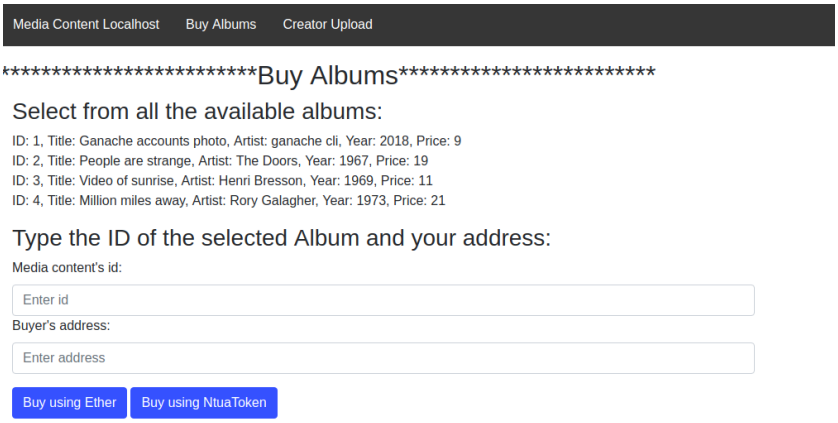
Εικόνα 6-14 Update price



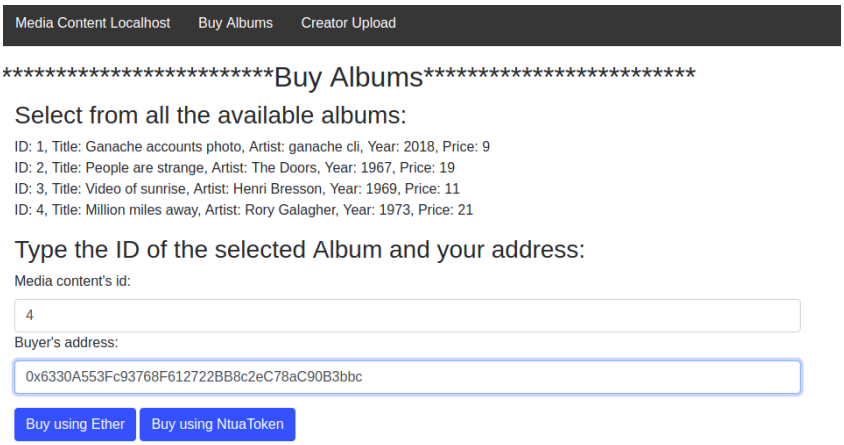
Εικόνα 6-15 Price was updated in the list of uploaded media files

Λειτουργία Buy:

Η λειτουργία της αγοράς κάποιου αρχείου πολυμέσων είναι κομβικής σημασίας για την εφαρμογή μας. Αυτή μπορεί να επιτευχθεί από την ιστοσελίδα Buy Media Content ως εξής: ο χρήστης που ενδιαφέρεται να αγοράσει κάποιο αρχείο πηγαίνει στην σελίδα και βλέπει την λίστα με τα διαθέσιμα αρχεία προς πώληση. Για την αγορά πρέπει να συμπληρώσει την Ethereum διεύθυνσή του και το ID του αρχείου που επιθυμεί να αγοράσει. Ακολούθως, διαλέγει αν θέλει να πληρώσει με Ethers ή με το κρυπτονόμισμα που αναπτύχθηκε για τις ανάγκες της εφαρμογής και πατάει το κατάλληλο πλήκτρο. Τότε θα γίνει έλεγχος αν διαθέτει το απαιτούμενο ποσό για την πληρωμή των ιδιοκτητών και αν ναι, κάνει άμεση συναλλαγή με τον καθένα εξ' αυτών και αφού εκτελεστούν οι συναλλαγές και καταγραφούν στο δίκτυο, δέχεται το url του αρχείου που αγόρασε ώστε να το προσπελάσει.

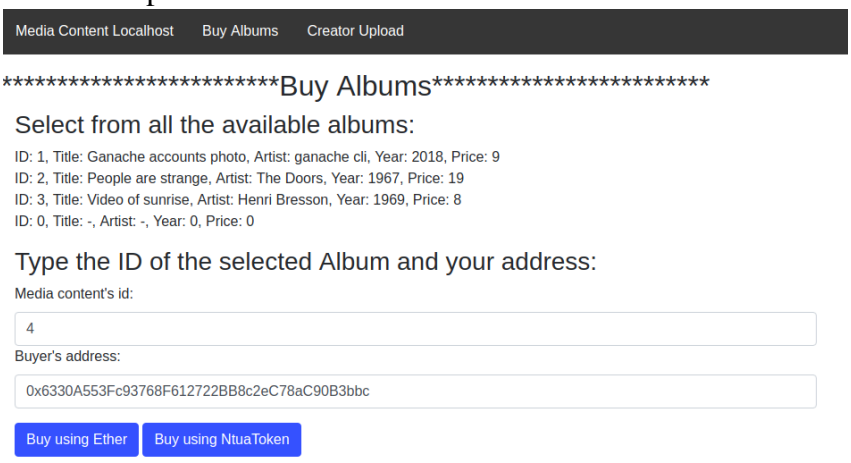


Εικόνα 6-16 Η λίστα με τα διαθέσιμα προς πώληση αρχεία και τα στοιχεία που πρέπει να συμπληρώσει ο υποψήφιος αγοραστής.

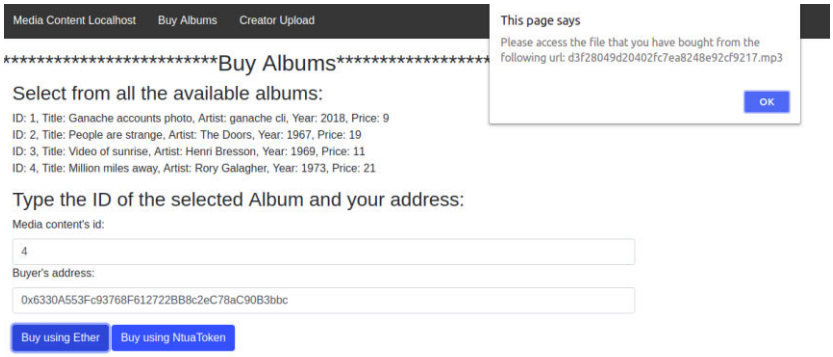


Εικόνα 6-17 Ο χρήστης συμπληρώνει τα απαιτούμενα στοιχεία

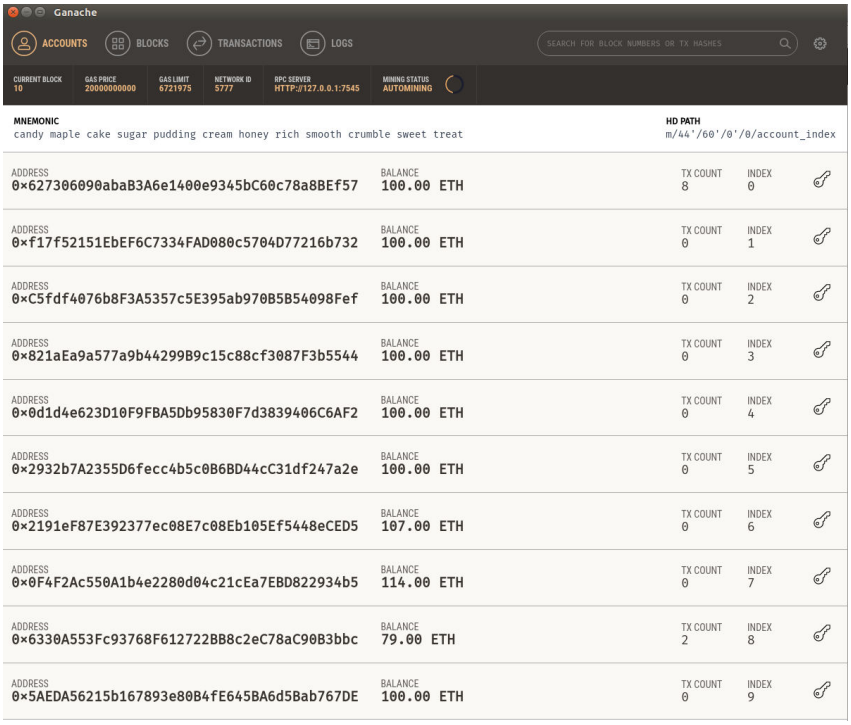
Η σελίδα αυτή ενημερώνεται αυτόματα από κάθε αλλαγή που γίνεται στην ιστοσελίδα Creator Upload.



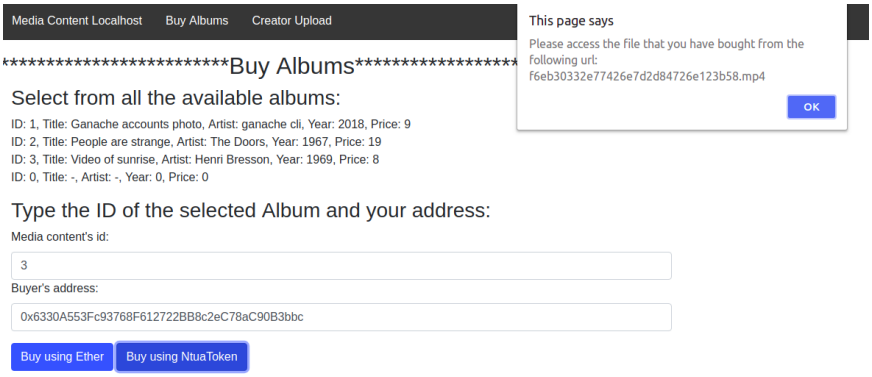
Εικόνα 6-18 Τα διαθέσιμα αρχεία προς πώληση μετά την διαγραφή ενός αρχείου από τον ιδιοκτήτη του και μετά την αλλαγή της τιμής του αρχείου με ID ίσο με 3



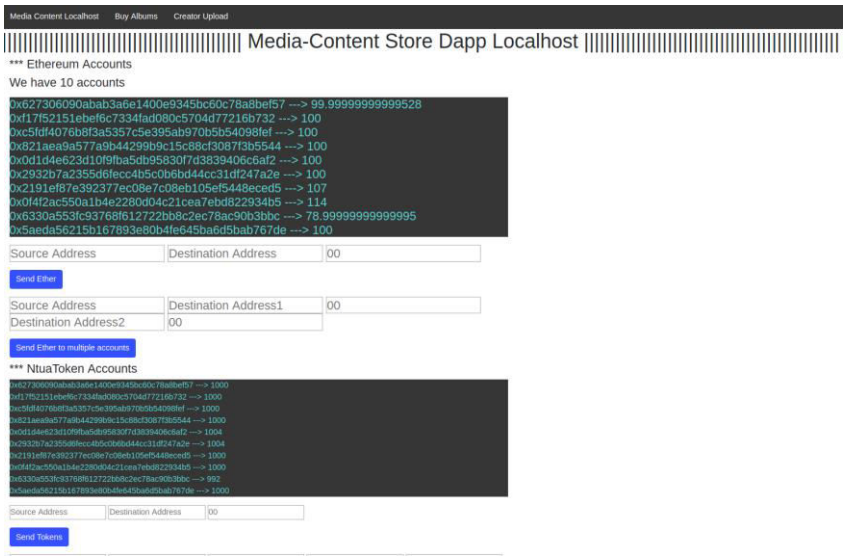
Εικόνα 6-19 Η εικόνα που βλέπει ο χρήστης που μόλις αγόρασε το αρχείο με ID: 4 πληρώνοντας σε Ethers.



Εικόνα 6-20 Τα νέα υπόλοιπα των Ethereum λογαριασμών μετά την αγορά ενός αρχείου με χρήση Ethers

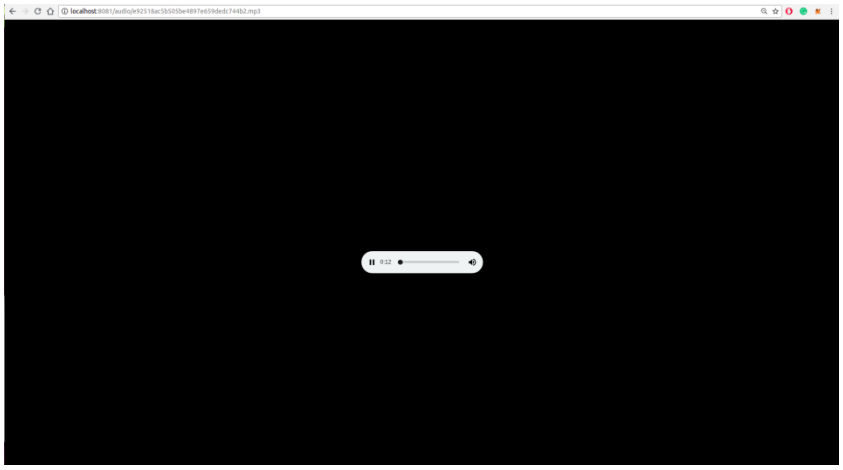


Εικόνα 6-21 Η εικόνα που βλέπει ο χρήστης που μόλις αγόρασε το αρχείο με ID: 3 πληρώνοντας σε NtuaTokens.

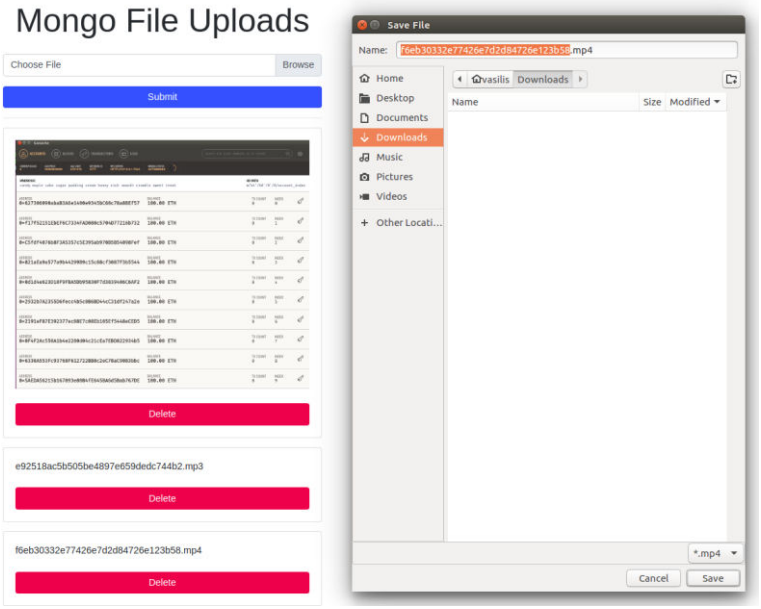


Εικόνα 6-22 Τα νέα υπόλοιπα των Ethereum λογαριασμών σε NtuaTokens και Ethers μετά την αγορά ενός αρχείου με χρήση NtuaTokens

Μόλις ολοκληρωθεί η αγορά, ο αγοραστής μπορεί να ανοίξει το url που έλαβε και να προβάλλει ή να κατεβάσει το αρχείο. Επιθυμώντας να γενικεύσω την εφαρμογή, έδωσα και τις δύο αυτές δυνατότητες.



Εικόνα 6-23 Αναπαραγωγή αρχείου ήχου με την κατοχή του url από τον αγοραστή



Εικόνα 6-24 Κατέβασμα αρχείου βίντεο με την κατοχή του url από τον αγοραστή

7

Επίλογος

7.1 Σύνοψη και συμπεράσματα

Σκοπός της διπλωματικής εργασίας ήταν η εξέταση των νέων τεχνολογιών αποκέντρωσης και η ανάπτυξη έξυπνων συμβολαίων για την λειτουργία μίας αποκεντρωμένης εφαρμογής (DApp – Decentralized Application). Έτσι, δημιουργήθηκε μια πρωτοποριακή αποκεντρωμένη εφαρμογή που αποτελεί μια πλατφόρμα αγοραπωλησιών αρχείων πολυμέσων μέσω του Ethereum blockchain.

Το Blockchain και ειδικότερα το Ethereum blockchain είναι σχετικά πολύ νέες, όχι ακόμα ώριμες και ραγδαία αναπτυσσόμενες τεχνολογίες. Αυτό έχει ως συνέπεια να υπάρχει διαρκής εξέλιξη, αλλαγή και σε κάποιες περιπτώσεις ακόμα και κατάργηση σχετικών εργαλείων αλλά και να μην υπάρχει ακόμα υποστήριξη από μια μεγάλη κοινότητα προγραμματιστών που να ασχολούνται με τον κλάδο των αποκεντρωμένων εφαρμογών.

Εξετάστηκαν πολλές διαφορετικές τεχνολογίες για την υλοποίηση της συγκεκριμένης εφαρμογής με στόχο να βρεθεί η καταλληλότερη. Πιο συγκεκριμένα, μελετήθηκε ο πηγαίος κώδικας του bitcoin [67], η σουίτα προγραμμάτων Truffle [68], καθώς και η δημιουργία ιδιωτικού δικτύου Ethereum blockchain με χρήση του geth [41] και εγκατάστασή του σε φυσικά και εικονικά μηχανήματα που το καθένα από αυτά αποτελούσε διαφορετικό κόμβο του δικτύου. Τελικά, επιλέχθηκε η αποκεντρωμένη πλατφόρμα του Ethereum καθώς μετά από αξιολόγηση των διαφορετικών επιλογών, κρίθηκε ότι το Ethereum είναι η καταλληλότερη τεχνολογία για τον σκοπό μας. Αυτό διότι, είναι ένα από τα πιο διαδεδομένα blockchain, πιο συγκεκριμένα είναι δεύτερο μετά από αυτό του Bitcoin και είναι το μόνο οικοσύστημα που συνδυάζει την δημοφιλία και την αξιοπιστία με την δυνατότητα ανάπτυξης έξυπνων συμβολαίων. Επίσης, στο Ethereum προσφέρονται τα περισσότερα και πιο πλήρη εργαλεία για την ανάπτυξη και λειτουργία αποκεντρωμένων εφαρμογών.

7.2 Μελλοντικές επεκτάσεις

Στην ενότητα αυτή θα παρουσιαστούν μερικές μελλοντικές επεκτάσεις της παρούσας εφαρμογής. Σίγουρα, ρόλο παίζει και η εξέλιξη των τεχνολογιών που χρησιμοποιήθηκαν, που όπως όλα δείχνουν θα είναι ραγδαία στο άμεσο μέλλον. Ο λόγος είναι ότι το blockchain και οι αποκεντρωμένες εφαρμογές έχουν την δυναμική

να αλλάξουν σε μεγάλο βαθμό αρκετούς τομείς της ζωής και συνεπώς έχουν τραβήξει το ενδιαφέρον πολλών ερευνητών, προγραμματιστών, εταιρειών και επενδυτών.

Αρχικά, θα μπορούσε να γίνει μια διάκριση ανάμεσα στο κατέβασμα του αρχείου από τον χρήστη και στην προβολή του. Αυτό θα είχε ως συνέπεια να υπάρχει μια διαφορετική τιμή για την απλή προβολή μιας εικόνας ή ενός βίντεο ή την απλή ακρόαση ενός τραγουδιού από την απόκτηση του αρχείου για πολλαπλές αναπαραγωγές. Με αυτό τον τρόπο θα μπορούσε για παράδειγμα να χρεώνεται μια μικρή τιμή για την αναπαραγωγή ενός τραγουδιού για εμπορική χρήση (π.χ. σε εμπορικά καταστήματα) και να πληρώνονται απευθείας οι κάτοχοι των πνευματικών δικαιωμάτων του. Το παραπάνω είναι μια πιθανή εναλλακτική της σημερινής αντιμετώπισης που είναι η πληρωμή συγκεκριμένου ποσού ετησίως από τα εμπορικά καταστήματα σε μεσάζοντες, που είναι συνήθως εταιρείες πνευματικών δικαιωμάτων.

Μια αλλαγή όσον αφορά την υλοποίηση θα μπορούσε να είναι η χρήση του αποκεντρωμένου συστήματος IPFS (InterPlanetary File System) [69] [70] αντί της βάσης δεδομένων που χρησιμοποιήθηκε για το ανέβασμα των αρχείων. Έτσι, το σύστημά μας θα ήταν στο σύνολό του αποκεντρωμένο, μιας και το IPFS είναι μια μέθοδος κατανεμημένης αποθήκευσης αρχείων.

Εναλλακτικά, αν θελήσουμε να επεκτείνουμε την παρούσα εφαρμογή με την υφιστάμενη αρχιτεκτονική, θα μπορούσαμε να δημιουργήσουμε και έναν πίνακα στην βάση δεδομένων στον οποίο θα αποθηκεύονται οι διευθύνσεις url για κάθε αρχείο με βάση το id του. Έτσι, αντί να είναι αποθηκευμένη η διεύθυνση στο blockchain, θα μπορούσαμε κάθε φορά που αγοράζεται κάποιο αρχείο να ανατρέχουμε στην βάση και να παίρνουμε το url για να το δώσουμε στον αγοραστή. Μια επέκταση του παραπάνω είναι να παράγεται μοναδική διεύθυνση για κάθε αγορά και αυτή να κρυπτογραφείται με το δημόσιο Ethereum κλειδί του αγοραστή. Κατ' αυτόν τον τρόπο, μόνο αυτός θα μπορούσε να την αποκρυπτογραφήσει με το ιδιωτικό του κλειδί Ethereum.

Ακόμα, θα μπορούσε να γίνει μια διεύρυνση του πεδίου της εφαρμογής. Δεδομένου ότι τα έξυπνα συμβόλαια έχουν αναπτυχθεί ώστε να είναι όσο το δυνατόν πιο ουδέτερα όσον αφορά τον τύπο του αρχείου που θα πωλείται μέσω της εφαρμογής, μπορούμε να επεκτείνουμε την πλατφόρμα ώστε σε αυτήν να γίνεται επιπρόσθετα και αγοραπωλησία πληροφορίας. Για παράδειγμα, θα μπορούσε να αποτελέσει εργαλείο ώστε οι ιδιοκτήτες αισθητήρων να πωλούν τις μετρήσεις των αισθητήρων σε πραγματικό χρόνο για χρήση από συσκευές που ανήκουν στο διαδίκτυο των πραγμάτων (Internet of Things – IoT). Οι ιδιοκτήτες των συσκευών που χρειάζονται τις μετρήσεις μπορούν να θέτουν κάποιες παραμέτρους και οι συσκευές να έχουν έναν Ethereum λογαριασμό και να αγοράζουν μόνες τους τις επιθυμητές μετρήσεις. Το τελευταίο, θα μπορούσε να έχει εφαρμογή στο άμεσο μέλλον στα αυτοκινούμενα αυτοκίνητα (driverless cars) που όπως όλα δείχνουν θα είναι ευρέως χρησιμοποιούμενα στο μέλλον. Πιο συγκεκριμένα, οι ιδιοκτήτες θα ήταν χρήσιμο να επιλέγουν τον προορισμό που τους ενδιαφέρει και τα ίδια τα αυτοκίνητα (μέσω των συστημάτων τους) να χρειάζονται μετρήσεις από ιδιόκτητους IoT αισθητήρες για τον υπολογισμό της βέλτιστης διαδρομής, τις οποίες θα αγοράζουν μέσω του παρόντος συστήματος.

Τέλος, μια ακόμα πρόταση θα ήταν να γίνει η κατάλληλη αλλαγή στο έξυπνο συμβόλαιο NtuaToken και με την εκτέλεσή του να δίνονται κάποια NtuaTokens στην διεύθυνση του συμβολαίου. Έτσι, οι χρήστες θα μπορούν να αγοράζουν από το συμβόλαιο NtuaTokens πληρώνοντας σε Ethers το ποσό που αναλογεί στην

καθορισμένη ισοτιμία NtuaToken-Ether. Αντίστοιχα, θα μπορούν να πωλούν πάλι πίσω τα NtuaTokens ώστε να πάρουν Ethers.

8

Βιβλιογραφία

w. e. forum, «Beyond bitcoin: 4 surprising uses for blockchain,» 13 12 2016. [Ηλεκτρονικό]. Available: https://www.weforum.org/agenda/2016/12/fighting-human-trafficking-tracing-blood-diamonds-and-other-surprising-uses-for-blockchain/. [Πρόσβαση 14 6 2018].

I. B. Times, «Microsoft Building Blockchain Identity System To Fight Human Trafficking, Prostitution And Child Abuse,» 1 6 2016. [Ηλεκτρονικό]. Available: http://www.ibtimes.com/microsoft-building-blockchain-identity-system-fight-human-trafficking-prostitution-2376580. [Πρόσβαση 14 6 2018].

«UNICEF TURNS TO CRYPTOCURRENCY MINING TO RAISE MONEY FOR REFUGEES,» [Ηλεκτρονικό]. Available: https://www.independent.co.uk/life-style/gadgets-and-tech/news/unicef-cryptocurrency-fundraising-child-refugees-digital-bitcoin-monero-a8332726.html. [Πρόσβαση 18 6 2018].

«UNICEF Is Mining Crypto to Raise Funds for Children,» [Ηλεκτρονικό]. Available: https://www.coindesk.com/unicef-taps-in-browser-mining-to-raise-funds-for-children-in-need/. [Πρόσβαση 18 6 2018].

«Hopepage,» [Ηλεκτρονικό]. Available: https://www.thehopepage.org/. [Πρόσβαση 18 6 2018].

Wikipedia, «ΑΕΠΙ,» [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/%CE%91%CE%95%CE%A0%CE%99. [Πρόσβαση 14 6 2018].

c. greece, «Οριστικό «λουκέτο» στην ΑΕΠΙ,» 15 5 2018. [Ηλεκτρονικό]. Available: http://www.cnn.gr/news/ellada/story/130110/oristiko-loyketo-stin-aepi. [Πρόσβαση 14 6 2018].

«What is the Web3? The Decentralized Web - Blockchain,» [Ηλεκτρονικό]. Available: https://blockchainhub.net/web3-decentralized-web/. [Πρόσβαση 12 6 2018].

B. Pon, «Blockchain will usher the era of decentralized computing,» 15 4 2016. [Ηλεκτρονικό]. Available: https://blog.bigchaindb.com/blockchain-will-usher-in-the-era-of-decentralised-computing-7f35e94af0b6. [Πρόσβαση 12 6

2018].

«Bigchain DB The scalable blockchain database,» [Ηλεκτρονικό]. Available: https://www.bigchaindb.com/. [Πρόσβαση 14 6 2018].
10]

J. Kurose και K. Ross, Δικτύωση Υπολογιστών, Αθήνα: Μ. Γκιούρδας.
11]

J. Ray, «Ethereum introduction,» [Ηλεκτρονικό]. Available: https://github.com/ethereum/wiki/wiki/Ethereum-introduction. [Πρόσβαση 14 6 2018].
12]

«Elliptic Curve Cryptography (ECC),» [Ηλεκτρονικό]. Available: https://www.certicom.com/content/certicom/en/ecc.html. [Πρόσβαση 10 6 2018].
13]

L. C. WASHINGTON, Elliptic Curves Number Theory and Cryptography, Maryland: Chapman & Hall, 2008.
14]

V. Buterin, «Exploring Elliptic Curve Pairings,» [Ηλεκτρονικό]. Available: https://medium.com/@VitalikButerin/exploring-elliptic-curve-pairings-c73c1864e627. [Πρόσβαση 10 6 2018].
15]

A. S. Tanenbaum και M. v. Steen, Κατανεμημένα Συστήματα, Αθήνα: Κλειδάριθμος, 2005, p. 26.
16]

S. Nakamoto, «Bitcoin: A Peer-to-Peer Electronic Cash System,» 2008. [Ηλεκτρονικό]. Available: https://bitcoin.org/bitcoin.pdf. [Πρόσβαση 14 6 2018].
17]

«What is Ethereum?,» [Ηλεκτρονικό]. Available: http://www.ethdocs.org/en/latest/introduction/what-is-ethereum.html. [Πρόσβαση 9 6 2018].
18]

waygie, «White Paper,» [Ηλεκτρονικό]. Available: https://github.com/ethereum/wiki/wiki/White-Paper#ethereum. [Πρόσβαση 14 6 2018].
19]

«Proof of work, bitcoinwiki,» [Ηλεκτρονικό]. Available: https://en.bitcoin.it/wiki/Proof_of_work. [Πρόσβαση 14 6 2018].
20]

«Proof-of-work system,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Proof-of-work_system. [Πρόσβαση 14 6 2018].
21]

«Proof-of-stake,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Proof-of-stake. [Πρόσβαση 14 6 2018].
22]

V. Buterin, «On Stake,» 5 7 2014. [Ηλεκτρονικό]. Available: https://blog.ethereum.org/2014/07/05/stake/. [Πρόσβαση 14 6 2018].
23]

F. Lange, «ÐΞVp2p Wire Protocol,» [Ηλεκτρονικό]. Available: https://github.com/ethereum/wiki/wiki/%C3%90%CE%9EVp2p-Wire-Protocol. [Πρόσβαση 14 6 2018].
24]

N. Savers, «Ethereum Wire Protocol,» [Ηλεκτρονικό]. Available: https://github.com/ethereum/wiki/wiki/Ethereum-Wire-Protocol. [Πρόσβαση 14
25]

6 2018].

G. Wood, «Ethereum Yellow Paper: a formal specification of Ethereum, a
26] programmable blockchain,» 7 2 2018. [Ηλεκτρονικό]. Available: https://ethereum.github.io/yellowpaper/paper.pdf. [Πρόσβαση 14 6 2018].

M. C. A. M. Andreas Bogner, «A Decentralised Sharing App running a
27] Smart Contract on the Ethereum Blockchain,» 2016. [Ηλεκτρονικό]. Available: https://dl.acm.org/citation.cfm?id=2998465. [Πρόσβαση 12 6 2018].

L. F. H. Z. Y. Y. L. W. Y. Z. Wei-Tek Tsai, «Intellectual-Property
28] Blockchain-Based Protection Model for Microfilms,» 2017. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/7943309/. [Πρόσβαση 14 6 2018].

A. d. B. H. R. H. A. Thomas Lundqvist, «Thing-to-thing electricity micro
29] payments using blockchain technology,» 2017. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/8016254/. [Πρόσβαση 12 6 2018].

«Proof of concept,» [Ηλεκτρονικό]. Available:
30] https://en.wikipedia.org/wiki/Proof_of_concept. [Πρόσβαση 14 6 2018].

F.-Y. W. Yong Yuan, «Towards blockchain-based intelligent transportation
31] systems,» 2016. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/7795984/. [Πρόσβαση 14 6 2018].

«Uber wikipedia,» [Ηλεκτρονικό]. Available:
32] https://en.wikipedia.org/wiki/Uber. [Πρόσβαση 18 6 2018].

H. W. A. N. T. Y. A. A. J. J. K. Shigeru Fujimura, «BRIGHT: A concept
33] for a decentralized rights management system based on blockchain,» 2015. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/7391275/. [Πρόσβαση 14 6 2018].

O. N. A. '. P. Guy Zyskind, «Decentralizing Privacy: Using Blockchain to
34] Protect Personal Data,» 2015. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/7163223/. [Πρόσβαση 14 6 2018].

G. O. Richard Dennis, «Rep on the block: A next generation reputation
35] system based on the blockchain,» 2015. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/7412073/. [Πρόσβαση 14 6 2018].

«Ujo music,» [Ηλεκτρονικό]. Available: https://ujomusic.com/.
36] [Πρόσβαση 15 6 2018].

«Bloomen,» [Ηλεκτρονικό]. Available: www.bloomen.io. [Πρόσβαση 22 6
37] 2018].

A. L. G. F. V. P. T. V. Georgios Palaiokrassas, «Deploying blockchains for
38] a new paradigm of media experience,» σε *15th International Conference on the Economics of Grids, Clouds, Systems, and Services (GECON 2018)*, Pisa, Italy, 2018.

«GECON 2018,» [Ηλεκτρονικό]. Available: http://2018.gecon-conference.org/index.php/program. [Πρόσβαση 25 6 2018].
39]

«Prince Warns Young Artists: Record Contracts Are 'Slavery',» 9 8 2015. [Ηλεκτρονικό]. Available: https://www.rollingstone.com/music/news/prince-warns-young-artists-record-contracts-are-slavery-20150809. [Πρόσβαση 14 6 2018].
40]

F. Lange, «Geth * ethereum/go-ethereum Wiki,» [Ηλεκτρονικό]. Available: https://github.com/ethereum/go-ethereum/wiki/geth. [Πρόσβαση 14 6 2018].
41]

«Ganache CLI,» [Ηλεκτρονικό]. Available: https://github.com/trufflesuite/ganache-cli. [Πρόσβαση 13 6 2018].
42]

«ethereumjs by ethereumjs,» [Ηλεκτρονικό]. Available: http://ethereumjs.github.io/. [Πρόσβαση 14 6 2018].
43]

«Web3.js,» [Ηλεκτρονικό]. Available: https://github.com/ethereum/web3.js/. [Πρόσβαση 14 6 2018].
44]

«Solidity documentation,» [Ηλεκτρονικό]. Available: https://solidity.readthedocs.io/en/develop/. [Πρόσβαση 14 6 2018].
45]

«Remix - Solidity IDE,» [Ηλεκτρονικό]. Available: http://remix.readthedocs.io/en/latest/. [Πρόσβαση 14 6 2018].
46]

«Remix -Solidity IDE,» [Ηλεκτρονικό]. Available: https://remix.ethereum.org. [Πρόσβαση 14 6 2018].
47]

«MetaMask,» [Ηλεκτρονικό]. Available: https://metamask.io/. [Πρόσβαση 14 6 2018].
48]

«What is npm?,» [Ηλεκτρονικό]. Available: https://docs.npmjs.com/getting-started/what-is-npm. [Πρόσβαση 8 6 2018].
49]

«Docs | Node.js,» [Ηλεκτρονικό]. Available: https://nodejs.org/en/docs/. [Πρόσβαση 10 6 2018].
50]

«Node.js Introduction,» [Ηλεκτρονικό]. Available: https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm. [Πρόσβαση 10 6 2018].
51]

«Chrome V8 | Google Developers,» [Ηλεκτρονικό]. Available: https://developers.google.com/v8/. [Πρόσβαση 10 6 2018].
52]

«Bootstrap,» [Ηλεκτρονικό]. Available: https://getbootstrap.com/docs/4.0/getting-started/introduction/. [Πρόσβαση 6 6 2018].
53]

«Bootstrap Get Started,» [Ηλεκτρονικό]. Available: https://www.w3schools.com/bootstrap/bootstrap_get_started.asp. [Πρόσβαση 6 6 2018].
54]

«JavaScript,» [Ηλεκτρονικό]. Available: https://www.javascript.com/. [Πρόσβαση 6 16 2018].
55]

«JavaScript Overview,» [Ηλεκτρονικό]. Available: https://www.tutorialspoint.com/javascript/javascript_overview.htm. [Πρόσβαση 14 6 2018].
56]

«jQuery API Documentation,» [Ηλεκτρονικό]. Available: https://api.jquery.com/. [Πρόσβαση 7 6 2018].
57]

«jQuery Introduction,» [Ηλεκτρονικό]. Available: https://www.w3schools.com/Jquery/jquery_intro.asp. [Πρόσβαση 7 6 2018].
58]

«MongoDB,» [Ηλεκτρονικό]. Available: https://www.mongodb.com/. [Πρόσβαση 14 6 2018].
59]

«NoSQL Wikipedia,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/NoSQL. [Πρόσβαση 14 6 2018].
60]

«GridFS,» [Ηλεκτρονικό]. Available: https://docs.mongodb.com/manual/core/gridfs/. [Πρόσβαση 14 6 2018].
61]

«Error handling: Assert, Require, Revert and Exceptions,» [Ηλεκτρονικό]. Available: http://solidity.readthedocs.io/en/v0.4.24/control-structures.html#error-handling-assert-require-revert-and-exceptions. [Πρόσβαση 18 6 2018].
62]

A. Gou, «RLP,» 19 December 2017. [Ηλεκτρονικό]. Available: https://github.com/ethereum/wiki/wiki/RLP. [Πρόσβαση 6 6 2018].
63]

Team Keccak, «Keccak,» Team Keccak, [Ηλεκτρονικό]. Available: https://keccak.team/keccak.html. [Πρόσβαση 6 6 2018].
64]

«How to upload files on Blockchain?,» 17 2 2018. [Ηλεκτρονικό]. Available: https://www.blockchainsemantics.com/blog/files-on-blockchain/. [Πρόσβαση 14 6 2018].
65]

«Node.js v10.4.1 Documentation,» [Ηλεκτρονικό]. Available: https://nodejs.org/api/crypto.html#crypto_crypto_randombytes_size_callback. [Πρόσβαση 14 6 2018].
66]

«bitcoin code github,» [Ηλεκτρονικό]. Available: https://github.com/bitcoin/bitcoin. [Πρόσβαση 15 6 2018].
67]

«truffle framework,» [Ηλεκτρονικό]. Available: http://truffleframework.com/. [Πρόσβαση 15 6 2018].
68]

«InterPlanetary File System,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/InterPlanetary_File_System. [Πρόσβαση 15 6 2018].
69]

«IPFS is the Distributed Web,» [Ηλεκτρονικό]. Available: https://ipfs.io/. [Πρόσβαση 15 6 2018].
70]

«Application Binary Interface Specification,» [Ηλεκτρονικό]. Available:

71] https://solidity.readthedocs.io/en/develop/abi-spec.html. [Πρόσβαση 18 6 2018].

72] Ethereum, «Application Binary Interface Specification,» Ethereum, 2017. [Ηλεκτρονικό]. Available: https://solidity.readthedocs.io/en/develop/abi-spec.html. [Πρόσβαση 12 2 2018].

73] «A Next-Generation Smart Contract and Decentralized Application Platform, Ethereum White Paper,» [Ηλεκτρονικό]. Available: https://github.com/ethereum/wiki/wiki/White-Paper. [Πρόσβαση 14 6 2018].

Παράρτημα Ι: Εγχειρίδιο χρήσης

Στο παράρτημα αυτό θα γίνει παρουσίαση οδηγιών για την εγκατάσταση της εφαρμογής σε λειτουργικό σύστημα Unix και συγκεκριμένα Linux Ubuntu 16.04 LTS

1 Εγκατάσταση

Εδώ παρουσιάζονται αναλυτικά τα προγράμματα που χρειάζονται για την εγκατάσταση και λειτουργία της εφαρμογής.

Βήμα 1

Εγκαθιστούμε το NodeJS και το npm γράφοντας τις παρακάτω γραμμές κώδικα σε ένα terminal:

```
sudo apt-get install nodejs
sudo apt-get install npm
```

Βήμα 2

Εγκαθιστούμε το git γράφοντας την παρακάτω εντολή κώδικα σε ένα terminal:

```
sudo apt-get install git
```

Βήμα 3

Εγκαθιστούμε το web3 γράφοντας την παρακάτω εντολή κώδικα σε ένα terminal:

```
sudo npm install web3
```

Βήμα 4

Κατεβάζουμε το ganache από την ιστοσελίδα:
www.truffleframework.com/ganache

Βήμα 5

Εγκατάσταση της βάσης δεδομένων MongoDB και του GridFS γράφοντας τις παρακάτω γραμμές κώδικα σε ένα terminal:

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 2930ADAE8CAF5059EE73BB4B58712A2291FA4AD5
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.6 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.6.list
sudo apt-get update
sudo apt-get install -y mongodb-org
```

```
sudo npm install gridfs
```

Βήμα 6

Θα πρέπει να ληφθεί ο client για την εκτέλεση του κόμβου του Ethereum blockchain. Πιο συγκεκριμένα χρησιμοποιήθηκε το geth, το command line interface υλοποιημένο στην γλώσσα προγραμματισμού Go. Ο installer του geth βρίσκεται εδώ: https://geth.ethereum.org/downloads/.

Να σημειωθεί ότι αφού ληφθεί και εγκατασταθεί ο client θα πρέπει να συγχρονιστεί με το υπόλοιπο Ethereum δίκτυο. Η διαδικασία αυτή θα καθυστερήσει, αφού πρέπει ο κόμβος να λάβει όλα τα παρελθόντα μπλοκ του blockchain (η ένα μέρος του αν επιλεχθεί γρήγορος συγχρονισμός) από το υπόλοιπο δίκτυο.

Για απλό έλεγχο λειτουργίας της εφαρμογής σε ένα ιδιωτικό Ethereum blockchain αρκεί η λήψη του ganache-cli ως npm πακέτου με την παρακάτω εντολή:

```
npm install ganache-cli
```

Βήμα 7

Εγκαθιστούμε το Metamask plugin του φυλλομετρητή μας. Το Metamask είναι διαθέσιμο στους φυλλομετρητές: Chrome, Firefox και Opera. Για τον Google Chrome είναι διαθέσιμο στο παρακάτω link:
https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn?hl=en.

2 Χρήση εφαρμογής

Βήμα 1

Εκκίνηση της βάσης δεδομένων MongoDB:

```
sudo service mongod start
mongo
```

Βήμα 2

Ανοίγω terminal πάω στον φάκελο που έχω εγκαταστήσει το ganache και το ξεκινώ. Αν δεν τρέχει πατώντας ./[όνομα αρχείου ganache], πατάμε δεξί κλικ στο appImage αρχείο -> properties-> permissions -> Allow executing file as program

Βήμα 3

Επισκεπτόμαστε την σελίδα http://remix.ethereum.org και εκεί κάνουμε τις παρακάτω ενέργειες:

Δημιουργούμε ένα αρχείο με όνομα Uploader.sol και επικολλούμε εκεί τον κώδικα από το ομώνυμο έξυπνο συμβόλαιο που δημιουργήθηκε στα πλαίσια της

παρούσας διπλωματικής εργασίας. Αντίστοιχα δημιουργούμε ένα αρχείο με όνομα NtuaToken.sol και επικολλούμε το περιεχόμενο του ομώνυμου κώδικα.

Πηγαίνουμε στην καρτέλα Run και στο Environment επιλέγουμε το Web3 Provider->OK-> http://localhost:7545->OK

Βήμα 4

Στην καρτέλα Run επιλέγουμε το έξυπνο συμβόλαιο NtuaToken.sol και πατάμε deploy. Ομοίως εκτελούμε και για το συμβόλαιο Uploader.sol.

Βήμα 5

Αντιγράφουμε τις διευθύνσεις των δύο συμβολαίων που φαίνονται από την καρτέλα Run μετά το deployment τους αλλά και τα ABI [71] του κάθε συμβολαίου που βρίσκονται στην καρτέλα Compile->Details. Τα παραπάνω τα αντικαθιστούμε στα κατάλληλα σημεία των ιστοσελίδων indexcreator.html, indexbuy.html και indexproject.html.

Βήμα 6

Σώζουμε τις αλλαγές στα αρχεία των ιστοσελίδων και ανοίγουμε τις 3 ιστοσελίδες πατώντας με διπλό κλικ πάνω τους.

Βήμα 7

Συνδεόμαστε στο metamask που έχουμε ήδη εγκαταστήσει στον φυλλομετρητή μας. Πάνω αριστερά επιλέγουμε Custom RPC και πατάμε Import Existing DEN. Αντιγράφουμε το Mnemonic που υπάρχει στο ganache και το επικολλούμε στο metamask στο πεδίο που μας ζητά το Wallet Seed. Έπειτα ο χρήστης προσθέτει τον Ethereum λογαριασμό του στο Metamask και είναι έτοιμος να χρησιμοποιήσει την εφαρμογή.

Βήμα 8

Ανοίγουμε ένα terminal και πάμε στον φάκελο που έχουμε αποθηκευμένο το αρχείο app.js για την βάση δεδομένων. Εκεί πατάμε:

```
code .
```

Και εκκινείται το visual studio code. Πατάμε View-> Integrated terminal και πατάμε:

```
node app.js
```

Έτσι εκκινείται το NodeJS και ενημερωνόμαστε ότι μπορούμε να δούμε την ιστοσελίδα της βάσης δεδομένων στο localhost:8081
Επομένως, σε μία νέα καρτέλα του φυλλομετρητή μας πηγαίνουμε στην διεύθυνση: https://localhost:8081

Παράρτημα ΙΙ: Κώδικες

Στο παράρτημα αυτό βρίσκεται ο κώδικας των δύο έξυπνων συμβολαίων που αναπτύχθηκαν στα πλαίσια της παρούσας διπλωματικής εργασίας καθώς και ο κώδικας των τριών ιστοσελίδων. Ο κώδικας για την βάση δεδομένων αλλά και το σύνολο των αρχείων κώδικα στις πιο επικαιροποιημένες εκδόσεις τους θα βρίσκονται στην σελίδα: https://github.com/vasilispapaefthymiou/Thesis-project-ECE-NTUA

1 Έξυπνα συμβόλαια

Σε αυτήν την ενότητα βρίσκεται ο κώδικας των έξυπνων συμβολαίων (smart contracts) γραμμένος στην γλώσσα Solidity.

Ο πηγαίος κώδικας του συμβολαίου NtuaToken:

```
pragma solidity ^0.4.18;

contract NtuaToken {

      //------Events------
      event Transfer(address indexed _from, address indexed _to, uint256
_value);

      //This creates an array with all balances
      mapping(address => uint256) public balanceOf;

      //Public variables of the NtuaToken
      string public name;
      string public symbol;
      uint8 public decimals;
      uint256 public totalSupply;

//Initializes contract with initial Supply tokens to each account (this balance
of NtuaTokens will have already been bought from the users)
      function NtuaToken() public{ //constructor function
        balanceOf[0x627306090abaB3A6e1400e9345bC60c78a8BEf57] = 1000 * 1 ether;
        balanceOf[0xf17f52151EbEF6C7334FAD080c5704D77216b732] = 1000 * 1 ether;
        balanceOf[0xC5fdf4076b8F3A5357c5E395ab970B5B54098Fef] = 1000 * 1 ether;
        balanceOf[0x821aEa9a577a9b44299B9c15c88cf3087F3b5544] = 1000 * 1 ether;
        balanceOf[0x0d1d4e623D10F9FBA5Db95830F7d3839406C6AF2] = 1000 * 1 ether;
        balanceOf[0x2932b7A2355D6fecc4b5c0B6BD44cC31df247a2e] = 1000 * 1 ether;
        balanceOf[0x2191eF87E392377ec08E7c08Eb105Ef5448eCED5] = 1000 * 1 ether;
        balanceOf[0x0F4F2Ac550A1b4e2280d04c21cEa7EBD822934b5] = 1000 * 1 ether;
        balanceOf[0x6330A553Fc93768F612722BB8c2eC78aC90B3bbc] = 1000 * 1 ether;
        balanceOf[0x5AEDA56215b167893e80B4fE645BA6d5Bab767DE] = 1000 * 1 ether;
        name = "NtuaToken";
        symbol = "NtuaTok";
        decimals = 18;
      }

      function balanceof(address _owner) external constant returns (uint256
balance){
        return balanceOf[_owner];
```

```
      }

    //Internal transfer, only can be called by this contract
    function _transfer(address _from, address _to, uint _value) internal {
      require(_to != 0x0);    //Prevent transfer to address 0x0
      require(balanceOf[_from] >= _value && balanceOf[_to] + _value >=
balanceOf[_to]);
      //Check if the sender has enough tokens - Check for overflow
      balanceOf[_from] -= _value; //Subtract from the sender
      balanceOf[_to] += _value; //Add the same to the recipient
      emit Transfer(_from,_to,_value);
    }

    function transfer(address _to, uint256 _value) external{
      _transfer(msg.sender,_to,_value);
      //Contract calls the internal method
    }

    function transferFrom(address _from, address _to, uint256 _value) external
returns (bool success){
      _transfer(_from,_to,_value);
      return true;
    }
}
```

Ο πηγαίος κώδικας του συμβολαίου Uploader:

```
pragma solidity ^0.4.18;

contract uploader {

    struct media_ownership {
      address owner1;
      uint percentage1;
      address owner2;
      uint percentage2;
    }
    struct media_content {
      string mediaTitle;
      string artist;
      uint year;
      string url;
      uint price;
    }
    //Creation of two structs to save each media file in the blockchain
    //In the struct album_ownership info regarding the owners of each file and
their compensation for selling the file will be stored
    //In the struct album_content info regarding the media file will be stored

    //mapping for all media (mediaID => media_content)
    mapping(uint => media_ownership[]) mediaOwnershipTable;
    //mediaOwnershipTable[int] array

    //mapping for all media (mediaID => media_content)
    mapping(uint => media_content[]) mediaContentTable;
    //mediaContentTable[int] array
    //Mapping creates an array-like structure for the structs above

    //constructor function
    function uploader() public {
      idStable = 1;
```

```
        //initialization of auto-incremented id used for uploading media files
    }

    //Function for the creator to upload the media content in the Ethereum
Blockchain.
    function createMedia(address o1, uint p1, address o2, uint p2, string url,
uint price, string mediaTitle, string artist, uint year) external {
        mediaOwnershipTable[idStable].push(media_ownership(o1, p1, o2, p2));
        mediaContentTable[idStable].push(media_content(mediaTitle, artist, year,
url, price));
        //use of push function to insert new rows into mappings
        emit MediaCreated(mediaTitle, artist, year, price, idStable);
        idStable++;
        //increment id after the insertion of a media file
    }

    function deleteMedia(address request_er, uint32 id) external{
        require((request_er == mediaOwnershipTable[id][0].owner1) || (request_er
== mediaOwnershipTable[id][0].owner2));
        //check if the account owner who requested the deletion of this specific
media file is one of its owners
        mediaOwnershipTable[id][0].owner1 = 0x0;
        mediaOwnershipTable[id][0].owner2 = 0x0;
        mediaOwnershipTable[id][0].percentage1 = 0;
        mediaOwnershipTable[id][0].percentage2 = 0;
        mediaContentTable[id][0].mediaTitle = "";
        mediaContentTable[id][0].artist = "";
        mediaContentTable[id][0].year = 0;
        mediaContentTable[id][0].url = "";
        mediaContentTable[id][0].price = 0;
        //replace blockchain info of deleted file with dummy info
        emit MediaDeleted(id);
    }

    function updatePrice(address request_er, uint32 id, uint p1, uint p2, uint
price) external{
        require((request_er == mediaOwnershipTable[id][0].owner1) || (request_er
== mediaOwnershipTable[id][0].owner2));
        //check if the account owner who requested the price change of this
specific media file is one of its owners
        mediaOwnershipTable[id][0].percentage1 = p1;
        mediaOwnershipTable[id][0].percentage2 = p2;
        mediaContentTable[id][0].price = price;
        emit MediaChangedPrice(id, mediaOwnershipTable[id][0].percentage1,
mediaOwnershipTable[id][0].percentage2, mediaContentTable[id][0].price);
    }

    function updateURL(address request_er, uint32 id, string url) external{
        require((request_er == mediaOwnershipTable[id][0].owner1) || (request_er
== mediaOwnershipTable[id][0].owner2));
        //check if the account owner who requested the URL change of this
specific media file is one of its owners
        mediaContentTable[id][0].url = url;
        emit MediaChangedURL(id, mediaContentTable[id][0].url);
    }

    //EVENTS
    //emit these events to send information to the html pages
    event MediaCreated(string mediaTitle, string artist, uint year, uint
price, uint id);
    event MediaDeleted(uint32 id);
    event MediaChangedPrice(uint32 id, uint p1, uint p2, uint price);
```

```
      event MediaChangedURL(uint32 id, string url);
      event PaymentInfo(address buyerAddress, address owner1, address owner2,
uint percentage1, uint percentage2, uint32 id, string url);
      event PaymentInfoNtuaToken(address buyerAddress, address owner1, address
owner2, uint percentage1, uint percentage2, uint32 id, string url);

      function buyMedia(uint32 chosenMediaID, address buyerAddress, uint x)
external{
       address owner1 = mediaOwnershipTable[chosenMediaID][0].owner1;
       //retrieve owner1 address from mapping
       address owner2 = mediaOwnershipTable[chosenMediaID][0].owner2;
       //retrieve owner2 address from mapping
       uint percentage1 = mediaOwnershipTable[chosenMediaID][0].percentage1;
       //retrieve percentage1 value from mapping
       uint percentage2 = mediaOwnershipTable[chosenMediaID][0].percentage2;
       //retrieve percentage2 value from mapping
       uint price = mediaContentTable[chosenMediaID][0].price;
       //retrieve price value from mapping
       string url = mediaContentTable[chosenMediaID][0].url;
       //retrieve url string from mapping
       if (x == 1){
          emit PaymentInfo(buyerAddress, owner1, owner2, percentage1,
percentage2, chosenMediaID, url);
       }
       else{
          emit PaymentInfoNtuaToken(buyerAddress, owner1, owner2, percentage1,
percentage2, chosenMediaID, url);
       }

       //initialize variables
       uint32 j;
       uint32 idStable;
}
```

2 Ιστοσελίδες

Σε αυτήν την ενότητα βρίσκεται ο κώδικας των ιστοσελίδων γραμμένος στις γλώσσες html και Javascript.

Ο πηγαίος κώδικας της ιστοσελίδας Creator Upload (indexcreator.html):

```
<!DOCTYPE html>
<html lang="en">
<head>
      <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css">

              <style>
                      ul {
                          list-style-type: none;
                          margin: 0;
                          padding: 0;
                          overflow: hidden;
                          background-color: #333;
                      }

                      li {
```

```
                        float: left;
                    }

                    li a {
                        display: block;
                        color: white;
                        text-align: center;
                        padding: 14px 16px;
                        text-decoration: none;
                    }

                    li a:hover:not(.active) {
                        background-color: #111;
                    }

                    .active {
                        background-color: #4CAF50;
                    }
                    </style>

        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <title>Upload</title>

        <script src="./node_modules/web3/dist/web3.min.js"></script>
    </head>
    <body>

    <ul>
      <li><a href="./indexproject.html">Media Content Localhost</a></li>
      <li><a href="./indexbuy.html">Buy Media files</a></li>
      <li><a href="./indexcreator.html">Creator Upload</a></li>
      <li style="float:right"><a class="active" href="#about">About</a></li>
    </ul>

    <h2>**********Media Files Store
Dapp**********</h2>

    <div class=" col-md-offset-4 col-md-6">

    <h3>Upload Media Files || Main Creator</h3>

    <input id="owner1" type="text" placeholder="Address of owner1"></input>
    <input id="percentage1" type="text" placeholder="percentage1
integer"></input></p>
    <input id="owner2" type="text" placeholder="Address of owner2"></input>
    <input id="percentage2" type="text" placeholder="percentage2
integer"></input></p>

    <input id="url" type="text" placeholder="URL Address"></input>
    <input id="price" type="text" placeholder="price integer (sum of
percentages)"></input>
    <input id="mediaTitle" type="text" placeholder="Media file title"></input>
    <input id="artist" type="text" placeholder="Artist"></input>
    <input id="year" type="text" placeholder="Year"></input></p>

    <button type="button" class="btn btn-primary" onclick="Upload()">Upload
file to blockchain</button>
    <button type="button" class="btn btn-primary"
onclick="window.open('http://localhost:8081')" >Upload file to
database</button>
```

```
      <h3>Delete media file</h3>
      <input id="request_er" type="text" placeholder="Address of the person who
requests the deletion"></input>
      <input id="ID1" type="text" placeholder="ID"></input></p>
      <button type="button" class="btn btn-primary"
onclick="Delete()">Delete</button>

      <h3>Update Price</h3>
      <input id="request_er2" type="text" placeholder="Address of the person who
requests the change"></input>
      <input id="ID2" type="text" placeholder="ID"></input></p>
      <input id="percentage1.2" type="text" placeholder="percentage1
integer"></input>
      <input id="percentage2.2" type="text" placeholder="percentage2
integer"></input>
      <input id="price2" type="text" placeholder="NEW price"></input></p>
      <button type="button" class="btn btn-primary"
onclick="UpdatePrice()">Update price</button>   <!--na kano thn synsarthsh edo-
->

      <h3>Update URL</h3>
      <input id="request_er3" type="text" placeholder="Address of the person who
requests the change"></input>
      <input id="ID3" type="text" placeholder="ID"></input></p>
      <input id="url3" type="text" placeholder="NEW URL Address"></input></p>
      <button type="button" class="btn btn-primary" onclick="UpdateURL()">Update
URL</button>    <!--na kano thn synsarthsh edo-->

      <h3>List of uploaded media files:</h3>

      <p id="listOfMedia"></p>

      </div>
      <script>
        var web3 = new Web3(new
Web3.providers.HttpProvider("http://localhost:7545"));
              console.log(web3);
              web3.eth.defaultAccount = web3.eth.accounts[0];


              //Creator Contract
              var contractCreatorAbi= [.....];
              var contractCreatorAddress =
"0xf12b5dd4ead5f743c6baa640b0216200e89b60da";

              //**************** Creator Uploading Media ****************//

              const CONTRACTCREATOR =
web3.eth.contract(contractCreatorAbi).at(contractCreatorAddress, (err, ctr) =>
{
                  console.log("Contract variable creacted: "+ctr);
             })

        var display = [];
        var text = "";
      //catch event MediaCreated
        var uploadEvent = CONTRACTCREATOR.MediaCreated();
        uploadEvent.watch(function(error, result){
            if(!error){
               var i = display.length;
```

```
                display[display.length] = result.args.id;
                display[display.length] = result.args.mediaTitle;
                display[display.length] = result.args.artist;
                display[display.length] = result.args.year;
                display[display.length] = result.args.price;
                text += ("ID: " + display[i] + ',  Title: ' + display[i+1] + ',
Artist: ' + display[i+2] + ', Year: ' + display[i+3] + ', Price: ' +
display[i+4]+ "<br>");
                document.getElementById("listOfMedia").innerHTML = text;
            }
            else {
                console.log(error);
            }
       });


      var flagDeleted = 0;
      //catch event MediaDeleted
      var deleteEvent = CONTRACTCREATOR.MediaDeleted();
      deleteEvent.watch(function(error, result){
        if(!error){
          var m = display.length;
          deletedID = result.args.id;
          for(n=0; n<m; n=n+5){   //check if owner1 is zero. If yes, it is a
deleted file so skip it
              var a = display[n];
              var b = deletedID;
              if(a==0){
                  continue;
              }
              var y = a.equals(b);
              if(y){
                  flagDeleted = 1;
                  display[n]=0;
                  display[n+1]="-";
                  display[n+2]="-";
                  display[n+3]=0;
                  display[n+4]=0;
                  break;
              }
              else{
                  continue;
              }
          }
          if(flagDeleted == 1){
              displayMediaFromScratch();
              flagDeleted = 0;
          }
        }
        else {
          console.log(error);
        }
      });


      var  flagPriceUpdated = 0;
      //catch event MediaChangedPrice
      var updatePriceEvent = CONTRACTCREATOR.MediaChangedPrice();
      updatePriceEvent.watch(function(error, result){
        if(!error){
           var g = display.length;
           var tempID = result.args.id;
```

```
          var tempP1 = result.args.p1;
          var tempP2 = result.args.p2;
          var tempPrice = result.args.price;
          for(s=0; s<g; s = s + 5){
             var a = display[s];
             if(a==0){
                continue;
             }
             if(display[s].equals(tempID)){
                flagPriceUpdated = 1;
                display[s+4] = tempPrice;
                break;
             }
             else{
                continue;
             }
          }
          if(flagPriceUpdated == 1){
             displayMediaFromScratch();
             flagPriceUpdated = 0;
          }
        }
        else {
          console.log(error);
        }
      });


      //Update Price Button
      function UpdatePrice() {
        var request_er2 = document.getElementById("request_er2").value;
        var ID2 = document.getElementById("ID2").value;
        var percentage1number = document.getElementById("percentage1.2").value;
        var percentage2number = document.getElementById("percentage2.2").value;
        var price = parseInt(document.getElementById("price2").value, 10);
        CONTRACTCREATOR.updatePrice(request_er2, ID2, percentage1number,
percentage2number, price, {gas: 500000});
        alert("Price updated in media file with ID: "+ ID2);
      }

      //Update URL Button
      function UpdateURL() {
        var request_er3 = document.getElementById("request_er3").value;
        var ID3 = document.getElementById("ID3").value;
        var url3 = document.getElementById("url3").value;
        CONTRACTCREATOR.updateURL(request_er3, ID3, url3, {gas: 500000});
        alert("URL updated in media file with ID: "+ ID3);
      }

      //Upload Button
      function Upload() {
        var owner1address = document.getElementById("owner1").value;
        var percentage1number = document.getElementById("percentage1").value;
        var owner2address = document.getElementById("owner2").value;
        var percentage2number = document.getElementById("percentage2").value;
        var url = document.getElementById("url").value;
        var price = parseInt(document.getElementById("price").value, 10);
        var mediaTitle = document.getElementById("mediaTitle").value;
        var artist = document.getElementById("artist").value;
        var year = document.getElementById("year").value;
```

```
       CONTRACTCREATOR.createMedia(owner1address, percentage1number,
owner2address, percentage2number, url, price, mediaTitle, artist, year,{gas:
500000});
       alert("Upload done.");
      }
      //Delete Button
      function Delete(){
       var request_er = document.getElementById("request_er").value;
       var ID = document.getElementById("ID1").value;
       CONTRACTCREATOR.deleteMedia(request_er, ID, {gas: 500000});
       alert("Media file with ID: "+ID+" was deleted.");
      }

      function displayMediaFromScratch(){
       var z = display.length;
       text = "";
       for(i=0; i<z; i= i+5){
           text += ("ID: " + display[i] + ',  Title: ' + display[i+1] + ',
Artist: ' + display[i+2] + ', Year: ' + display[i+3] + ', Price: ' +
display[i+4]+ "<br>");
       }
       document.getElementById("listOfMedia").innerHTML = text;
      }

              //************************************************************

      //==================================== Top Button ==================
              var input = document.getElementById('input');
              var button = document.getElementById('button');
              var output = document.getElementById('output');

          </script>

      </body>
      </html>
```

Ο πηγαίος κώδικας της ιστοσελίδας Buy media files (indexbuy.html):

```
<!DOCTYPE html>
<html lang="en">
<head>
      <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css">

      <style>
                      ul {
                           list-style-type: none;
                           margin: 0;
                           padding: 0;
                           overflow: hidden;
                           background-color: #333;
                       }

                       li {
                           float: left;
                       }

                       li a {
                           display: block;
                           color: white;
                           text-align: center;
```

```
                        padding: 14px 16px;
                        text-decoration: none;
                    }

                    li a:hover:not(.active) {
                        background-color: #111;
                    }

                    .active {
                        background-color: #4CAF50;
                    }
                    </style>

          <meta charset="UTF-8">
          <meta name="viewport" content="width=device-width, initial-scale=1.0">
          <meta http-equiv="X-UA-Compatible" content="ie=edge">
          <title>Buy</title>

          <script src="./node_modules/web3/dist/web3.min.js"></script>

          <meta charset="utf-8">
          <meta name="viewport" content="width=device-width, initial-scale=1">
          <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css">
          <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script
>
          <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.0/umd/popper.min.js"
></script>
          <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/js/bootstrap.min.js"></scr
ipt>


      </head>

      <body>

      <ul>
        <li><a href="./indexproject.html">Media Content Localhost</a></li>
        <li><a href="./indexbuy.html">Buy Media files</a></li>
        <li><a href="./indexcreator.html">Creator Upload</a></li>
        <li style="float:right"><a class="active" href="#about">About</a></li>
      </ul>

      <h2>************************Buy Media files************************</h2>

      <div class=" col-md-offset-4 col-md-6">

      <h3>Select from all the available media items:</h3>
      <p id="listOfMedia"></p>

      <h3>Type the ID of the selected media item and your address:</h3>

      <div class="form-group" >
        <label for="ID">Media content's id:</label>
        <input type="text" class="form-control" id="ID" placeholder="Enter id">

        <label for="buyer">Buyer's address:</label>
        <input type="text" class="form-control" id="buyer" placeholder="Enter
address">
```

```
      </div>
      <button type="button" class="btn btn-primary" onclick="Buy()">Buy using
Ether</button>
      <button type="button" class="btn btn-primary" onclick="BuyNtuaToken()">Buy
using NtuaToken</button>

      </div>

      <script>
        var web3 = new Web3(new
Web3.providers.HttpProvider("http://localhost:7545"));
        console.log(web3);
        web3.eth.defaultAccount = web3.eth.accounts[0];

      //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Contracts Initialization
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~//

              //-----------------NtuaToken Contract Address & Abi------------//
        var contractNtuaTokenAbi = […];
        var contractNtuaTokenAddress =
"0x8cdaf0cd259887258bc13a92c0a6da92698644c0";

              //-----------------Creator Contract Address & Abi--------------//
        var contractCreatorAbi = […];
        var contractCreatorAddress =
"0xf12b5dd4ead5f743c6baa640b0216200e89b60da";

              //**************** Creator Uploading Media *****************//
        const CONTRACTNTUA =
web3.eth.contract(contractNtuaTokenAbi).at(contractNtuaTokenAddress, (err, ctr)
=> {
        console.log("Contract variable creacted: "+ctr);
      })

        const CONTRACTCREATOR =
web3.eth.contract(contractCreatorAbi).at(contractCreatorAddress, (err, ctr) =>
{
        console.log("Contract variable creacted: "+ctr);
      })
        var display = [];
        var text = "";
      //catch event MediaCreated
        var uploadEvent = CONTRACTCREATOR.MediaCreated();
        uploadEvent.watch(function(error, result){
            if(!error){
               var i = display.length;
                display[display.length] = result.args.id;
                display[display.length] = result.args.mediaTitle;
                display[display.length] = result.args.artist;
                display[display.length] = result.args.year;
                display[display.length] = result.args.price;
                text += ("ID: " + display[i] + ',  Title: ' + display[i+1] + ',
Artist: ' + display[i+2] + ', Year: ' + display[i+3] + ', Price: ' +
display[i+4]+ "<br>");
                document.getElementById("listOfMedia").innerHTML = text;
             }
             else {
                console.log(error);
             }
       });
```

```
      var flagDeleted = 0;
      //catch event MediaDeleted
      var deleteEvent = CONTRACTCREATOR.MediaDeleted();
      deleteEvent.watch(function(error, result){
       if(!error){
         var m = display.length;
         deletedID = result.args.id;
         for(n=0; n<m; n=n+5){   //check if owner1 is zero. If yes, it is a
deleted file so skip it
            var a = display[n];
            var b = deletedID;
            if(a==0){
               continue;
            }
            var y = a.equals(b);
            if(y){
                flagDeleted = 1;
                display[n]=0;
                display[n+1]="-";
                display[n+2]="-";
                display[n+3]=0;
                display[n+4]=0;
                break;
            }
            else{
                continue;
            }
         }
         if(flagDeleted == 1){
            displayMediaFromScratch();
            flagDeleted = 0;
         }
       }
       else {
         console.log(error);
       }
      });

      var  flagPriceUpdated = 0;
      //catch event MediaChangedPrice
      var updatePriceEvent = CONTRACTCREATOR.MediaChangedPrice();
      updatePriceEvent.watch(function(error, result){
       if(!error){
          var g = display.length;
          var tempID = result.args.id;
          var tempP1 = result.args.p1;
          var tempP2 = result.args.p2;
          var tempPrice = result.args.price;
          for(s=0; s<g; s = s + 5){
             var a = display[s];
             if(a==0){
                continue;
             }
             if(display[s].equals(tempID)){
                flagPriceUpdated = 1;
                display[s+4] = tempPrice;
                break;
             }
             else{
                continue;
             }
          }
```

```
          if(flagPriceUpdated == 1){
             displayMediaFromScratch();
             flagPriceUpdated = 0;
          }
        }
        else {
          console.log(error);
        }
      });

      var paymentInfoEvent = CONTRACTCREATOR.PaymentInfo();
      paymentInfoEvent.watch(function(error, result){
        if(!error){
            var tempBuyerAddress = result.args.buyerAddress;
            var tempowner1 = result.args.owner1;
            var tempowner2 = result.args.owner2;
            var tempP1 = result.args.percentage1;
            var tempP2 = result.args.percentage2;
            var tempid = result.args.id;
            var tempurl = result.args.url;
            var amountToSend1 =web3.toWei(tempP1, "ether");
            var amountToSend2 =web3.toWei(tempP2, "ether");
            var balance = web3.eth.getBalance(tempBuyerAddress);
            if(balance - amountToSend1 - amountToSend2 >= 0){
                web3.eth.sendTransaction({from:tempBuyerAddress, to:tempowner1,
value:amountToSend1});
                web3.eth.sendTransaction({from:tempBuyerAddress, to:tempowner2,
value:amountToSend2});
                alert("Please access the file that you have bought from the
following url: "+ tempurl);
            }
            else{
                alert("Account "+ tempBuyerAddress+" does not have enough Ether
for this Media item.");
            }
        }
        else {
            console.log(error);
        }
      });

      var updatePriceEvent = CONTRACTCREATOR.MediaChangedPrice();
      updatePriceEvent.watch(function(error, result){
        if(!error){
            var g = display.length;
            var tempID = result.args.id;
            var tempP1 = result.args.p1;
            var tempP2 = result.args.p2;
            var tempPrice = result.args.price;
            for(s=0; s<g; s = s + 5){
                var a = display[s];
                   if(a==0){
                      continue;
                   }
                   if(display[s].equals(tempID)){
                      flagPriceUpdated = 1;
                      display[s+4] = tempPrice;
                      break;
                   }
                   else{
                      continue;
                   }
```

```
            }
            if(flagPriceUpdated == 1){
                displayMediaFromScratch();
                flagPriceUpdated = 0;
            }
        }
        else {
            console.log(error);
        }
      });


      var paymentInfoEventNtuaToken = CONTRACTCREATOR.PaymentInfoNtuaToken();
      paymentInfoEventNtuaToken.watch(function(error, result){
        if(!error){
            var tempBuyerAddress = result.args.buyerAddress;
            var tempowner1 = result.args.owner1;
            var tempowner2 = result.args.owner2;
            var tempP1 = result.args.percentage1;
            var tempP2 = result.args.percentage2;
            var tempid = result.args.id;
            var tempurl = result.args.url;
            var butAmountTokens1 = parseInt(tempP1, 10);
            var butAmountTokens2 = parseInt(tempP2, 10);

            CONTRACTNTUA.transferFrom(tempBuyerAddress, tempowner1,
web3.toWei(butAmountTokens1, "ether"));
            CONTRACTNTUA.transferFrom(tempBuyerAddress, tempowner2,
web3.toWei(butAmountTokens2, "ether"));
            alert("Please access the file that you have bought from the following
url: "+ tempurl);
        }
        else {
            console.log(error);
        }
      });

      //buy using ether
      function Buy(){
        var ID = document.getElementById("ID").value;
        var buyer = document.getElementById("buyer").value;
        //from the smart contract I want to retrieve the payment info, not to do
the transaction
        CONTRACTCREATOR.buyMedia(ID, buyer, 1, {gas: 50000});
      }

      //buy using NtuaToken
      function BuyNtuaToken(){
        var ID = document.getElementById("ID").value;
        var buyer = document.getElementById("buyer").value;
        //from the smart contract I want to retrieve the payment info, not to do
the transaction
        CONTRACTCREATOR.buyMedia(ID, buyer, 2, {gas: 50000});
      }

      function displayMediaFromScratch(){
        var z = display.length;
        text = "";
        for(i=0; i<z; i= i+5){
            text += ("ID: " + display[i] + ',  Title: ' + display[i+1] + ',
Artist: ' + display[i+2] + ', Year: ' + display[i+3] + ', Price: ' +
display[i+4]+ "<br>");
```

```
        }
        document.getElementById("listOfMedia").innerHTML = text;
    }

            //*********************************************************

    //=================================== Top Button ==================
    var input = document.getElementById('input');
    var button = document.getElementById('button');
    var output = document.getElementById('output');

    </script>
    </body>
    </html>
```